PDF

Release 1.0.0a29

oceanai

Apr 17, 2024

1 Research data 3

2 Certificate of state registration of a computer program 5

3 Certificate of state registration of a database 7

4 Publications 9

5 Indexing 323

Python Module Index 325

Index 327

OCEAN-AI is an open-source library consisting of a set of algorithms for intellectual analysis of human behavior based on multimodal data for automatic personality traits (PT) assessment. The library evaluates five PT: **O**penness to experience, **C**onscientiousness, **E**xtraversion, **A**greeableness, Non-**N**euroticism.

OCEAN-AI includes four main algorithms:

1. Audio Information Analysis Algorithm (AIA).
2. Video Information Analysis Algorithm (VIA).
3. Text Information Analysis Algorithm (TIA).
4. Multimodal Information Fusion Algorithm (MIF).

The AIA, VIA and TIA algorithms implement the functions of strong artificial intelligence (AI) in terms of complexing acoustic, visual and linguistic features built on different principles (hand-crafted and deep features), i.e. these algorithms implement the approaches of composite (hybrid) AI. The necessary pre-processing is carried out in the algorithms audio, video and text information, the calculation of acoustic, visual and linguistic features and the output of PT predictions based on them.

The MIF algorithm is a link between three information analysis algorithms (AIA, VIA and TIA). This algorithm performs a weighted neural network combination of prediction personality traits obtained using the AIA, VIA and TIA algorithms.

OCEAN-AI provides examples of solving practical tasks based on obtained PT scores:

1. **Ranking potential candidates by professional responsibilities by:**
    1. professional groups;
    2. professional skills.
2. **Predicting consumer preferences for industrial goods through:**
    1. the example of car characteristics;
    2. the example of mobile device application categories.
3. **Forming effective work teams:**
    1. finding a suitable junior colleague;
    2. finding a suitable senior colleague.

In addition to the main task - multimodal personality traits assessment, the features implemented in OCEAN-AI features will allow researchers to solve other problems of analyzing human behavior, for example, recognizing his affective states.

OCEAN-AI uses the latest open-source libraries for audio, video and text processing: librosa, openSMILE, openCV, mediapipe, transformers.

OCEAN-AI is written in the python programming language. Neural network models are implemented and trained using an open-source library code TensorFlow.

ONE

RESEARCH DATA

The OCEAN-AI library was tested on two corpora:

1. Общедоступном и крупномаштабном корпусе First Impressions V2.
2. On the first publicly available Russian-language Multimodal Personality Traits Assessment (MuPTA) corpus.

TWO

CERTIFICATE OF STATE REGISTRATION OF A COMPUTER PROGRAM

Library of algorithms for intelligent analysis of human behavior based on multimodal data, providing human's personality traits assessment to perform professional duties (OCEAN-AI)

THREE

CERTIFICATE OF STATE REGISTRATION OF A DATABASE

MuPTA - Multimodal Personality Traits Assessment Corpus

FOUR

PUBLICATIONS

4.1 Journals

```
@article{ryumina22_neurocomputing,
    author = {Elena Ryumina and Denis Dresvyanskiy and Alexey Karpov},
    title = {In Search of a Robust Facial Expressions Recognition Model: A Large-Scale Visual Cross-Corpus Study},
    journal = {Neurocomputing},
    volume = {514},
    pages = {435-450},
    year = {2022},
    doi = {https://doi.org/10.1016/j.neucom.2022.10.013},
}
```

```
@article{ryumina24_eswa,
    author = {Elena Ryumina and Maxim Markitantov and Dmitry Ryumin and Alexey Karpov},
    title = {OCEAN-AI Framework with EmoFormer Cross-Hemiface Attention Approach for Personality Traits Assessment},
    journal = {Expert Systems with Applications},
    volume = {239},
    pages = {122441},
    year = {2024},
    doi = {https://doi.org/10.1016/j.eswa.2023.122441},
}
```

4.2 Conferences

```
@inproceedings{ryumina23_interspeech,
    author = {Elena Ryumina and Dmitry Ryumin and Maxim Markitantov and Heysem Kaya and Alexey Karpov},
    title = {Multimodal Personality Traits Assessment (MuPTA) Corpus: The Impact of Spontaneous and Read Speech},
    year = {2023},
    booktitle = {INTERSPEECH},
    pages = {4049--4053},
    doi = {https://doi.org/10.21437/Interspeech.2023-1686},
}
```

Supported by The study is supported by the Research Center Strong Artificial Intelligence in Industry of ITMO University.

4.2.1 Quick Start

Installation and Update

Installation with PyPi

```
pip install oceanai
```

Update with PyPi

```
pip install --upgrade oceanai
```

Dependencies

Таблица1: Installed automatically

The library	Recommended Version	Current version
ipython	8.7.0	
jupyterlab	3.5.0	
tensorflow	2.11.0	
keras	2.11.0	
Keras-Applications	1.0.8	
numpy	1.23.5	
scipy	1.9.3	
pandas	1.5.2	
requests	2.28.1	
opensmile	2.4.1	
librosa	0.9.2	
audioread	3.0.0	
scikit-learn	1.1.3	
opencv-contrib-python	4.6.0.66	
pymediainfo	6.0.1	
mediapipe	0.9.0	
liwc	0.5.0	
transformers	4.36.0	
sentencepiece	0.1.99	
torch	2.0.1	
torchaudio	2.0.2	
sacremoses	0.0.1	

Examples

Solution of practical tasks

Solution of practical task 1

Task: Ranking of potential candidates by professional responsibilities

The solution of the practical task is performed in two stages. At the first stage it is necessary to use the OCEAN-AI library to obtain predictions (personality traits scores). The second step is to use the methods _candidate_ranking and _priority_skill_calculation from the OCEAN-AI library to solve the presented practical task. Examples of the results of the work and implementation are presented below.

Thus, the OCEAN-AI library provides tools to analyze the personality traits of candidates and their suitability for the position, which can significantly improve the recruitment process and help to make more

FI V2

[2]:
```
# Import required tools
import os
import pandas as pd

# Module import
from oceanai.modules.lab.build import Run

# Creating an instance of a class
_b5 = Run(lang = 'en')

# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

corpus = 'fi'

# Building audio models
res_load_model_hc = _b5.load_audio_model_hc()
res_load_model_nn = _b5.load_audio_model_nn()

# Loading audio model weights
url = _b5.weights_for_big5_['audio'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_audio_model_weights_hc(url = url)

url = _b5.weights_for_big5_['audio'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_audio_model_weights_nn(url = url)

# Loading audio model weights
```

(continues on next page)

(continued from previous page)

```
res_load_model_hc = _b5.load_video_model_hc(lang='en')
res_load_model_deep_fe = _b5.load_video_model_deep_fe()
res_load_model_nn = _b5.load_video_model_nn()

# Loading video model weights
url = _b5.weights_for_big5_['video'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_video_model_weights_hc(url = url)

url = _b5.weights_for_big5_['video'][corpus]['fe']['sberdisk']
res_load_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(url = url)

url = _b5.weights_for_big5_['video'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_video_model_weights_nn(url = url)

# Loading a dictionary with hand-crafted features (text modality)
res_load_text_features = _b5.load_text_features()

# Building text models
res_setup_translation_model = _b5.setup_translation_model()
res_setup_translation_model = _b5.setup_bert_encoder()
res_load_text_model_hc_fi = _b5.load_text_model_hc(corpus=corpus)
res_load_text_model_nn_fi = _b5.load_text_model_nn(corpus=corpus)

# Loading text model weights
url = _b5.weights_for_big5_['text'][corpus]['hc']['sberdisk']
res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(url = url)

url = _b5.weights_for_big5_['text'][corpus]['nn']['sberdisk']
res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(url = url)

# Building model for multimodal information fusion
res_load_avt_model_b5 = _b5.load_avt_model_b5()

# Loading model weights for multimodal information fusion
url = _b5.weights_for_big5_['avt'][corpus]['b5']['sberdisk']
res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(url = url)

PATH_TO_DIR = './video_FI/'
PATH_SAVE_VIDEO = './video_FI/test/'

_b5.path_to_save_ = PATH_SAVE_VIDEO

# Loading 10 test files from the First Impressions V2 corpus
# URL: https://chalearnlap.cvc.uab.cat/dataset/24/description/
domain = 'https://download.sberdisk.ru/download/file/'
tets_name_files = [
    '429713680?token=FqHdMLSSh7zYSZt&filename=_plk5k7PBEg.003.mp4',
    '429713681?token=Hz9b4lQkrLfic33&filename=be0DQawtVkE.002.mp4',
    '429713683?token=EgUXS9Xs8xHm5gz&filename=2d6btbaNdfo.000.mp4',
    '429713684?token=1U26753kmPYdIgt&filename=300gK3CnzW0.003.mp4',
    '429713685?token=LyigAWLTzDNwKJO&filename=300gK3CnzW0.001.mp4',
    '429713686?token=EpfRbCKHyuc4HPu&filename=cLaZxEf1nE4.004.mp4',
```

(continues on next page)

(continued from previous page)

```
    '429713687?token=FNTkwqBr4jOS95l&filename=g24JGYuT74A.004.mp4',
    '429713688?token=qDT95nz7hfm2Nki&filename=JZNMxa3OKHY.000.mp4',
    '429713689?token=noLguEGXDpbcKhg&filename=nvlqJbHk_Lc.003.mp4',
    '429713679?token=9L7RQ0hgdJlcek6&filename=4vdJGgZpj4k.003.mp4'
]

for curr_files in tets_name_files:
    _b5.download_file_from_url(url = domain + curr_files, out = True)

# Getting scores
_b5.path_to_dataset_ = PATH_TO_DIR # Dataset directory
_b5.ext_ = ['.mp4'] # Search file extensions

# Full path to the file with ground truth scores for accuracy calculation
url_accuracy = _b5.true_traits_[corpus]['sberdisk']

_b5.get_avt_predictions(url_accuracy = url_accuracy, lang = 'en')
```

[2023-12-16 18:42:02] Feature extraction (hand-crafted and deep) from text ...

[2023-12-16 18:42:05] Getting scores and accuracy calculation (multimodal fusion) ...

10 from 10 (100.0%) ... GitHub:nbsphinx-math:*OCEANAI* _guide:nbsphinx-math:*notebooks* _FI:nbsphinx-math:*test* _plk5k7PBEg.003.mp4
...

Person ID	Path	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
1	2d6btbaNdfo.000.mp4	0.581159	0.628822	0.466609	0.622129	0.553832
2	300gK3CnzW0.001.mp4	0.463991	0.418851	0.41301	0.493329	0.423093
3	300gK3CnzW0.003.mp4	0.454281	0.415049	0.39189	0.485114	0.420741
4	4vdJGgZpj4k.003.mp4	0.588461	0.643233	0.530789	0.603038	0.593398
5	be0DQawtVkE.002.mp4	0.633433	0.533295	0.523742	0.608591	0.588456
6	cLaZxEf1nE4.004.mp4	0.636944	0.542386	0.558461	0.570975	0.558983
7	g24JGYuT74A.004.mp4	0.531518	0.376987	0.393309	0.4904	0.447881
8	JZNMxa3OKHY.000.mp4	0.610342	0.541418	0.563163	0.595013	0.569461
9	nvlqJbHk_Lc.003.mp4	0.495809	0.458526	0.414436	0.469152	0.435461
10	_plk5k7PBEg.003.mp4	0.60707	0.591893	0.520662	0.603938	0.565726

[2023-12-16 18:42:05] Trait-wise accuracy ...

	Openness	Conscientiousness	Extraversion	Agreeableness	\

(continues on next page)

(continued from previous page)

```
Metrics
MAE         0.0589              0.0612         0.0864          0.0697
Accuracy    0.9411              0.9388         0.9136          0.9303

          Non-Neuroticism    Mean
Metrics
MAE                0.0582  0.0669
Accuracy           0.9418  0.9331
```

[2023-12-16 18:42:05] Mean absolute errors: 0.0669, average accuracy: 0.9331 ...

Log files saved successfully ...

— Runtime: 64.481 sec. —

[2]:
```
True
```

Thus, the OCEAN-AI library provides tools to analyze the personality traits of candidates and their suitability for the position, which can significantly improve the recruitment process and help to make more objective and systematic decisions when ranking candidates.

The weight coefficients for 5 professions based on scientific articles are proposed:

1) Sajjad H. et al. Personality and Career Choices // African Journal of Business Management. - 2012. – Vol. 6 (6) – pp. 2255-2260.
2) Alkhelil A. H. The Relationship between Personality Traits and Career Choice: A Case Study of Secondary School Students // International Journal of Academic Research in Progressive Education and Development. – 2016. – Vol. 5(2). – pp. 2226-6348.
3) De Jong N. et al. Personality Traits and Career Role Enactment: Career Role Preferences as a Mediator // Frontiers in Psychology. – 2019. – Vol. 10. – pp. 1720.

The user can set their own weights; the sum of the weights must be equal to 100.

[3]:
```
# Loading a dataframe with weights
url = 'https://download.sberdisk.ru/download/file/478675798?token=fF5fNZVpthQlEV0&
↪filename=traits_priority_for_professions.csv'
traits_priority_for_professions = pd.read_csv(url)

traits_priority_for_professions.index.name = 'ID'
traits_priority_for_professions.index += 1
traits_priority_for_professions.index = traits_priority_for_professions.index.map(str)

traits_priority_for_professions
```

[3]:
```
                              Profession  Openness  Conscientiousness  \
ID
1                     Managers/executives        15                 35
2                        Entrepreneurship        30                 30
3    Social/Non profit making professions         5                  5
4                Public sector professions        15                 50
5   Scientists/researchers, and engineers        50                 15

    Extraversion  Agreeableness  Non-Neuroticism
ID
1             15             30                5
2              5              5               30
```

(continues on next page)

(continued from previous page)

```
3              35              35              20
4              15              15               5
5               5              15              15
```

Ranking of candidates for the position of engineer

[4]:
```
weights = traits_priority_for_professions.iloc[4].values[1:]
weights = list(map(int, weights))

_b5._candidate_ranking(
    weigths_openness = weights[0],
    weigths_conscientiousness = weights[1],
    weigths_extraversion = weights[2],
    weigths_agreeableness = weights[3],
    weigths_non_neuroticism = weights[4],
    out = False
)

_b5._save_logs(df = _b5.df_files_ranking_, name = 'engineer_candidate_ranking_fi_en',␣
↪out = True)

# Optional
df = _b5.df_files_ranking_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON',
↪ 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[4]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	\
5	beODQawtVkE.002.mp4	0.633	0.533	0.524	0.609	0.588	
6	cLaZxEf1nE4.004.mp4	0.637	0.542	0.558	0.571	0.559	
4	4vdJGgZpj4k.003.mp4	0.588	0.643	0.531	0.603	0.593	
10	_plk5k7PBEg.003.mp4	0.607	0.592	0.521	0.604	0.566	
8	JZNMxa3OKHY.000.mp4	0.610	0.541	0.563	0.595	0.569	
1	2d6btbaNdfo.000.mp4	0.581	0.629	0.467	0.622	0.554	
7	g24JGYuT74A.004.mp4	0.532	0.377	0.393	0.490	0.448	
9	nvlqJbHk_Lc.003.mp4	0.496	0.459	0.414	0.469	0.435	
2	300gK3CnzW0.001.mp4	0.464	0.419	0.413	0.493	0.423	
3	300gK3CnzW0.003.mp4	0.454	0.415	0.392	0.485	0.421	

Person ID	Candidate score
5	60.246
6	59.725
4	59.672
10	59.380
8	58.921
1	58.463
7	48.271
9	47.310

(continues on next page)

(continued from previous page)

```
2                    45.294
3                    44.487
```

Ranking of candidates for the position of manager

[5]:
```
weights = traits_priority_for_professions.iloc[0].values[1:]
weights = list(map(int, weights))

_b5._candidate_ranking(
    weigths_openness = weights[0],
    weigths_conscientiousness = weights[1],
    weigths_extraversion = weights[2],
    weigths_agreeableness = weights[3],
    weigths_non_neuroticism = weights[4],
    out = False
)

_b5._save_logs(df = _b5.df_files_ranking_, name = 'executive_candidate_ranking_fi_en',␣
↪out = True)

# Optional
df = _b5.df_files_ranking_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON',
↪ 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[5]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Candidate score
4	4vdJGgZpj4k.003.mp4	0.588	0.643	0.531	0.603	0.593	60.360
1	2d6btbaNdfo.000.mp4	0.581	0.629	0.467	0.622	0.554	59.158
10	_plk5k7PBEg.003.mp4	0.607	0.592	0.521	0.604	0.566	58.579
8	JZNMxa3OKHY.000.mp4	0.610	0.541	0.563	0.595	0.569	57.250
5	be0DQawtVkE.002.mp4	0.633	0.533	0.524	0.609	0.588	57.223
6	cLaZxEf1nE4.004.mp4	0.637	0.542	0.558	0.571	0.559	56.839
9	nvlqJbHk_Lc.003.mp4	0.496	0.459	0.414	0.469	0.435	45.954
2	300gK3CnzW0.001.mp4	0.464	0.419	0.413	0.493	0.423	44.730
7	g24JGYuT74A.004.mp4	0.532	0.377	0.393	0.490	0.448	
3	300gK3CnzW0.003.mp4	0.454	0.415	0.392	0.485	0.421	

(continues on next page)

(continued from previous page)

```
7                    44.018
3                    43.876
```

To rank candidates by skills, two correlation coefficients must be set for each personality trait and skill, as well as a threshold for the polarity of the traits. These coefficients should show how a person's trait score changes when it is above or below a given trait polarity threshold.

As an example, the use of correlation coefficients between five traits and four professional skills presented in the article is suggested:

1) Wehner C., de Grip A., Pfeifer H. Do recruiters select workers with different personality traits for different tasks? A discrete choice experiment // Labour Economics. - 2022. - vol. 78. - pp. 102186.

There are 4 professional skills presented:

1) Analytical. The ability to effectively solve new problems that require in-depth analysis.
2) Interactive. The ability to persuade and compromise with clients and colleagues.
3) Routine. The ability to perform routine tasks effectively with accuracy and attention to detail.
4) Non-Routine. The ability to respond to and solve problems that have no set order, demonstrating adaptability and creative problem solving skills.

The users can set their own correlation coefficients and rank candidates by other professional skills.

Ranking candidates by professional skills

[6]:
```
# Loading a dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478678231?token=0qiZwliLtHWWYMv&
↪filename=professional_skills.csv'
df_professional_skills = pd.read_csv(url)

df_professional_skills.index.name = 'ID'
df_professional_skills.index += 1
df_professional_skills.index = df_professional_skills.index.map(str)

df_professional_skills
```

[6]:

ID	Trait	Score_level	Analytical	Interactive	Routine	Non-Routine
1	Openness	high	0.082	0.348	0.571	0.510
2	Openness	low	0.196	0.152	0.148	
3	Conscientiousness	high	0.994	1.333	1.507	
4	Conscientiousness	low	0.241	0.188	0.191	
5	Extraversion	high	0.169	-0.060	0.258	
6	Extraversion	low	0.181	0.135	0.130	
7	Agreeableness	high	1.239	0.964	1.400	
8	Agreeableness	low	0.226	0.180	0.189	
9	Non-Neuroticism	high	0.636	0.777	0.876	
10	Non-Neuroticism	low	0.207	0.159	0.166	

(continues on next page)

(continued from previous page)

```
2         0.218
3         1.258
4         0.267
5         0.017
6         0.194
7         1.191
8         0.259
9         0.729
10        0.238
```

[7]:
```
_b5._priority_skill_calculation(
    correlation_coefficients = df_professional_skills,
    threshold = 0.5,
    out = True
)

_b5._save_logs(df = _b5.df_files_priority_skill_, name = 'skill_candidate_ranking_fi_en',
↪ out = True)

# Optional
df = _b5.df_files_priority_skill_.rename(columns = {'Openness':'OPE', 'Conscientiousness
↪':'CON', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[7]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Analytical	Interactive	Routine	Non-Routine
4	4vdJGgZpj4k.003.mp4	0.588	0.643	0.531	0.603	0.593	0.380	0.415	0.561	0.454
1	2d6btbaNdfo.000.mp4	0.581	0.629	0.467	0.622	0.554	0.376	0.427	0.539	0.465
10	_plk5k7PBEg.003.mp4	0.607	0.592	0.521	0.604	0.566	0.367	0.398	0.543	0.439
5	be0DQawtVkE.002.mp4	0.633	0.533	0.524	0.609	0.588	0.360	0.389	0.534	0.431
8	JZNMxa3OKHY.000.mp4	0.610	0.541	0.563	0.595	0.569	0.357	0.383	0.528	0.425
6	cLaZxEf1nE4.004.mp4	0.637	0.542	0.558	0.571	0.559	0.350	0.379	0.523	0.421
9	nvlqJbHk_Lc.003.mp4	0.496	0.459	0.414	0.469	0.435	0.096	0.074	0.075	0.107
2	300gK3CnzW0.001.mp4	0.464	0.419	0.413	0.493	0.423	0.093	0.072	0.073	0.104
3	300gK3CnzW0.003.mp4	0.454	0.415	0.392	0.485	0.421	0.091	0.071	0.072	0.102
7	g24JGYuT74A.004.mp4	0.532	0.377	0.393	0.490	0.448	0.082	0.094	0.119	0.136

MuPTA (ru)

[9]:
```
import os
import pandas as pd

# Module import
from oceanai.modules.lab.build import Run

# Creating an instance of a class
_b5 = Run(lang = 'en')

corpus = 'mupta'
lang = 'ru'

# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

# Building audio models
res_load_model_hc = _b5.load_audio_model_hc()
res_load_model_nn = _b5.load_audio_model_nn()

# Loading audio model weights
url = _b5.weights_for_big5_['audio'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_audio_model_weights_hc(url = url)

url = _b5.weights_for_big5_['audio'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_audio_model_weights_nn(url = url)

# Building video models
res_load_model_hc = _b5.load_video_model_hc(lang=lang)
res_load_model_deep_fe = _b5.load_video_model_deep_fe()
res_load_model_nn = _b5.load_video_model_nn()

# Loading video model weights
url = _b5.weights_for_big5_['video'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_video_model_weights_hc(url = url)

url = _b5.weights_for_big5_['video'][corpus]['fe']['sberdisk']
res_load_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(url = url)

url = _b5.weights_for_big5_['video'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_video_model_weights_nn(url = url)

# Loading a dictionary with hand-crafted features (text modality)
res_load_text_features = _b5.load_text_features()

# Building text models
res_setup_translation_model = _b5.setup_translation_model()
res_setup_translation_model = _b5.setup_bert_encoder()
res_load_text_model_hc_fi = _b5.load_text_model_hc(corpus=corpus)
res_load_text_model_nn_fi = _b5.load_text_model_nn(corpus=corpus)
```

(continues on next page)

(continued from previous page)

```
# Loading text model weights
url = _b5.weights_for_big5_['text'][corpus]['hc']['sberdisk']
res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(url = url)

url = _b5.weights_for_big5_['text'][corpus]['nn']['sberdisk']
res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(url = url)

# Building model for multimodal information fusion
res_load_avt_model_b5 = _b5.load_avt_model_b5()

# Loading model weights for multimodal information fusion
url = _b5.weights_for_big5_['avt'][corpus]['b5']['sberdisk']
res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(url = url)

PATH_TO_DIR = './video_MuPTA/'
PATH_SAVE_VIDEO = './video_MuPTA/test/'

_b5.path_to_save_ = PATH_SAVE_VIDEO

# Loading 10 test files from the MuPTA corpus
# URL: https://hci.nw.ru/en/pages/mupta-corpus
domain = 'https://download.sberdisk.ru/download/file/'
tets_name_files = [
    '477995979?token=2cvyk7CS0mHx2MJ&filename=speaker_06_center_83.mov',
    '477995980?token=jGPtBPS69uzFU6Y&filename=speaker_01_center_83.mov',
    '477995967?token=zCaRbNB6ht5wMPq&filename=speaker_11_center_83.mov',
    '477995966?token=B1rbinDYRQKrI3T&filename=speaker_15_center_83.mov',
    '477995978?token=dEpVDtZg1EQiEQ9&filename=speaker_07_center_83.mov',
    '477995961?token=o1hVjw8G45q9L9Z&filename=speaker_19_center_83.mov',
    '477995964?token=5K22OAqf673VHPq&filename=speaker_23_center_83.mov',
    '477995965?token=v1LVD2KT1cU7Lpb&filename=speaker_24_center_83.mov',
    '477995962?token=tmaSGyyWLA6XCy9&filename=speaker_27_center_83.mov',
    '477995963?token=bTpo96qNDPcwGqb&filename=speaker_10_center_83.mov',
]

for curr_files in tets_name_files:
    _b5.download_file_from_url(url = domain + curr_files, out = True)

# Getting scores
_b5.path_to_dataset_ = PATH_TO_DIR # Dataset directory
_b5.ext_ = ['.mov'] # Search file extensions

# Full path to the file with ground truth scores for accuracy calculation
url_accuracy = _b5.true_traits_['mupta']['sberdisk']

_b5.get_avt_predictions(url_accuracy = url_accuracy, lang = lang)
```

[2023-12-16 18:51:57] Feature extraction (hand-crafted and deep) from text ...

[2023-12-16 18:52:01] Getting scores and accuracy calculation (multimodal fusion) ...

10 from 10 (100.0%) ... GitHub:nbsphinx-math:*OCEANAI* _guide:nbsphinx-

(continues on next page)

(continued from previous page)

math:*notebooks*_MuPTA:nbsphinx-math:*test*_27_center_83.mov
...

Person ID	Path	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
1	speaker_01_center_83.mov	0.758137	0.693356	0.650108	0.744589	0.488671
2	speaker_06_center_83.mov	0.681602	0.654339	0.607156	0.731282	0.417908
3	speaker_07_center_83.mov	0.666104	0.656836	0.567863	0.685067	0.378102
4	speaker_10_center_83.mov	0.694171	0.596195	0.571414	0.66223	0.348639
5	speaker_11_center_83.mov	0.712885	0.594764	0.571709	0.716696	0.37802
6	speaker_15_center_83.mov	0.664158	0.670411	0.60421	0.696056	0.399842
7	speaker_19_center_83.mov	0.761213	0.652635	0.651028	0.788677	0.459676
8	speaker_23_center_83.mov	0.692788	0.68324	0.616737	0.795205	0.447242
9	speaker_24_center_83.mov	0.705923	0.658382	0.610645	0.697415	0.411988
10	speaker_27_center_83.mov	0.753417	0.708372	0.654608	0.816416	0.504743

[2023-12-16 18:52:01] Trait-wise accuracy ...

Metrics	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism	Mean
MAE	0.0673	0.0789	0.1325	0.102	0.1002	0.0962
Accuracy	0.9327	0.9211	0.8675	0.898	0.8998	0.9038

[2023-12-16 18:52:01] Mean absolute errors: 0.0962, average accuracy: 0.9038 ...

Log files saved successfully ...

— Runtime: 415.41 sec. —

[9]: `True`

Thus, the OCEAN-AI library provides tools to analyze the personality traits of candidates and their suitability for the position, which can significantly improve the recruitment process and help to make more objective and systematic decisions when ranking candidates.

The weight coefficients for 5 professions based on scientific articles are proposed:

1) Sajjad H. et al. Personality and Career Choices // African Journal of Business Management. - 2012. – Vol. 6 (6) – pp. 2255-2260.

2) Alkhelil A. H. The Relationship between Personality Traits and Career Choice: A Case Study of

Secondary School Students // International Journal of Academic Research in Progressive Education and Development. – 2016. – Vol. 5(2). – pp. 2226-6348.

3) De Jong N. et al. Personality Traits and Career Role Enactment: Career Role Preferences as a Mediator // Frontiers in Psychology. – 2019. – Vol. 10. – pp. 1720.

The user can set their own weights; the sum of the weights must be equal to 100.

[10]:
```
# Loading a dataframe with weights
url = 'https://download.sberdisk.ru/download/file/478675798?token=fF5fNZVpthQlEV0&
↪filename=traits_priority_for_professions.csv'
traits_priority_for_professions = pd.read_csv(url)

traits_priority_for_professions.index.name = 'ID'
traits_priority_for_professions.index += 1
traits_priority_for_professions.index = traits_priority_for_professions.index.map(str)

traits_priority_for_professions
```

[10]:

ID	Profession	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
1	Managers/executives	15	35	15	30	5
2	Entrepreneurship	30	30	5	5	30
3	Social/Non profit making professions	5	5	35	35	20
4	Public sector professions	15	50	15	15	5
5	Scientists/researchers, and engineers	50	15	5	15	15

Ranking of candidates for the position of engineer

[11]:
```
weights = traits_priority_for_professions.iloc[4].values[1:]
weights = list(map(int, weights))

_b5._candidate_ranking(
    weigths_openness = weights[0],
    weigths_conscientiousness = weights[1],
    weigths_extraversion = weights[2],
    weigths_agreeableness = weights[3],
    weigths_non_neuroticism = weights[4],
    out = False
)

_b5._save_logs(df = _b5.df_files_ranking_, name = 'engineer_candidate_ranking_mupta_ru',␣
↪out = True)

# Optional
```

(continues on next page)

(continued from previous page)

```
df = _b5.df_files_ranking_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON',
↪ 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[11]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Candidate score
10	speaker_27_center_83.mov	0.753	0.708	0.655	0.816	0.505	71.387
1	speaker_01_center_83.mov	0.758	0.693	0.650	0.745	0.489	70.057
7	speaker_19_center_83.mov	0.761	0.653	0.651	0.789	0.460	69.831
8	speaker_23_center_83.mov	0.693	0.683	0.617	0.795	0.447	66.608
9	speaker_24_center_83.mov	0.706	0.658	0.611	0.697	0.412	64.866
2	speaker_06_center_83.mov	0.682	0.654	0.607	0.731	0.418	64.169
5	speaker_11_center_83.mov	0.713	0.595	0.572	0.717	0.378	63.845
6	speaker_15_center_83.mov	0.664	0.670	0.604	0.696	0.400	62.724
3	speaker_07_center_83.mov	0.666	0.657	0.568	0.685	0.378	61.945
4	speaker_10_center_83.mov	0.694	0.596	0.571	0.662	0.349	61.672

Ranking of candidates for the position of manager

[12]:

```
weights = traits_priority_for_professions.iloc[0].values[1:]
weights = list(map(int, weights))

_b5._candidate_ranking(
    weigths_openness = weights[0],
    weigths_conscientiousness = weights[1],
    weigths_extraversion = weights[2],
    weigths_agreeableness = weights[3],
    weigths_non_neuroticism = weights[4],
    out = False
)

_b5._save_logs(df = _b5.df_files_ranking_, name = 'executive_candidate_ranking_mupta_ru',
↪ out = True)

# Optional
df = _b5.df_files_ranking_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON',
```

(continues on next page)

(continued from previous page)

```
↪ 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[12]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	\
10	speaker_27_center_83.mov	0.753	0.708	0.655	0.816	0.505	
1	speaker_01_center_83.mov	0.758	0.693	0.650	0.745	0.489	
7	speaker_19_center_83.mov	0.761	0.653	0.651	0.789	0.460	
8	speaker_23_center_83.mov	0.693	0.683	0.617	0.795	0.447	
2	speaker_06_center_83.mov	0.682	0.654	0.607	0.731	0.418	
9	speaker_24_center_83.mov	0.706	0.658	0.611	0.697	0.412	
6	speaker_15_center_83.mov	0.664	0.670	0.604	0.696	0.400	
3	speaker_07_center_83.mov	0.666	0.657	0.568	0.685	0.378	
5	speaker_11_center_83.mov	0.713	0.595	0.572	0.717	0.378	
4	speaker_10_center_83.mov	0.694	0.596	0.571	0.662	0.349	

Person ID	Candidate score
10	72.930
1	70.172
7	69.985
8	69.649
2	66.261
9	65.774
6	65.371
3	63.941
5	63.477
4	61.461

To rank candidates by skills, two correlation coefficients must be set for each personality trait and skill, as well as a threshold for the polarity of the traits. These coefficients should show how a person's trait score changes when it is above or below a given trait polarity threshold.

As an example, the use of correlation coefficients between five traits and four professional skills presented in the article is suggested:

1) Wehner C., de Grip A., Pfeifer H. Do recruiters select workers with different personality traits for different tasks? A discrete choice experiment // Labour Economics. - 2022. - vol. 78. - pp. 102186.

There are 4 professional skills presented:

1) Analytical. The ability to effectively solve new problems that require in-depth analysis.
2) Interactive. The ability to persuade and compromise with clients and colleagues.
3) Routine. The ability to perform routine tasks effectively with accuracy and attention to detail.
4) Non-Routine. The ability to respond to and solve problems that have no set order, demonstrating adaptability and creative problem solving skills.

The users can set their own correlation coefficients and rank candidates by other professional skills.

Ranking candidates by professional skills

[13]:
```
# Loading a dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478678231?token=0qiZwliLtHWWYMv&
↪filename=professional_skills.csv'
df_professional_skills = pd.read_csv(url)

df_professional_skills.index.name = 'ID'
df_professional_skills.index += 1
df_professional_skills.index = df_professional_skills.index.map(str)

df_professional_skills
```

[13]:

ID	Trait	Score_level	Analytical	Interactive	Routine	Non-Routine
1	Openness	high	0.082	0.348	0.571	0.510
2	Openness	low	0.196	0.152	0.148	0.218
3	Conscientiousness	high	0.994	1.333	1.507	1.258
4	Conscientiousness	low	0.241	0.188	0.191	0.267
5	Extraversion	high	0.169	-0.060	0.258	0.017
6	Extraversion	low	0.181	0.135	0.130	0.194
7	Agreeableness	high	1.239	0.964	1.400	1.191
8	Agreeableness	low	0.226	0.180	0.189	0.259
9	Non-Neuroticism	high	0.636	0.777	0.876	0.729
10	Non-Neuroticism	low	0.207	0.159	0.166	0.238

[14]:
```
_b5._priority_skill_calculation(
    correlation_coefficients = df_professional_skills,
    threshold = 0.5,
    out = True
)

_b5._save_logs(df = _b5.df_files_priority_skill_, name = 'skill_candidate_ranking_mupta_
↪ru', out = True)

# Optional
df = _b5.df_files_priority_skill_.rename(columns = {'Openness':'OPE', 'Conscientiousness
↪':'CON', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
```

(continues on next page)

(continued from previous page)

```
df
```

[14]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU
10	speaker_27_center_83.mov	0.753	0.708	0.655	0.816	0.505
8	speaker_23_center_83.mov	0.693	0.683	0.617	0.795	0.447
7	speaker_19_center_83.mov	0.761	0.653	0.651	0.789	0.460
1	speaker_01_center_83.mov	0.758	0.693	0.650	0.745	0.489
2	speaker_06_center_83.mov	0.682	0.654	0.607	0.731	0.418
6	speaker_15_center_83.mov	0.664	0.670	0.604	0.696	0.400
9	speaker_24_center_83.mov	0.706	0.658	0.611	0.697	0.412
3	speaker_07_center_83.mov	0.666	0.657	0.568	0.685	0.378
5	speaker_11_center_83.mov	0.713	0.595	0.572	0.717	0.378
4	speaker_10_center_83.mov	0.694	0.596	0.571	0.662	0.349

Person ID	Analytical	Interactive	Routine	Non-Routine
10	0.442	0.469	0.650	0.525
8	0.384	0.391	0.554	0.455
7	0.379	0.386	0.553	0.454
1	0.377	0.389	0.554	0.455
2	0.360	0.369	0.525	0.430
6	0.354	0.365	0.517	0.423
9	0.353	0.365	0.520	0.425
3	0.346	0.359	0.508	0.416
5	0.343	0.352	0.503	0.413
4	0.328	0.339	0.485	0.397

MuPTA (en)

[15]:

```
import os
import pandas as pd

# Module import
from oceanai.modules.lab.build import Run

# Creating an instance of a class
_b5 = Run(lang = 'en')

corpus = 'fi'
lang = 'en'

# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

# Building audio models
res_load_model_hc = _b5.load_audio_model_hc()
res_load_model_nn = _b5.load_audio_model_nn()

# Loading audio model weights
```

(continues on next page)

(continued from previous page)

```
url = _b5.weights_for_big5_['audio'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_audio_model_weights_hc(url = url)

url = _b5.weights_for_big5_['audio'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_audio_model_weights_nn(url = url)

# Building video models
res_load_model_hc = _b5.load_video_model_hc(lang=lang)
res_load_model_deep_fe = _b5.load_video_model_deep_fe()
res_load_model_nn = _b5.load_video_model_nn()

# Loading video model weights
url = _b5.weights_for_big5_['video'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_video_model_weights_hc(url = url)

url = _b5.weights_for_big5_['video'][corpus]['fe']['sberdisk']
res_load_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(url = url)

url = _b5.weights_for_big5_['video'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_video_model_weights_nn(url = url)

# Loading a dictionary with hand-crafted features (text modality)
res_load_text_features = _b5.load_text_features()

# Building text models
res_setup_translation_model = _b5.setup_translation_model()
res_setup_translation_model = _b5.setup_bert_encoder()
res_load_text_model_hc_fi = _b5.load_text_model_hc(corpus=corpus)
res_load_text_model_nn_fi = _b5.load_text_model_nn(corpus=corpus)

# Loading text model weights
url = _b5.weights_for_big5_['text'][corpus]['hc']['sberdisk']
res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(url = url)

url = _b5.weights_for_big5_['text'][corpus]['nn']['sberdisk']
res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(url = url)

# Building model for multimodal information fusion
res_load_avt_model_b5 = _b5.load_avt_model_b5()

# Building model for multimodal information fusion
url = _b5.weights_for_big5_['avt'][corpus]['b5']['sberdisk']
res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(url = url)

PATH_TO_DIR = './video_MuPTA/'
PATH_SAVE_VIDEO = './video_MuPTA/test/'

_b5.path_to_save_ = PATH_SAVE_VIDEO

# Loading 10 test files from the MuPTA corpus
# URL: https://hci.nw.ru/en/pages/mupta-corpus
domain = 'https://download.sberdisk.ru/download/file/'
```

(continues on next page)

(continued from previous page)

```
tets_name_files = [
    '477995979?token=2cvyk7CS0mHx2MJ&filename=speaker_06_center_83.mov',
    '477995980?token=jGPtBPS69uzFU6Y&filename=speaker_01_center_83.mov',
    '477995967?token=zCaRbNB6ht5wMPq&filename=speaker_11_center_83.mov',
    '477995966?token=B1rbinDYRQKrI3T&filename=speaker_15_center_83.mov',
    '477995978?token=dEpVDtZg1EQiEQ9&filename=speaker_07_center_83.mov',
    '477995961?token=o1hVjw8G45q9L9Z&filename=speaker_19_center_83.mov',
    '477995964?token=5K220Aqf673VHPq&filename=speaker_23_center_83.mov',
    '477995965?token=v1LVD2KT1cU7Lpb&filename=speaker_24_center_83.mov',
    '477995962?token=tmaSGyyWLA6XCy9&filename=speaker_27_center_83.mov',
    '477995963?token=bTpo96qNDPcwGqb&filename=speaker_10_center_83.mov',
]

for curr_files in tets_name_files:
    _b5.download_file_from_url(url = domain + curr_files, out = True)

# Getting scores
_b5.path_to_dataset_ = PATH_TO_DIR # Dataset directory
_b5.ext_ = ['.mov'] # Search file extensions

# Full path to the file with ground truth scores for accuracy calculation
url_accuracy = _b5.true_traits_['mupta']['sberdisk']

_b5.get_avt_predictions(url_accuracy = url_accuracy, lang = lang)
```

[2023-12-16 19:00:49] Feature extraction (hand-crafted and deep) from text ...

[2023-12-16 19:00:52] Getting scores and accuracy calculation (multimodal fusion) ...

10 from 10 (100.0%) ... GitHub:nbsphinx-math:*OCEANAI*_guide:nbsphinx-math:*notebooks*_MuPTA:nbsphinx-math:*test*_27_center_83.mov
...

Person ID	Path	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
1	speaker_01_center_83.mov	0.564985	0.539052	0.440615	0.59251	0.488763
2	speaker_06_center_83.mov	0.650774	0.663849	0.607308	0.643847	0.620627
3	speaker_07_center_83.mov	0.435976	0.486683	0.313828	0.415446	0.396618
4	speaker_10_center_83.mov	0.498542	0.511243	0.412592	0.468947	0.44399
5	speaker_11_center_83.mov	0.394776	0.341608	0.327082	0.427304	0.354936
6	speaker_15_center_83.mov	0.566107	0.543811	0.492766	0.587411	0.499433
7	speaker_19_center_83.mov	0.506271	0.438215	0.430894		
8	speaker_23_center_83.mov	0.486463	0.521755	0.309894		
9	speaker_24_center_83.mov	0.417404	0.473339	0.320714		
10	speaker_27_center_83.mov	0.526112	0.661107	0.443167		

(continues on next page)

(continued from previous page)

```
7              0.456177         0.44075
8              0.432291        0.433601
9              0.445086        0.414649
10             0.558965        0.554224
```

[2023-12-16 19:00:52] Trait-wise accuracy ...

```
          Openness  Conscientiousness  Extraversion  Agreeableness  \
Metrics
MAE         0.1727             0.1672        0.1661         0.2579
Accuracy    0.8273             0.8328        0.8339         0.7421

          Non-Neuroticism    Mean
Metrics
MAE                 0.107  0.1742
Accuracy            0.893  0.8258
```

[2023-12-16 19:00:52] Mean absolute errors: 0.1742, average accuracy: 0.8258 ...

Log files saved successfully ...

— Runtime: 372.823 sec. —

[15]:
```
True
```

Thus, the OCEAN-AI library provides tools to analyze the personality traits of candidates and their suitability for the position, which can significantly improve the recruitment process and help to make more objective and systematic decisions when ranking candidates.

The weight coefficients for 5 professions based on scientific articles are proposed:

1) Sajjad H. et al. Personality and Career Choices // African Journal of Business Management. - 2012. – Vol. 6 (6) – pp. 2255-2260.
2) Alkhelil A. H. The Relationship between Personality Traits and Career Choice: A Case Study of Secondary School Students // International Journal of Academic Research in Progressive Education and Development. – 2016. – Vol. 5(2). – pp. 2226-6348.
3) De Jong N. et al. Personality Traits and Career Role Enactment: Career Role Preferences as a Mediator // Frontiers in Psychology. – 2019. – Vol. 10. – pp. 1720.

The user can set their own weights; the sum of the weights must be equal to 100.

[16]:
```
# Loading a dataframe with weights
url = 'https://download.sberdisk.ru/download/file/478675798?token=fF5fNZVpthQlEV0&
↪filename=traits_priority_for_professions.csv'
traits_priority_for_professions = pd.read_csv(url)

traits_priority_for_professions.index.name = 'ID'
traits_priority_for_professions.index += 1
traits_priority_for_professions.index = traits_priority_for_professions.index.map(str)

traits_priority_for_professions
```

[16]:
```
                              Profession  Openness  Conscientiousness  \
ID
1                     Managers/executives        15                 35
2                        Entrepreneurship        30                 30
3    Social/Non profit making professions         5                  5
```

(continues on next page)

(continued from previous page)

```
4              Public sector professions        15                 50
5   Scientists/researchers, and engineers        50                 15

    Extraversion  Agreeableness  Non-Neuroticism
ID
1             15             30                5
2              5              5               30
3             35             35               20
4             15             15                5
5              5             15               15
```

Ranking of candidates for the position of engineer

[17]:
```
weights = traits_priority_for_professions.iloc[4].values[1:]
weights = list(map(int, weights))

_b5._candidate_ranking(
    weigths_openness = weights[0],
    weigths_conscientiousness = weights[1],
    weigths_extraversion = weights[2],
    weigths_agreeableness = weights[3],
    weigths_non_neuroticism = weights[4],
    out = False
)

_b5._save_logs(df = _b5.df_files_ranking_, name = 'engineer_candidate_ranking_mupta_en',␣
↪out = True)

# Optional
df = _b5.df_files_ranking_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON',
↪ 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[17]:
```
                              Path    OPE    CON    EXT    AGR   NNEU  \
Person ID
2          speaker_06_center_83.mov  0.651  0.664  0.607  0.644  0.621
6          speaker_15_center_83.mov  0.566  0.544  0.493  0.587  0.499
10         speaker_27_center_83.mov  0.526  0.661  0.443  0.559  0.554
1          speaker_01_center_83.mov  0.565  0.539  0.441  0.593  0.489
4          speaker_10_center_83.mov  0.499  0.511  0.413  0.469  0.444
7          speaker_19_center_83.mov  0.506  0.438  0.431  0.456  0.441
8          speaker_23_center_83.mov  0.486  0.522  0.310  0.432  0.434
3          speaker_07_center_83.mov  0.436  0.487  0.314  0.415  0.397
9          speaker_24_center_83.mov  0.417  0.473  0.321  0.445  0.415
5          speaker_11_center_83.mov  0.395  0.342  0.327  0.427  0.355

           Candidate score
Person ID
2                   64.500
```

(continues on next page)

(continued from previous page)

```
6                    55.229
10                   55.136
1                    54.757
4                    48.353
7                    47.495
8                    46.687
3                    42.849
9                    42.470
5                    38.232
```

Ranking of candidates for the position of manager

[18]:
```
weights = traits_priority_for_professions.iloc[0].values[1:]
weights = list(map(int, weights))

_b5._candidate_ranking(
    weigths_openness = weights[0],
    weigths_conscientiousness = weights[1],
    weigths_extraversion = weights[2],
    weigths_agreeableness = weights[3],
    weigths_non_neuroticism = weights[4],
    out = False
)

_b5._save_logs(df = _b5.df_files_ranking_, name = 'executive_candidate_ranking_mupta_en',
↪ out = True)

# Optional
df = _b5.df_files_ranking_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON',
↪ 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[18]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Candidate score
2	speaker_06_center_83.mov	0.651	0.664	0.607	0.644	0.621	64.524
10	speaker_27_center_83.mov	0.526	0.661	0.443	0.559	0.554	57.218
6	speaker_15_center_83.mov	0.566	0.544	0.493	0.587	0.499	55.229
1	speaker_01_center_83.mov	0.565	0.539	0.441	0.593	0.489	54.757
4	speaker_10_center_83.mov	0.499	0.511	0.413	0.469	0.444	48.353
8	speaker_23_center_83.mov	0.486	0.522	0.310	0.432	0.434	46.687
7	speaker_19_center_83.mov	0.506	0.438	0.431	0.456	0.441	47.495
9	speaker_24_center_83.mov	0.417	0.473	0.321	0.445	0.415	42.470
3	speaker_07_center_83.mov	0.436	0.487	0.314	0.415	0.397	42.849
5	speaker_11_center_83.mov	0.395	0.342	0.327	0.427	0.355	38.232

(continues on next page)

(continued from previous page)

```
6                    55.036
1                    54.170
4                    47.849
8                    45.344
7                    45.284
9                    43.064
3                    42.727
5                    37.378
```

To rank candidates by skills, two correlation coefficients must be set for each personality trait and skill, as well as a threshold for the polarity of the traits. These coefficients should show how a person's trait score changes when it is above or below a given trait polarity threshold.

As an example, the use of correlation coefficients between five traits and four professional skills presented in the article is suggested:

1) Wehner C., de Grip A., Pfeifer H. Do recruiters select workers with different personality traits for different tasks? A discrete choice experiment // Labour Economics. - 2022. - vol. 78. - pp. 102186.

There are 4 professional skills presented:

1) Analytical. The ability to effectively solve new problems that require in-depth analysis.
2) Interactive. The ability to persuade and compromise with clients and colleagues.
3) Routine. The ability to perform routine tasks effectively with accuracy and attention to detail.
4) Non-Routine. The ability to respond to and solve problems that have no set order, demonstrating adaptability and creative problem solving skills.

The users can set their own correlation coefficients and rank candidates by other professional skills.

Ranking candidates by professional skills

[19]:
```
# Loading a dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478678231?token=0qiZwliLtHWWYMv&
↪filename=professional_skills.csv'
df_professional_skills = pd.read_csv(url)

df_professional_skills.index.name = 'ID'
df_professional_skills.index += 1
df_professional_skills.index = df_professional_skills.index.map(str)

df_professional_skills
```

[19]:

ID	Trait	Score_level	Analytical	Interactive	Routine	\
1	Openness	high	0.082	0.348	0.571	
2	Openness	low	0.196	0.152	0.148	
3	Conscientiousness	high	0.994	1.333	1.507	
4	Conscientiousness	low	0.241	0.188	0.191	
5	Extraversion	high	0.169	-0.060	0.258	
6	Extraversion	low	0.181	0.135	0.130	
7	Agreeableness	high	1.239	0.964	1.400	
8	Agreeableness	low	0.226	0.180	0.189	

(continues on next page)

(continued from previous page)

```
9     Non-Neuroticism        high       0.636        0.777    0.876
10    Non-Neuroticism         low       0.207        0.159    0.166

    Non-Routine
ID
1         0.510
2         0.218
3         1.258
4         0.267
5         0.017
6         0.194
7         1.191
8         0.259
9         0.729
10        0.238
```

[20]:
```
_b5._priority_skill_calculation(
    correlation_coefficients = df_professional_skills,
    threshold = 0.5,
    out = True
)

_b5._save_logs(df = _b5.df_files_priority_skill_, name = 'skill_candidate_ranking_mupta_
↪en', out = True)

# Optional
df = _b5.df_files_priority_skill_.rename(columns = {'Openness':'OPE', 'Conscientiousness
↪':'CON', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[20]:
```
                              Path    OPE    CON    EXT    AGR   NNEU  \
Person ID
2          speaker_06_center_83.mov  0.651  0.664  0.607  0.644  0.621
10         speaker_27_center_83.mov  0.526  0.661  0.443  0.559  0.554
6          speaker_15_center_83.mov  0.566  0.544  0.493  0.587  0.499
1          speaker_01_center_83.mov  0.565  0.539  0.441  0.593  0.489
4          speaker_10_center_83.mov  0.499  0.511  0.413  0.469  0.444
8          speaker_23_center_83.mov  0.486  0.522  0.310  0.432  0.434
9          speaker_24_center_83.mov  0.417  0.473  0.321  0.445  0.415
3          speaker_07_center_83.mov  0.436  0.487  0.314  0.415  0.397
7          speaker_19_center_83.mov  0.506  0.438  0.431  0.456  0.441
5          speaker_11_center_83.mov  0.395  0.342  0.327  0.427  0.355

           Analytical  Interactive  Routine  Non-Routine
Person ID
2               0.402        0.436    0.595        0.479
10              0.365        0.419    0.524        0.451
6               0.301        0.327    0.422        0.377
1               0.299        0.325    0.421        0.375
4               0.176        0.194    0.212        0.212
```

(continues on next page)

(continued from previous page)

```
8          0.172       0.192    0.210       0.208
9          0.088       0.068    0.069       0.099
3          0.087       0.068    0.069       0.098
7          0.084       0.094    0.118       0.136
5          0.078       0.060    0.061       0.087
```

Solution of practical task 2

Task: Predicting consumer preferences for industrial goods

The solution of the practical task is performed in two stages. At the first stage it is necessary to use the OCEAN-AI library to obtain predictions (personality traits scores). The second step is to use the _priority_calculation method from the OCEAN-AI library to solve the presented practical task. Examples of the results of the work and implementation are presented below.

Thus, the OCEAN-AI library provides tools to analyze the personality traits of consumers, aiding in predicting their interests. This enables companies to tailor products and services more accurately to consumer preferences, enhancing uniqueness and personalization.

FI V2

[2]:
```
# Import required tools
import os
import pandas as pd

# Module import
from oceanai.modules.lab.build import Run

# Creating an instance of a class
_b5 = Run(lang = 'en')

# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

corpus = 'fi'

# Building audio models
res_load_model_hc = _b5.load_audio_model_hc()
res_load_model_nn = _b5.load_audio_model_nn()

# Loading audio model weights
url = _b5.weights_for_big5_['audio'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_audio_model_weights_hc(url = url)

url = _b5.weights_for_big5_['audio'][corpus]['nn']['sberdisk']
```

(continues on next page)

(continued from previous page)

```
res_load_model_weights_nn = _b5.load_audio_model_weights_nn(url = url)

# Loading audio model weights
res_load_model_hc = _b5.load_video_model_hc(lang='en')
res_load_model_deep_fe = _b5.load_video_model_deep_fe()
res_load_model_nn = _b5.load_video_model_nn()

# Loading video model weights
url = _b5.weights_for_big5_['video'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_video_model_weights_hc(url = url)

url = _b5.weights_for_big5_['video'][corpus]['fe']['sberdisk']
res_load_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(url = url)

url = _b5.weights_for_big5_['video'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_video_model_weights_nn(url = url)

# Loading a dictionary with hand-crafted features (text modality)
res_load_text_features = _b5.load_text_features()

# Building text models
res_setup_translation_model = _b5.setup_translation_model()
res_setup_translation_model = _b5.setup_bert_encoder()
res_load_text_model_hc_fi = _b5.load_text_model_hc(corpus=corpus)
res_load_text_model_nn_fi = _b5.load_text_model_nn(corpus=corpus)

# Loading text model weights
url = _b5.weights_for_big5_['text'][corpus]['hc']['sberdisk']
res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(url = url)

url = _b5.weights_for_big5_['text'][corpus]['nn']['sberdisk']
res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(url = url)

# Building model for multimodal information fusion
res_load_avt_model_b5 = _b5.load_avt_model_b5()

# Loading model weights for multimodal information fusion
url = _b5.weights_for_big5_['avt'][corpus]['b5']['sberdisk']
res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(url = url)

PATH_TO_DIR = './video_FI/'
PATH_SAVE_VIDEO = './video_FI/test/'

_b5.path_to_save_ = PATH_SAVE_VIDEO

# Loading 10 test files from the First Impressions V2 corpus
# URL: https://chalearnlap.cvc.uab.cat/dataset/24/description/
domain = 'https://download.sberdisk.ru/download/file/'
tets_name_files = [
    '429713680?token=FqHdMLSSh7zYSZt&filename=_plk5k7PBEg.003.mp4',
    '429713681?token=Hz9b4lQkrLfic33&filename=be0DQawtVkE.002.mp4',
    '429713683?token=EgUXS9Xs8xHm5gz&filename=2d6btbaNdfo.000.mp4',
```

(continues on next page)

(continued from previous page)

```
    '429713684?token=1U26753kmPYdIgt&filename=300gK3CnzW0.003.mp4',
    '429713685?token=LyigAWLTzDNwKJO&filename=300gK3CnzW0.001.mp4',
    '429713686?token=EpfRbCKHyuc4HPu&filename=cLaZxEf1nE4.004.mp4',
    '429713687?token=FNTkwqBr4jOS95l&filename=g24JGYuT74A.004.mp4',
    '429713688?token=qDT95nz7hfm2Nki&filename=JZNMxa3OKHY.000.mp4',
    '429713689?token=noLguEGXDpbcKhg&filename=nvlqJbHk_Lc.003.mp4',
    '429713679?token=9L7RQ0hgdJlcek6&filename=4vdJGgZpj4k.003.mp4'
]

for curr_files in tets_name_files:
    _b5.download_file_from_url(url = domain + curr_files, out = True)

# Getting scores
_b5.path_to_dataset_ = PATH_TO_DIR # Dataset directory
_b5.ext_ = ['.mp4'] # Search file extensions

# Full path to the file with ground truth scores for accuracy calculation
url_accuracy = _b5.true_traits_[corpus]['sberdisk']

_b5.get_avt_predictions(url_accuracy = url_accuracy, lang = 'en')
```

[2023-12-16 19:05:15] Feature extraction (hand-crafted and deep) from text ...

[2023-12-16 19:05:17] Getting scores and accuracy calculation (multimodal fusion) ...

10 from 10 (100.0%) ... GitHub:nbsphinx-math:*OCEANAI* _guide:nbsphinx-math:*notebooks* _FI:nbsphinx-math:*test* _plk5k7PBEg.003.mp4
...

Person ID	Path	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
1	2d6btbaNdfo.000.mp4	0.581159	0.628822	0.466609	0.622129	0.553832
2	300gK3CnzW0.001.mp4	0.463991	0.418851	0.41301	0.493329	0.423093
3	300gK3CnzW0.003.mp4	0.454281	0.415049	0.39189	0.485114	0.420741
4	4vdJGgZpj4k.003.mp4	0.588461	0.643233	0.530789	0.603038	0.593398
5	be0DQawtVkE.002.mp4	0.633433	0.533295	0.523742	0.608591	0.588456
6	cLaZxEf1nE4.004.mp4	0.636944	0.542386	0.558461	0.570975	0.558983
7	g24JGYuT74A.004.mp4	0.531518	0.376987	0.393309	0.4904	0.447881
8	JZNMxa3OKHY.000.mp4	0.610342	0.541418	0.563163	0.595013	0.569461
9	nvlqJbHk_Lc.003.mp4	0.495809	0.458526	0.414436	0.469152	0.435461
10	_plk5k7PBEg.003.mp4	0.60707	0.591893	0.520662	0.603938	0.565726

```
[2023-12-16 19:05:17] Trait-wise accuracy ...
```

Metrics	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism	Mean
MAE	0.0589	0.0612	0.0864	0.0697	0.0582	0.0669
Accuracy	0.9411	0.9388	0.9136	0.9303	0.9418	0.9331

```
[2023-12-16 19:05:17] Mean absolute errors: 0.0669, average accuracy: 0.9331 ...
Log files saved successfully ...
— Runtime: 64.147 sec. —
```

[2]:
```
True
```

To predict consumer preferences for industrial goods, it is necessary to know the correlation coefficients that determine the relationship between personality traits and preferences in goods or services.

As an example, it is proposed to use the correlation coefficients between the personality traits and the characteristics of the cars presented in the article:

1) O'Connor P. J. et al. What Drives Consumer Automobile Choice? Investigating Personality Trait Predictors of Vehicle Preference Factors // Personality and Individual Differences. – 2022. – Vol. 184. – pp. 111220.

The user can set their own correlation coefficients.

Predicting consumer preferences for industrial goods on the example of car characteristics

[3]:
```
# Loading dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478675818?token=EjfLMqOeK8cfnOu&
↪filename=auto_characteristics.csv'
df_correlation_coefficients = pd.read_csv(url)
df_correlation_coefficients = pd.DataFrame(
    df_correlation_coefficients.drop(['Style and performance', 'Safety and practicality
↪'], axis = 1)
)
df_correlation_coefficients.index.name = 'ID'
df_correlation_coefficients.index += 1
df_correlation_coefficients.index = df_correlation_coefficients.index.map(str)

df_correlation_coefficients
```

[3]:

ID	Trait	Performance	Classic car features	Luxury additions	Fashion and attention	Recreation	Technology	Family friendly
1	Openness	0.020000	-0.033333	-0.030000				
2	Conscientiousness	0.013333	-0.193333	-0.063333				
3	Extraversion	0.133333	0.060000	0.106667				
4	Agreeableness	-0.036667	-0.193333	-0.133333				
5	Non-Neuroticism	0.016667	-0.006667	-0.010000				

(continues on next page)

(continued from previous page)

```
ID
1               -0.050000    0.033333    0.013333        -0.030000
2               -0.096667   -0.096667    0.086667        -0.063333
3                0.123333    0.126667    0.120000         0.090000
4               -0.133333   -0.090000    0.046667        -0.016667
5               -0.006667   -0.033333    0.046667        -0.023333

    Safe and reliable  Practical and easy to use  Economical/low cost  \
ID
1            0.136667                   0.106667             0.093333
2            0.280000                   0.180000             0.130000
3            0.136667                   0.043333             0.073333
4            0.240000                   0.160000             0.120000
5            0.093333                   0.046667             0.046667

    Basic features
ID
1         0.006667
2         0.143333
3         0.050000
4         0.083333
5        -0.040000
```

[4]:
```
_b5._priority_calculation(
    correlation_coefficients = df_correlation_coefficients,
    col_name_ocean = 'Trait',
    threshold = 0.55,
    number_priority = 3,
    number_importance_traits = 3,
    out = False
)

_b5._save_logs(df = _b5.df_files_priority_, name = 'auto_characteristics_priorities_fi_en
↪', out = True)

# Optional
df = _b5.df_files_priority_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = ['OPE', 'CON', 'EXT', 'AGR', 'NNEU']
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[4]:
```
                          Path    OPE    CON    EXT    AGR   NNEU  \
Person ID
1          2d6btbaNdfo.000.mp4  0.581  0.629  0.467  0.622  0.554
2          300gK3CnzW0.001.mp4  0.464  0.419  0.413  0.493  0.423
3          300gK3CnzW0.003.mp4  0.454  0.415  0.392  0.485  0.421
4          4vdJGgZpj4k.003.mp4  0.588  0.643  0.531  0.603  0.593
5          be0DQawtVkE.002.mp4  0.633  0.533  0.524  0.609  0.588
6          cLaZxEf1nE4.004.mp4  0.637  0.542  0.558  0.571  0.559
7          g24JGYuT74A.004.mp4  0.532  0.377  0.393  0.490  0.448
8          JZNMxa3OKHY.000.mp4  0.610  0.541  0.563  0.595  0.569
```

(continues on next page)

(continued from previous page)

```
9          nvlqJbHk_Lc.003.mp4  0.496  0.459  0.414  0.469  0.435
10         _plk5k7PBEg.003.mp4  0.607  0.592  0.521  0.604  0.566

                         Priority 1                 Priority 2  \
Person ID
1                   Safe and reliable  Practical and easy to use
2                Classic car features      Fashion and attention
3                Classic car features      Fashion and attention
4                   Safe and reliable  Practical and easy to use
5           Practical and easy to use           Safe and reliable
6                   Safe and reliable         Economical/low cost
7                Classic car features      Fashion and attention
8                   Safe and reliable         Economical/low cost
9                Classic car features      Fashion and attention
10                  Safe and reliable  Practical and easy to use

                         Priority 3 Trait importance 1 Trait importance 2  \
Person ID
1                 Economical/low cost  Conscientiousness      Agreeableness
2                    Luxury additions      Agreeableness  Conscientiousness
3                    Luxury additions      Agreeableness  Conscientiousness
4                 Economical/low cost  Conscientiousness      Agreeableness
5                 Economical/low cost      Agreeableness           Openness
6           Practical and easy to use      Agreeableness           Openness
7                    Luxury additions      Agreeableness  Conscientiousness
8           Practical and easy to use      Agreeableness           Openness
9                    Luxury additions      Agreeableness  Conscientiousness
10                Economical/low cost  Conscientiousness      Agreeableness

           Trait importance 3
Person ID
1                    Openness
2                    Openness
3                    Openness
4                    Openness
5              Non-Neuroticism
6                 Extraversion
7                    Openness
8                 Extraversion
9                    Openness
10                   Openness
```

Predicting consumer preferences for industrial goods on the example of mobile device application categories

As an example, it is proposed to use the correlation coefficients between the personality traits and the mobile device application categories presented in the article:

1) Peltonen E., Sharmila P., Asare K. O., Visuri A., Lagerspetz E., Ferreira D. (2020). When phones get personal: Predicting Big Five personality traits from application usage // Pervasive and Mobile Computing. – 2020. – Vol. 69. – 101269.

[5]:
```
# Loading a dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478676690?token=7KcAxPqMpWiYQnx&
↪filename=divice_characteristics.csv'
df_divice_characteristics = pd.read_csv(url)

df_divice_characteristics.index.name = 'ID'
df_divice_characteristics.index += 1
df_divice_characteristics.index = df_divice_characteristics.index.map(str)

df_divice_characteristics
```

[5]:

ID	Trait	Communication	Game Action	Game Board	Game Casino	\
1	Openness	0.118	0.056	0.079	0.342	
2	Conscientiousness	0.119	0.043	0.107	0.448	
3	Extraversion	0.246	0.182	0.211	0.311	
4	Agreeableness	0.218	0.104	0.164	0.284	
5	Non-Neuroticism	0.046	0.047	0.125	0.515	

ID	Game Educational	Game Simulation	Game Trivia	Entertainment	Finance	\
1	0.027	0.104	0.026	0.000	0.006	
2	0.039	0.012	0.119	0.000	0.005	
3	0.102	0.165	0.223	0.001	0.003	
4	0.165	0.122	0.162	0.000	0.003	
5	0.272	0.179	0.214	0.002	0.030	

ID	Health and Fitness	Media and Video	Music and Audio	News and Magazines	\
1	0.002	0.000	0.000	0.001	
2	0.001	0.000	0.002	0.002	
3	0.000	0.001	0.001	0.001	
4	0.001	0.000	0.002	0.002	
5	0.001	0.000	0.005	0.003	

ID	Personalisation	Travel and Local	Weather
1	0.004	0.002	0.004
2	0.001	0.001	0.003
3	0.004	0.009	0.003
4	0.001	0.004	0.003
5	0.008	0.004	0.007

[6]:
```
_b5._priority_calculation(
    correlation_coefficients = df_divice_characteristics,
```

(continues on next page)

(continued from previous page)

```
    col_name_ocean = 'Trait',
    threshold = 0.55,
    number_priority = 3,
    number_importance_traits = 3,
    out = True
)

_b5._save_logs(df = _b5.df_files_priority_, name = 'divice_characteristics_priorities_fi_
↪en', out = True)

# Optional
df = _b5.df_files_priority_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = ['OPE', 'CON', 'EXT', 'AGR', 'NNEU']
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[6]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Priority 1	Priority 2	Priority 3	Trait importance 1	Trait importance 2	Trait importance 3
1	2d6btbaNdfo.000.mp4	0.581	0.629	0.467	0.622	0.554	Game Casino	Game Educational	Game Trivia	Non-Neuroticism	Conscientiousness	Agreeableness
2	300gK3CnzW0.001.mp4	0.464	0.419	0.413	0.493	0.423	Media and Video	Entertainment	Health and Fitness	Conscientiousness	Agreeableness	Extraversion
3	300gK3CnzW0.003.mp4	0.454	0.415	0.392	0.485	0.421	Media and Video	Entertainment	Health and Fitness	Conscientiousness	Agreeableness	Extraversion
4	4vdJGgZpj4k.003.mp4	0.588	0.643	0.531	0.603	0.593	Game Casino	Game Educational	Game Trivia	Non-Neuroticism	Conscientiousness	Agreeableness
5	be0DQawtVkE.002.mp4	0.633	0.533	0.524	0.609	0.588	Game Casino	Game Educational	Game Simulation	Non-Neuroticism	Agreeableness	Openness
6	cLaZxEf1nE4.004.mp4	0.637	0.542	0.558	0.571	0.559	Game Casino	Game Simulation	Game Educational	Non-Neuroticism	Agreeableness	Extraversion
7	g24JGYuT74A.004.mp4	0.532	0.377	0.393	0.490	0.448	Media and Video	Entertainment	Health and Fitness	Conscientiousness	Agreeableness	Extraversion
8	JZNMxa3OKHY.000.mp4	0.610	0.541	0.563	0.595	0.569	Game Casino	Game Simulation	Game Educational			
9	nvlqJbHk_Lc.003.mp4	0.496	0.459	0.414	0.469	0.435	Media and Video	Entertainment	Health and Fitness			
10	_plk5k7PBEg.003.mp4	0.607	0.592	0.521	0.604	0.566	Game Casino	Game Educational	Game Trivia			

(continues on next page)

(continued from previous page)

```
8            Non-Neuroticism      Agreeableness       Extraversion
9          Conscientiousness      Agreeableness       Extraversion
10           Non-Neuroticism      Agreeableness  Conscientiousness
```

MuPTA (ru)

[7]:
```
import os
import pandas as pd

# Module import
from oceanai.modules.lab.build import Run

# Creating an instance of a class
_b5 = Run(lang = 'en')

corpus = 'mupta'
lang = 'ru'

# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

# Building audio models
res_load_model_hc = _b5.load_audio_model_hc()
res_load_model_nn = _b5.load_audio_model_nn()

# Loading audio model weights
url = _b5.weights_for_big5_['audio'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_audio_model_weights_hc(url = url)

url = _b5.weights_for_big5_['audio'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_audio_model_weights_nn(url = url)

# Building video models
res_load_model_hc = _b5.load_video_model_hc(lang=lang)
res_load_model_deep_fe = _b5.load_video_model_deep_fe()
res_load_model_nn = _b5.load_video_model_nn()

# Loading video model weights
url = _b5.weights_for_big5_['video'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_video_model_weights_hc(url = url)

url = _b5.weights_for_big5_['video'][corpus]['fe']['sberdisk']
res_load_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(url = url)

url = _b5.weights_for_big5_['video'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_video_model_weights_nn(url = url)

# Loading a dictionary with hand-crafted features (text modality)
res_load_text_features = _b5.load_text_features()
```

(continues on next page)

(continued from previous page)

```
# Building text models
res_setup_translation_model = _b5.setup_translation_model()
res_setup_translation_model = _b5.setup_bert_encoder()
res_load_text_model_hc_fi = _b5.load_text_model_hc(corpus=corpus)
res_load_text_model_nn_fi = _b5.load_text_model_nn(corpus=corpus)

# Loading text model weights
url = _b5.weights_for_big5_['text'][corpus]['hc']['sberdisk']
res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(url = url)

url = _b5.weights_for_big5_['text'][corpus]['nn']['sberdisk']
res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(url = url)

# Building model for multimodal information fusion
res_load_avt_model_b5 = _b5.load_avt_model_b5()

# Loading model weights for multimodal information fusion
url = _b5.weights_for_big5_['avt'][corpus]['b5']['sberdisk']
res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(url = url)

PATH_TO_DIR = './video_MuPTA/'
PATH_SAVE_VIDEO = './video_MuPTA/test/'

_b5.path_to_save_ = PATH_SAVE_VIDEO

# Loading 10 test files from the MuPTA corpus
# URL: https://hci.nw.ru/en/pages/mupta-corpus
domain = 'https://download.sberdisk.ru/download/file/'
tets_name_files = [
    '477995979?token=2cvyk7CS0mHx2MJ&filename=speaker_06_center_83.mov',
    '477995980?token=jGPtBPS69uzFU6Y&filename=speaker_01_center_83.mov',
    '477995967?token=zCaRbNB6ht5wMPq&filename=speaker_11_center_83.mov',
    '477995966?token=B1rbinDYRQKrI3T&filename=speaker_15_center_83.mov',
    '477995978?token=dEpVDtZg1EQiEQ9&filename=speaker_07_center_83.mov',
    '477995961?token=o1hVjw8G45q9L9Z&filename=speaker_19_center_83.mov',
    '477995964?token=5K220Aqf673VHPq&filename=speaker_23_center_83.mov',
    '477995965?token=v1LVD2KT1cU7Lpb&filename=speaker_24_center_83.mov',
    '477995962?token=tmaSGyyWLA6XCy9&filename=speaker_27_center_83.mov',
    '477995963?token=bTpo96qNDPcwGqb&filename=speaker_10_center_83.mov',
]

for curr_files in tets_name_files:
    _b5.download_file_from_url(url = domain + curr_files, out = True)

# Getting scores
_b5.path_to_dataset_ = PATH_TO_DIR # Dataset directory
_b5.ext_ = ['.mov'] # Search file extensions

# Full path to the file with ground truth scores for accuracy calculation
url_accuracy = _b5.true_traits_['mupta']['sberdisk']
```

(continues on next page)

(continued from previous page)

```
_b5.get_avt_predictions(url_accuracy = url_accuracy, lang = lang)
```

[2023-12-16 19:13:25] Feature extraction (hand-crafted and deep) from text ...

[2023-12-16 19:13:30] Getting scores and accuracy calculation (multimodal fusion) ...

10 from 10 (100.0%) ... GitHub:nbsphinx-math:*OCEANAI*_guide:nbsphinx-math:*notebooks*_MuPTA:nbsphinx-math:*test*_27_center_83.mov
...

Person ID	Path	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
1	speaker_01_center_83.mov	0.758137	0.693356	0.650108	0.744589	0.488671
2	speaker_06_center_83.mov	0.681602	0.654339	0.607156	0.731282	0.417908
3	speaker_07_center_83.mov	0.666104	0.656836	0.567863	0.685067	0.378102
4	speaker_10_center_83.mov	0.694171	0.596195	0.571414	0.66223	0.348639
5	speaker_11_center_83.mov	0.712885	0.594764	0.571709	0.716696	0.37802
6	speaker_15_center_83.mov	0.664158	0.670411	0.60421	0.696056	0.399842
7	speaker_19_center_83.mov	0.761213	0.652635	0.651028	0.788677	0.459676
8	speaker_23_center_83.mov	0.692788	0.68324	0.616737	0.795205	0.447242
9	speaker_24_center_83.mov	0.705923	0.658382	0.610645	0.697415	0.411988
10	speaker_27_center_83.mov	0.753417	0.708372	0.654608	0.816416	0.504743

[2023-12-16 19:13:30] Trait-wise accuracy ...

Metrics	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism	Mean
MAE	0.0673	0.0789	0.1325	0.102	0.1002	0.0962
Accuracy	0.9327	0.9211	0.8675	0.898	0.8998	0.9038

[2023-12-16 19:13:30] Mean absolute errors: 0.0962, average accuracy: 0.9038 ...

Log files saved successfully ...

— Runtime: 416.453 sec. —

[7]: True

To predict consumer preferences for industrial goods, it is necessary to know the correlation coefficients that determine the relationship between personality traits and preferences in goods or services.

As an example, it is proposed to use the correlation coefficients between the personality traits and the characteristics of the cars presented in the article:

1) O'Connor P. J. et al. What Drives Consumer Automobile Choice? Investigating Personality Trait Predictors of Vehicle Preference Factors // Personality and Individual Differences. – 2022. – Vol. 184. – pp. 111220.

The user can set their own correlation coefficients.

Predicting consumer preferences for industrial goods on the example of car characteristics

[8]:
```
# Loading dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478675818?token=EjfLMqOeK8cfnOu&
↪filename=auto_characteristics.csv'
df_correlation_coefficients = pd.read_csv(url)
df_correlation_coefficients = pd.DataFrame(
    df_correlation_coefficients.drop(['Style and performance', 'Safety and practicality
↪'], axis = 1)
)
df_correlation_coefficients.index.name = 'ID'
df_correlation_coefficients.index += 1
df_correlation_coefficients.index = df_correlation_coefficients.index.map(str)

df_correlation_coefficients
```

[8]:

ID	Trait	Performance	Classic car features	Luxury additions	Fashion and attention	Recreation	Technology	Family friendly	Safe and reliable	Practical and easy to use	Economical/low cost	Basic features
1	Openness	0.020000	-0.033333	-0.030000	-0.050000	0.033333	0.013333	-0.030000	0.136667	0.106667	0.093333	0.006667
2	Conscientiousness	0.013333	-0.193333	-0.063333	-0.096667	-0.096667	0.086667	-0.063333	0.280000	0.180000	0.130000	0.143333
3	Extraversion	0.133333	0.060000	0.106667	0.123333	0.126667	0.120000	0.090000	0.136667	0.043333	0.073333	
4	Agreeableness	-0.036667	-0.193333	-0.133333	-0.133333	-0.090000	0.046667	-0.016667	0.240000	0.160000	0.120000	
5	Non-Neuroticism	0.016667	-0.006667	-0.010000	-0.006667	-0.033333	0.046667	-0.023333	0.093333	0.046667	0.046667	

(continues on next page)

(continued from previous page)

```
3         0.050000
4         0.083333
5        -0.040000
```

[9]:
```
_b5._priority_calculation(
    correlation_coefficients = df_correlation_coefficients,
    col_name_ocean = 'Trait',
    threshold = 0.55,
    number_priority = 3,
    number_importance_traits = 3,
    out = False
)

_b5._save_logs(df = _b5.df_files_priority_, name = 'auto_characteristics_priorities_
↪mupta_ru', out = True)

# Optional
df = _b5.df_files_priority_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = ['OPE', 'CON', 'EXT', 'AGR', 'NNEU']
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[9]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Priority 1	Priority 2	Priority 3	Trait importance 1	Trait importance 2	Trait importance 3
1	speaker_01_center_83.mov	0.758	0.693	0.650	0.745	0.489	Safe and reliable	Practical and easy to use	Economical/low cost	Conscientiousness	Agreeableness	Openness
2	speaker_06_center_83.mov	0.682	0.654	0.607	0.731	0.418	Safe and reliable	Practical and easy to use	Economical/low cost			
3	speaker_07_center_83.mov	0.666	0.657	0.568	0.685	0.378	Safe and reliable	Practical and easy to use	Economical/low cost			
4	speaker_10_center_83.mov	0.694	0.596	0.571	0.662	0.349	Safe and reliable	Practical and easy to use	Economical/low cost			
5	speaker_11_center_83.mov	0.713	0.595	0.572	0.717	0.378	Safe and reliable	Practical and easy to use	Economical/low cost			
6	speaker_15_center_83.mov	0.664	0.670	0.604	0.696	0.400	Safe and reliable	Practical and easy to use	Economical/low cost			
7	speaker_19_center_83.mov	0.761	0.653	0.651	0.789	0.460	Safe and reliable	Practical and easy to use	Economical/low cost			
8	speaker_23_center_83.mov	0.693	0.683	0.617	0.795	0.447	Safe and reliable	Practical and easy to use	Economical/low cost			
9	speaker_24_center_83.mov	0.706	0.658	0.611	0.697	0.412	Safe and reliable	Practical and easy to use	Economical/low cost			
10	speaker_27_center_83.mov	0.753	0.708	0.655	0.816	0.505	Safe and reliable	Practical and easy to use	Economical/low cost			

(continues on next page)

(continued from previous page)

```
2          Conscientiousness      Agreeableness           Openness
3          Conscientiousness      Agreeableness           Openness
4          Conscientiousness      Agreeableness           Openness
5              Agreeableness  Conscientiousness           Openness
6          Conscientiousness      Agreeableness           Openness
7              Agreeableness  Conscientiousness           Openness
8              Agreeableness  Conscientiousness           Openness
9          Conscientiousness      Agreeableness           Openness
10             Agreeableness  Conscientiousness           Openness
```

Predicting consumer preferences for industrial goods on the example of mobile device application categories

As an example, it is proposed to use the correlation coefficients between the personality traits and the mobile device application categories presented in the article:

1) Peltonen E., Sharmila P., Asare K. O., Visuri A., Lagerspetz E., Ferreira D. (2020). When phones get personal: Predicting Big Five personality traits from application usage // Pervasive and Mobile Computing. – 2020. – Vol. 69. – 101269.

[10]:
```
# Loading a dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478676690?token=7KcAxPqMpWiYQnx&
↪filename=divice_characteristics.csv'
df_divice_characteristics = pd.read_csv(url)

df_divice_characteristics.index.name = 'ID'
df_divice_characteristics.index += 1
df_divice_characteristics.index = df_divice_characteristics.index.map(str)

df_divice_characteristics
```

[10]:
```
                Trait  Communication  Game Action  Game Board  Game Casino  \
ID
1            Openness          0.118        0.056       0.079        0.342
2   Conscientiousness          0.119        0.043       0.107        0.448
3        Extraversion          0.246        0.182       0.211        0.311
4       Agreeableness          0.218        0.104       0.164        0.284
5     Non-Neuroticism          0.046        0.047       0.125        0.515

    Game Educational  Game Simulation  Game Trivia  Entertainment  Finance  \
ID
1              0.027            0.104        0.026          0.000    0.006
2              0.039            0.012        0.119          0.000    0.005
3              0.102            0.165        0.223          0.001    0.003
4              0.165            0.122        0.162          0.000    0.003
5              0.272            0.179        0.214          0.002    0.030

    Health and Fitness  Media and Video  Music and Audio  News and Magazines  \
ID
1                0.002            0.000            0.000               0.001
2                0.001            0.000            0.002               0.002
3                0.000            0.001            0.001               0.001
```

(continues on next page)

(continued from previous page)

```
4                0.001            0.000            0.002                0.002
5                0.001            0.000            0.005                0.003

    Personalisation  Travel and Local  Weather
ID
1             0.004             0.002    0.004
2             0.001             0.001    0.003
3             0.004             0.009    0.003
4             0.001             0.004    0.003
5             0.008             0.004    0.007
```

[11]:
```
_b5._priority_calculation(
    correlation_coefficients = df_divice_characteristics,
    col_name_ocean = 'Trait',
    threshold = 0.55,
    number_priority = 3,
    number_importance_traits = 3,
    out = True
)

_b5._save_logs(df = _b5.df_files_priority_, name = 'divice_characteristics_priorities_
↪mupta_ru', out = True)

# Optional
df = _b5.df_files_priority_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = ['OPE', 'CON', 'EXT', 'AGR', 'NNEU']
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[11]:
```
                              Path    OPE    CON    EXT    AGR   NNEU  \
Person ID
1          speaker_01_center_83.mov  0.758  0.693  0.650  0.745  0.489
2          speaker_06_center_83.mov  0.682  0.654  0.607  0.731  0.418
3          speaker_07_center_83.mov  0.666  0.657  0.568  0.685  0.378
4          speaker_10_center_83.mov  0.694  0.596  0.571  0.662  0.349
5          speaker_11_center_83.mov  0.713  0.595  0.572  0.717  0.378
6          speaker_15_center_83.mov  0.664  0.670  0.604  0.696  0.400
7          speaker_19_center_83.mov  0.761  0.653  0.651  0.789  0.460
8          speaker_23_center_83.mov  0.693  0.683  0.617  0.795  0.447
9          speaker_24_center_83.mov  0.706  0.658  0.611  0.697  0.412
10         speaker_27_center_83.mov  0.753  0.708  0.655  0.816  0.505

            Priority 1     Priority 2  Priority 3 Trait importance 1  \
Person ID
1          Game Casino  Communication  Game Board       Extraversion
2          Game Casino  Communication  Game Board      Agreeableness
3          Game Casino  Communication  Game Board      Agreeableness
4          Game Casino  Communication  Game Board      Agreeableness
5          Game Casino  Communication  Game Board      Agreeableness
6          Game Casino  Communication  Game Board       Extraversion
7          Game Casino  Communication  Game Board      Agreeableness
```

(continues on next page)

(continued from previous page)

```
8          Game Casino  Communication  Game Board      Agreeableness
9          Game Casino  Communication  Game Board       Extraversion
10         Game Casino  Communication  Game Board      Agreeableness

          Trait importance 2 Trait importance 3
Person ID
1              Agreeableness  Conscientiousness
2               Extraversion  Conscientiousness
3          Conscientiousness       Extraversion
4               Extraversion  Conscientiousness
5               Extraversion  Conscientiousness
6              Agreeableness  Conscientiousness
7               Extraversion  Conscientiousness
8               Extraversion  Conscientiousness
9              Agreeableness  Conscientiousness
10              Extraversion  Conscientiousness
```

MuPTA (en)

[12]:
```
import os
import pandas as pd

# Module import
from oceanai.modules.lab.build import Run

# Creating an instance of a class
_b5 = Run(lang = 'en')

corpus = 'fi'
lang = 'en'

# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

# Building audio models
res_load_model_hc = _b5.load_audio_model_hc()
res_load_model_nn = _b5.load_audio_model_nn()

# Loading audio model weights
url = _b5.weights_for_big5_['audio'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_audio_model_weights_hc(url = url)

url = _b5.weights_for_big5_['audio'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_audio_model_weights_nn(url = url)

# Building video models
res_load_model_hc = _b5.load_video_model_hc(lang=lang)
res_load_model_deep_fe = _b5.load_video_model_deep_fe()
res_load_model_nn = _b5.load_video_model_nn()
```

(continues on next page)

(continued from previous page)

```
# Loading video model weights
url = _b5.weights_for_big5_['video'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_video_model_weights_hc(url = url)

url = _b5.weights_for_big5_['video'][corpus]['fe']['sberdisk']
res_load_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(url = url)

url = _b5.weights_for_big5_['video'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_video_model_weights_nn(url = url)

# Loading a dictionary with hand-crafted features (text modality)
res_load_text_features = _b5.load_text_features()

# Building text models
res_setup_translation_model = _b5.setup_translation_model()
res_setup_translation_model = _b5.setup_bert_encoder()
res_load_text_model_hc_fi = _b5.load_text_model_hc(corpus=corpus)
res_load_text_model_nn_fi = _b5.load_text_model_nn(corpus=corpus)

# Loading text model weights
url = _b5.weights_for_big5_['text'][corpus]['hc']['sberdisk']
res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(url = url)

url = _b5.weights_for_big5_['text'][corpus]['nn']['sberdisk']
res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(url = url)

# Building model for multimodal information fusion
res_load_avt_model_b5 = _b5.load_avt_model_b5()

# Building model for multimodal information fusion
url = _b5.weights_for_big5_['avt'][corpus]['b5']['sberdisk']
res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(url = url)

PATH_TO_DIR = './video_MuPTA/'
PATH_SAVE_VIDEO = './video_MuPTA/test/'

_b5.path_to_save_ = PATH_SAVE_VIDEO

# Loading 10 test files from the MuPTA corpus
# URL: https://hci.nw.ru/en/pages/mupta-corpus
domain = 'https://download.sberdisk.ru/download/file/'
tets_name_files = [
    '477995979?token=2cvyk7CS0mHx2MJ&filename=speaker_06_center_83.mov',
    '477995980?token=jGPtBPS69uzFU6Y&filename=speaker_01_center_83.mov',
    '477995967?token=zCaRbNB6ht5wMPq&filename=speaker_11_center_83.mov',
    '477995966?token=B1rbinDYRQKrI3T&filename=speaker_15_center_83.mov',
    '477995978?token=dEpVDtZg1EQiEQ9&filename=speaker_07_center_83.mov',
    '477995961?token=o1hVjw8G45q9L9Z&filename=speaker_19_center_83.mov',
    '477995964?token=5K220Aqf673VHPq&filename=speaker_23_center_83.mov',
    '477995965?token=v1LVD2KT1cU7Lpb&filename=speaker_24_center_83.mov',
    '477995962?token=tmaSGyyWLA6XCy9&filename=speaker_27_center_83.mov',
```

(continues on next page)

(continued from previous page)

```
    '477995963?token=bTpo96qNDPcwGqb&filename=speaker_10_center_83.mov',
]

for curr_files in tets_name_files:
    _b5.download_file_from_url(url = domain + curr_files, out = True)

# Getting scores
_b5.path_to_dataset_ = PATH_TO_DIR # Dataset directory
_b5.ext_ = ['.mov'] # Search file extensions

# Full path to the file with ground truth scores for accuracy calculation
url_accuracy = _b5.true_traits_['mupta']['sberdisk']

_b5.get_avt_predictions(url_accuracy = url_accuracy, lang = lang)
```

[2023-12-16 19:20:55] Feature extraction (hand-crafted and deep) from text ...

[2023-12-16 19:20:57] Getting scores and accuracy calculation (multimodal fusion) ...

10 from 10 (100.0%) ... GitHub:nbsphinx-math:*OCEANAI*_guide:nbsphinx-math:*notebooks*_MuPTA:nbsphinx-math:*test*_27_center_83.mov
...

Person ID	Path	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
1	speaker_01_center_83.mov	0.564985	0.539052	0.440615	0.59251	0.488763
2	speaker_06_center_83.mov	0.650774	0.663849	0.607308	0.643847	0.620627
3	speaker_07_center_83.mov	0.435976	0.486683	0.313828	0.415446	0.396618
4	speaker_10_center_83.mov	0.498542	0.511243	0.412592	0.468947	0.44399
5	speaker_11_center_83.mov	0.394776	0.341608	0.327082	0.427304	0.354936
6	speaker_15_center_83.mov	0.566107	0.543811	0.492766	0.587411	0.499433
7	speaker_19_center_83.mov	0.506271	0.438215	0.430894	0.456177	0.44075
8	speaker_23_center_83.mov	0.486463	0.521755	0.309894	0.432291	0.433601
9	speaker_24_center_83.mov	0.417404	0.473339	0.320714	0.445086	0.414649
10	speaker_27_center_83.mov	0.526112	0.661107	0.443167	0.558965	0.554224

[2023-12-16 19:20:57] Trait-wise accuracy ...

Metrics	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
MAE	0.1727	0.1672	0.1661	0.2579	0.1727
Accuracy	0.8273	0.8328	0.8339	0.7421	0.8273

(continues on next page)

(continued from previous page)

```
          Non-Neuroticism    Mean
Metrics
MAE                 0.107  0.1742
Accuracy            0.893  0.8258
```

[2023-12-16 19:20:57] Mean absolute errors: 0.1742, average accuracy: 0.8258 ...

Log files saved successfully ...

— Runtime: 379.936 sec. —

[12]:
```
True
```

To predict consumer preferences for industrial goods, it is necessary to know the correlation coefficients that determine the relationship between personality traits and preferences in goods or services.

As an example, it is proposed to use the correlation coefficients between the personality traits and the characteristics of the cars presented in the article:

1) O'Connor P. J. et al. What Drives Consumer Automobile Choice? Investigating Personality Trait Predictors of Vehicle Preference Factors // Personality and Individual Differences. – 2022. – Vol. 184. – pp. 111220.

The user can set their own correlation coefficients.

Predicting consumer preferences for industrial goods on the example of car characteristics

[13]:
```
# Loading dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478675818?token=EjfLMqOeK8cfnOu&
↪filename=auto_characteristics.csv'
df_correlation_coefficients = pd.read_csv(url)
df_correlation_coefficients = pd.DataFrame(
    df_correlation_coefficients.drop(['Style and performance', 'Safety and practicality
↪'], axis = 1)
)
df_correlation_coefficients.index.name = 'ID'
df_correlation_coefficients.index += 1
df_correlation_coefficients.index = df_correlation_coefficients.index.map(str)

df_correlation_coefficients
```

[13]:
```
                Trait  Performance  Classic car features  Luxury additions  \
ID
1            Openness     0.020000             -0.033333         -0.030000
2   Conscientiousness     0.013333             -0.193333         -0.063333
3        Extraversion     0.133333              0.060000          0.106667
4       Agreeableness    -0.036667             -0.193333         -0.133333
5     Non-Neuroticism     0.016667             -0.006667         -0.010000

    Fashion and attention  Recreation  Technology  Family friendly  \
ID
1               -0.050000    0.033333    0.013333        -0.030000
2               -0.096667   -0.096667    0.086667        -0.063333
3                0.123333    0.126667    0.120000         0.090000
```

(continues on next page)

(continued from previous page)

```
4               -0.133333   -0.090000    0.046667        -0.016667
5               -0.006667   -0.033333    0.046667        -0.023333

    Safe and reliable  Practical and easy to use  Economical/low cost  \
ID
1            0.136667                   0.106667             0.093333
2            0.280000                   0.180000             0.130000
3            0.136667                   0.043333             0.073333
4            0.240000                   0.160000             0.120000
5            0.093333                   0.046667             0.046667

    Basic features
ID
1         0.006667
2         0.143333
3         0.050000
4         0.083333
5        -0.040000
```

[14]:
```
_b5._priority_calculation(
    correlation_coefficients = df_correlation_coefficients,
    col_name_ocean = 'Trait',
    threshold = 0.55,
    number_priority = 3,
    number_importance_traits = 3,
    out = False
)

_b5._save_logs(df = _b5.df_files_priority_, name = 'auto_characteristics_priorities_
↪mupta_en', out = True)

# Optional
df = _b5.df_files_priority_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = ['OPE', 'CON', 'EXT', 'AGR', 'NNEU']
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[14]:
```
                              Path    OPE    CON    EXT    AGR   NNEU  \
Person ID
1          speaker_01_center_83.mov  0.565  0.539  0.441  0.593  0.489
2          speaker_06_center_83.mov  0.651  0.664  0.607  0.644  0.621
3          speaker_07_center_83.mov  0.436  0.487  0.314  0.415  0.397
4          speaker_10_center_83.mov  0.499  0.511  0.413  0.469  0.444
5          speaker_11_center_83.mov  0.395  0.342  0.327  0.427  0.355
6          speaker_15_center_83.mov  0.566  0.544  0.493  0.587  0.499
7          speaker_19_center_83.mov  0.506  0.438  0.431  0.456  0.441
8          speaker_23_center_83.mov  0.486  0.522  0.310  0.432  0.434
9          speaker_24_center_83.mov  0.417  0.473  0.321  0.445  0.415
10         speaker_27_center_83.mov  0.526  0.661  0.443  0.559  0.554

                          Priority 1                Priority 2  \
```

(continues on next page)

(continued from previous page)

```
Person ID
1          Practical and easy to use        Economical/low cost
2                  Safe and reliable  Practical and easy to use
3               Classic car features      Fashion and attention
4               Classic car features      Fashion and attention
5               Classic car features      Fashion and attention
6          Practical and easy to use        Economical/low cost
7               Classic car features      Fashion and attention
8               Classic car features      Fashion and attention
9               Classic car features      Fashion and attention
10                 Safe and reliable  Practical and easy to use

                    Priority 3 Trait importance 1 Trait importance 2  \
Person ID
1                Family friendly      Agreeableness           Openness
2           Economical/low cost  Conscientiousness      Agreeableness
3              Luxury additions      Agreeableness  Conscientiousness
4              Luxury additions      Agreeableness  Conscientiousness
5              Luxury additions      Agreeableness  Conscientiousness
6               Family friendly      Agreeableness           Openness
7              Luxury additions      Agreeableness  Conscientiousness
8              Luxury additions      Agreeableness  Conscientiousness
9              Luxury additions      Agreeableness  Conscientiousness
10          Economical/low cost  Conscientiousness      Agreeableness

           Trait importance 3
Person ID
1             Non-Neuroticism
2                    Openness
3                    Openness
4                    Openness
5                    Openness
6             Non-Neuroticism
7                    Openness
8                    Openness
9                    Openness
10            Non-Neuroticism
```

Predicting consumer preferences for industrial goods on the example of mobile device application categories

As an example, it is proposed to use the correlation coefficients between the personality traits and the mobile device application categories presented in the article:

1) Peltonen E., Sharmila P., Asare K. O., Visuri A., Lagerspetz E., Ferreira D. (2020). When phones get personal: Predicting Big Five personality traits from application usage // Pervasive and Mobile Computing. – 2020. – Vol. 69. – 101269.

[15]:
```
# Loading a dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478676690?token=7KcAxPqMpWiYQnx&
↪filename=divice_characteristics.csv'
df_divice_characteristics = pd.read_csv(url)
```

(continues on next page)

(continued from previous page)

```
df_divice_characteristics.index.name = 'ID'
df_divice_characteristics.index += 1
df_divice_characteristics.index = df_divice_characteristics.index.map(str)

df_divice_characteristics
```

[15]:

ID	Trait	Communication	Game Action	Game Board	Game Casino	\
1	Openness	0.118	0.056	0.079	0.342	
2	Conscientiousness	0.119	0.043	0.107	0.448	
3	Extraversion	0.246	0.182	0.211	0.311	
4	Agreeableness	0.218	0.104	0.164	0.284	
5	Non-Neuroticism	0.046	0.047	0.125	0.515	

ID	Game Educational	Game Simulation	Game Trivia	Entertainment	Finance	\
1	0.027	0.104	0.026	0.000	0.006	
2	0.039	0.012	0.119	0.000	0.005	
3	0.102	0.165	0.223	0.001	0.003	
4	0.165	0.122	0.162	0.000	0.003	
5	0.272	0.179	0.214	0.002	0.030	

ID	Health and Fitness	Media and Video	Music and Audio	News and Magazines	\
1	0.002	0.000	0.000	0.001	
2	0.001	0.000	0.002	0.002	
3	0.000	0.001	0.001	0.001	
4	0.001	0.000	0.002	0.002	
5	0.001	0.000	0.005	0.003	

ID	Personalisation	Travel and Local	Weather
1	0.004	0.002	0.004
2	0.001	0.001	0.003
3	0.004	0.009	0.003
4	0.001	0.004	0.003
5	0.008	0.004	0.007

[16]:
```
_b5._priority_calculation(
    correlation_coefficients = df_divice_characteristics,
    col_name_ocean = 'Trait',
    threshold = 0.55,
    number_priority = 3,
    number_importance_traits = 3,
    out = True
)

_b5._save_logs(df = _b5.df_files_priority_, name = 'divice_characteristics_priorities_
↪mupta_en', out = True)

# Optional
```

(continues on next page)

(continued from previous page)

```
df = _b5.df_files_priority_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = ['OPE', 'CON', 'EXT', 'AGR', 'NNEU']
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[16]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Priority 1	Priority 2	Priority 3	Trait importance 1	Trait importance 2	Trait importance 3
1	speaker_01_center_83.mov	0.565	0.539	0.441	0.593	0.489	Communication	Health and Fitness	Media and Video	Agreeableness	Openness	Non-Neuroticism
2	speaker_06_center_83.mov	0.651	0.664	0.607	0.644	0.621	Game Casino	Communication	Game Trivia	Non-Neuroticism	Extraversion	Conscientiousness
3	speaker_07_center_83.mov	0.436	0.487	0.314	0.415	0.397	Media and Video	Entertainment	Health and Fitness	Agreeableness	Conscientiousness	Extraversion
4	speaker_10_center_83.mov	0.499	0.511	0.413	0.469	0.444	Media and Video	Entertainment	Health and Fitness	Agreeableness	Conscientiousness	Extraversion
5	speaker_11_center_83.mov	0.395	0.342	0.327	0.427	0.355	Media and Video	Entertainment	Health and Fitness	Conscientiousness	Agreeableness	Extraversion
6	speaker_15_center_83.mov	0.566	0.544	0.493	0.587	0.499	Health and Fitness	Media and Video	News and Magazines	Agreeableness	Openness	Extraversion
7	speaker_19_center_83.mov	0.506	0.438	0.431	0.456	0.441	Media and Video	Entertainment	Health and Fitness	Conscientiousness	Agreeableness	Extraversion
8	speaker_23_center_83.mov	0.486	0.522	0.310	0.432	0.434	Media and Video	Entertainment	Health and Fitness	Agreeableness	Conscientiousness	Extraversion
9	speaker_24_center_83.mov	0.417	0.473	0.321	0.445	0.415	Media and Video	Entertainment	Health and Fitness	Agreeableness	Conscientiousness	Extraversion
10	speaker_27_center_83.mov	0.526	0.661	0.443	0.559	0.554	Game Casino	Game Educational	Game Trivia	Non-Neuroticism	Conscientiousness	Agreeableness

Solution of practical task 3

Task: Forming effective work teams

The solution of the practical task is performed in two stages. At the first stage it is necessary to use the OCEAN-AI library to obtain predictions (personality traits scores). The second step is to use the _colleague_ranking method from the OCEAN-AI library to solve the presented practical task using the example of finding suitable colleagues for the target colleague. Examples of the results of the work and implementation are presented below.

Thus, the OCEAN-AI library provides tools to analyze the personality traits of colleagues' personalities and can help to form effective work groups, improve communication, and reduce team conflicts.

FI V2

[2]:
```
# Import required tools
import os
import pandas as pd

# Module import
from oceanai.modules.lab.build import Run

# Creating an instance of a class
_b5 = Run(lang = 'en')

# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

corpus = 'fi'

# Building audio models
res_load_model_hc = _b5.load_audio_model_hc()
res_load_model_nn = _b5.load_audio_model_nn()

# Loading audio model weights
url = _b5.weights_for_big5_['audio'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_audio_model_weights_hc(url = url)

url = _b5.weights_for_big5_['audio'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_audio_model_weights_nn(url = url)

# Loading audio model weights
res_load_model_hc = _b5.load_video_model_hc(lang='en')
res_load_model_deep_fe = _b5.load_video_model_deep_fe()
res_load_model_nn = _b5.load_video_model_nn()

# Loading video model weights
```

(continues on next page)

(continued from previous page)

```
url = _b5.weights_for_big5_['video'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_video_model_weights_hc(url = url)

url = _b5.weights_for_big5_['video'][corpus]['fe']['sberdisk']
res_load_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(url = url)

url = _b5.weights_for_big5_['video'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_video_model_weights_nn(url = url)

# Loading a dictionary with hand-crafted features (text modality)
res_load_text_features = _b5.load_text_features()

# Building text models
res_setup_translation_model = _b5.setup_translation_model()
res_setup_translation_model = _b5.setup_bert_encoder()
res_load_text_model_hc_fi = _b5.load_text_model_hc(corpus=corpus)
res_load_text_model_nn_fi = _b5.load_text_model_nn(corpus=corpus)

# Loading text model weights
url = _b5.weights_for_big5_['text'][corpus]['hc']['sberdisk']
res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(url = url)

url = _b5.weights_for_big5_['text'][corpus]['nn']['sberdisk']
res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(url = url)

# Building model for multimodal information fusion
res_load_avt_model_b5 = _b5.load_avt_model_b5()

# Loading model weights for multimodal information fusion
url = _b5.weights_for_big5_['avt'][corpus]['b5']['sberdisk']
res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(url = url)

PATH_TO_DIR = './video_FI/'
PATH_SAVE_VIDEO = './video_FI/test/'

_b5.path_to_save_ = PATH_SAVE_VIDEO

# Loading 10 test files from the First Impressions V2 corpus
# URL: https://chalearnlap.cvc.uab.cat/dataset/24/description/
domain = 'https://download.sberdisk.ru/download/file/'
tets_name_files = [
    '429713680?token=FqHdMLSSh7zYSZt&filename=_plk5k7PBEg.003.mp4',
    '429713681?token=Hz9b4lQkrLfic33&filename=be0DQawtVkE.002.mp4',
    '429713683?token=EgUXS9Xs8xHm5gz&filename=2d6btbaNdfo.000.mp4',
    '429713684?token=1U26753kmPYdIgt&filename=300gK3CnzW0.003.mp4',
    '429713685?token=LyigAWLTzDNwKJO&filename=300gK3CnzW0.001.mp4',
    '429713686?token=EpfRbCKHyuc4HPu&filename=cLaZxEf1nE4.004.mp4',
    '429713687?token=FNTkwqBr4jOS95l&filename=g24JGYuT74A.004.mp4',
    '429713688?token=qDT95nz7hfm2Nki&filename=JZNMxa3OKHY.000.mp4',
    '429713689?token=noLguEGXDpbcKhg&filename=nvlqJbHk_Lc.003.mp4',
    '429713679?token=9L7RQ0hgdJlcek6&filename=4vdJGgZpj4k.003.mp4'
]
```

(continues on next page)

(continued from previous page)

```
for curr_files in tets_name_files:
    _b5.download_file_from_url(url = domain + curr_files, out = True)

# Getting scores
_b5.path_to_dataset_ = PATH_TO_DIR # Dataset directory
_b5.ext_ = ['.mp4'] # Search file extensions

# Full path to the file with ground truth scores for accuracy calculation
url_accuracy = _b5.true_traits_[corpus]['sberdisk']

_b5.get_avt_predictions(url_accuracy = url_accuracy, lang = 'en')
```

[2023-12-16 19:24:17] Feature extraction (hand-crafted and deep) from text ...

[2023-12-16 19:24:19] Getting scores and accuracy calculation (multimodal fusion) ...

10 from 10 (100.0%) ... GitHub:nbsphinx-math:*OCEANAI*_guide:nbsphinx-math:*notebooks*_FI:nbsphinx-math:*test*_plk5k7PBEg.003.mp4
...

Person ID	Path	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
1	2d6btbaNdfo.000.mp4	0.581159	0.628822	0.466609	0.622129	0.553832
2	300gK3CnzW0.001.mp4	0.463991	0.418851	0.41301	0.493329	0.423093
3	300gK3CnzW0.003.mp4	0.454281	0.415049	0.39189	0.485114	0.420741
4	4vdJGgZpj4k.003.mp4	0.588461	0.643233	0.530789	0.603038	0.593398
5	be0DQawtVkE.002.mp4	0.633433	0.533295	0.523742	0.608591	0.588456
6	cLaZxEf1nE4.004.mp4	0.636944	0.542386	0.558461	0.570975	0.558983
7	g24JGYuT74A.004.mp4	0.531518	0.376987	0.393309	0.4904	0.447881
8	JZNMxa3OKHY.000.mp4	0.610342	0.541418	0.563163	0.595013	0.569461
9	nvlqJbHk_Lc.003.mp4	0.495809	0.458526	0.414436	0.469152	0.435461
10	_plk5k7PBEg.003.mp4	0.60707	0.591893	0.520662	0.603938	0.565726

[2023-12-16 19:24:19] Trait-wise accuracy ...

Metrics	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism	Mean
MAE	0.0589	0.0612	0.0864	0.0697		
Accuracy	0.9411	0.9388	0.9136	0.9303		

(continues on next page)

(continued from previous page)

```
Metrics
MAE                 0.0582  0.0669
Accuracy            0.9418  0.9331
```

[2023-12-16 19:24:19] Mean absolute errors: 0.0669, average accuracy: 0.9331 ...

Log files saved successfully ...

— Runtime: 67.109 sec. —

[2]:
```
True
```

To find the suitable colleague, you need to know two correlation coefficients for each of a personality traits. These coefficients should show how the trait score of one person changes when it is higher or lower than the same trait score of another person.

As an example, it is proposed to use the correlation coefficients between two people in the context of a manager-employee relationship presented in the article:

1) Kuroda S., Yamamoto I. Good boss, bad boss, workers' mental health and productivity: Evidence from Japan // Japan & The World Economy. – 2018. – vol. 48. – pp. 106-118.

The user can set their own correlation coefficients.

[3]:
```
# Loading dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478675819?token=LuB7L1QsEY0UuSs&
↪filename=colleague_ranking.csv'
df_correlation_coefficients = pd.read_csv(url)
df_correlation_coefficients = pd.DataFrame(
    df_correlation_coefficients.drop(['ID'], axis = 1)
)

df_correlation_coefficients.index.name = 'ID'
df_correlation_coefficients.index += 1
df_correlation_coefficients.index = df_correlation_coefficients.index.map(str)

df_correlation_coefficients
```

[3]:
```
   Score_comparison  Openness  Conscientiousness  Extraversion  Agreeableness  \
ID
1            higher   -0.0602             0.0471       -0.1070        -0.0832
2             lower   -0.1720            -0.1050        0.0772         0.0703

    Non-Neuroticism
ID
1             0.190
2            -0.229
```

Finding a suitable senior colleague

[4]:
```
# List of personality traits scores of the target person
target_scores = [0.527886, 0.522337, 0.458468, 0.51761, 0.444649]

_b5._colleague_ranking(
    correlation_coefficients = df_correlation_coefficients,
    target_scores = target_scores,
    colleague = 'major',
    equal_coefficients = 0.5,
    out = False
)

_b5._save_logs(df = _b5.df_files_colleague_, name = 'major_colleague_ranking_fi_en', out␣
↪= True)

# Optional
df = _b5.df_files_colleague_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[4]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Match
7	g24JGYuT74A.004.mp4	0.532	0.377	0.393	0.490	0.448	0.078
4	4vdJGgZpj4k.003.mp4	0.588	0.643	0.531	0.603	0.593	0.001
1	2d6btbaNdfo.000.mp4	0.581	0.629	0.467	0.622	0.554	-0.002
10	_plk5k7PBEg.003.mp4	0.607	0.592	0.521	0.604	0.566	-0.007
5	be0DQawtVkE.002.mp4	0.633	0.533	0.524	0.609	0.588	-0.008
8	JZNMxa3OKHY.000.mp4	0.610	0.541	0.563	0.595	0.569	-0.013
6	cLaZxEf1nE4.004.mp4	0.637	0.542	0.558	0.571	0.559	-0.014
3	300gK3CnzW0.003.mp4	0.454	0.415	0.392	0.485	0.421	-0.154
2	300gK3CnzW0.001.mp4	0.464	0.419	0.413	0.493	0.423	-0.154
9	nvlqJbHk_Lc.003.mp4	0.496	0.459	0.414	0.469	0.435	-0.168

Finding a suitable junior colleague

[5]:
```
# List of personality traits scores of the target person
target_scores = [0.527886, 0.522337, 0.458468, 0.51761, 0.444649]

_b5._colleague_ranking(
    correlation_coefficients = df_correlation_coefficients,
    target_scores = target_scores,
    colleague = 'minor',
    equal_coefficients = 0.5,
    out = False
)

_b5._save_logs(df = _b5.df_files_colleague_, name = 'minor_colleague_ranking_fi_en', out␣
↪= True)
```

(continues on next page)

(continued from previous page)

```
# Optional
df = _b5.df_files_colleague_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[5]:

	Path	OPE	CON	EXT	AGR	NNEU	Match
Person ID							
9	nvlqJbHk_Lc.003.mp4	0.496	0.459	0.414	0.469	0.435	-0.009
3	300gK3CnzW0.003.mp4	0.454	0.415	0.392	0.485	0.421	-0.010
2	300gK3CnzW0.001.mp4	0.464	0.419	0.413	0.493	0.423	-0.013
8	JZNMxa3OKHY.000.mp4	0.610	0.541	0.563	0.595	0.569	-0.207
6	cLaZxEf1nE4.004.mp4	0.637	0.542	0.558	0.571	0.559	-0.211
1	2d6btbaNdfo.000.mp4	0.581	0.629	0.467	0.622	0.554	-0.213
10	_plk5k7PBEg.003.mp4	0.607	0.592	0.521	0.604	0.566	-0.213
5	be0DQawtVkE.002.mp4	0.633	0.533	0.524	0.609	0.588	-0.216
4	4vdJGgZpj4k.003.mp4	0.588	0.643	0.531	0.603	0.593	-0.221
7	g24JGYuT74A.004.mp4	0.532	0.377	0.393	0.490	0.448	-0.259

MuPTA (ru)

[6]:
```
import os
import pandas as pd

# Module import
from oceanai.modules.lab.build import Run

# Creating an instance of a class
_b5 = Run(lang = 'en')

corpus = 'mupta'
lang = 'ru'

# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

# Building audio models
res_load_model_hc = _b5.load_audio_model_hc()
res_load_model_nn = _b5.load_audio_model_nn()

# Loading audio model weights
url = _b5.weights_for_big5_['audio'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_audio_model_weights_hc(url = url)

url = _b5.weights_for_big5_['audio'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_audio_model_weights_nn(url = url)

# Building video models
```

(continues on next page)

(continued from previous page)

```
res_load_model_hc = _b5.load_video_model_hc(lang=lang)
res_load_model_deep_fe = _b5.load_video_model_deep_fe()
res_load_model_nn = _b5.load_video_model_nn()

# Loading video model weights
url = _b5.weights_for_big5_['video'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_video_model_weights_hc(url = url)

url = _b5.weights_for_big5_['video'][corpus]['fe']['sberdisk']
res_load_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(url = url)

url = _b5.weights_for_big5_['video'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_video_model_weights_nn(url = url)

# Loading a dictionary with hand-crafted features (text modality)
res_load_text_features = _b5.load_text_features()

# Building text models
res_setup_translation_model = _b5.setup_translation_model()
res_setup_translation_model = _b5.setup_bert_encoder()
res_load_text_model_hc_fi = _b5.load_text_model_hc(corpus=corpus)
res_load_text_model_nn_fi = _b5.load_text_model_nn(corpus=corpus)

# Loading text model weights
url = _b5.weights_for_big5_['text'][corpus]['hc']['sberdisk']
res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(url = url)

url = _b5.weights_for_big5_['text'][corpus]['nn']['sberdisk']
res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(url = url)

# Building model for multimodal information fusion
res_load_avt_model_b5 = _b5.load_avt_model_b5()

# Loading model weights for multimodal information fusion
url = _b5.weights_for_big5_['avt'][corpus]['b5']['sberdisk']
res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(url = url)

PATH_TO_DIR = './video_MuPTA/'
PATH_SAVE_VIDEO = './video_MuPTA/test/'

_b5.path_to_save_ = PATH_SAVE_VIDEO

# Loading 10 test files from the MuPTA corpus
# URL: https://hci.nw.ru/en/pages/mupta-corpus
domain = 'https://download.sberdisk.ru/download/file/'
tets_name_files = [
    '477995979?token=2cvyk7CS0mHx2MJ&filename=speaker_06_center_83.mov',
    '477995980?token=jGPtBPS69uzFU6Y&filename=speaker_01_center_83.mov',
    '477995967?token=zCaRbNB6ht5wMPq&filename=speaker_11_center_83.mov',
    '477995966?token=B1rbinDYRQKrI3T&filename=speaker_15_center_83.mov',
    '477995978?token=dEpVDtZg1EQiEQ9&filename=speaker_07_center_83.mov',
    '477995961?token=o1hVjw8G45q9L9Z&filename=speaker_19_center_83.mov',
```

(continues on next page)

(continued from previous page)

```
    '477995964?token=5K220Aqf673VHPq&filename=speaker_23_center_83.mov',
    '477995965?token=v1LVD2KT1cU7Lpb&filename=speaker_24_center_83.mov',
    '477995962?token=tmaSGyyWLA6XCy9&filename=speaker_27_center_83.mov',
    '477995963?token=bTpo96qNDPcwGqb&filename=speaker_10_center_83.mov',
]

for curr_files in tets_name_files:
    _b5.download_file_from_url(url = domain + curr_files, out = True)

# Getting scores
_b5.path_to_dataset_ = PATH_TO_DIR # Dataset directory
_b5.ext_ = ['.mov'] # Search file extensions

# Full path to the file with ground truth scores for accuracy calculation
url_accuracy = _b5.true_traits_['mupta']['sberdisk']

_b5.get_avt_predictions(url_accuracy = url_accuracy, lang = lang)
```

[2023-12-16 19:32:56] Feature extraction (hand-crafted and deep) from text ...

[2023-12-16 19:33:00] Getting scores and accuracy calculation (multimodal fusion) ...

10 from 10 (100.0%) ... GitHub:nbsphinx-math:*OCEANAI* _guide:nbsphinx-math:*notebooks* _MuPTA:nbsphinx-math:*test* _27_center_83.mov
...

Person ID	Path	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
1	speaker_01_center_83.mov	0.758137	0.693356	0.650108	0.744589	0.488671
2	speaker_06_center_83.mov	0.681602	0.654339	0.607156	0.731282	0.417908
3	speaker_07_center_83.mov	0.666104	0.656836	0.567863	0.685067	0.378102
4	speaker_10_center_83.mov	0.694171	0.596195	0.571414	0.66223	0.348639
5	speaker_11_center_83.mov	0.712885	0.594764	0.571709	0.716696	0.37802
6	speaker_15_center_83.mov	0.664158	0.670411	0.60421	0.696056	0.399842
7	speaker_19_center_83.mov	0.761213	0.652635	0.651028	0.788677	0.459676
8	speaker_23_center_83.mov	0.692788	0.68324	0.616737	0.795205	0.447242
9	speaker_24_center_83.mov	0.705923	0.658382	0.610645	0.697415	0.411988
10	speaker_27_center_83.mov	0.753417	0.708372	0.654608	0.816416	0.504743

[2023-12-16 19:33:00] Trait-wise accuracy ...

	Openness	Conscientiousness	Extraversion	Agreeableness	\

(continues on next page)

(continued from previous page)

```
Metrics
MAE         0.0673              0.0789         0.1325          0.102
Accuracy    0.9327              0.9211         0.8675          0.898

          Non-Neuroticism    Mean
Metrics
MAE                0.1002  0.0962
Accuracy           0.8998  0.9038
```

[2023-12-16 19:33:00] Mean absolute errors: 0.0962, average accuracy: 0.9038 ...

Log files saved successfully ...

— Runtime: 444.191 sec. —

[6]:
```
True
```

To find the suitable colleague, you need to know two correlation coefficients for each of a personality traits. These coefficients should show how the trait score of one person changes when it is higher or lower than the same trait score of another person.

As an example, it is proposed to use the correlation coefficients between two people in the context of a manager-employee relationship presented in the article:

1) Kuroda S., Yamamoto I. Good boss, bad boss, workers' mental health and productivity: Evidence from Japan // Japan & The World Economy. – 2018. – vol. 48. – pp. 106-118.

The user can set their own correlation coefficients.

[7]:
```
# Loading dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478675819?token=LuB7L1QsEYOUuSs&
↪filename=colleague_ranking.csv'
df_correlation_coefficients = pd.read_csv(url)
df_correlation_coefficients = pd.DataFrame(
    df_correlation_coefficients.drop(['ID'], axis = 1)
)

df_correlation_coefficients.index.name = 'ID'
df_correlation_coefficients.index += 1
df_correlation_coefficients.index = df_correlation_coefficients.index.map(str)

df_correlation_coefficients
```

[7]:
```
   Score_comparison  Openness  Conscientiousness  Extraversion  Agreeableness  \
ID
1            higher   -0.0602             0.0471       -0.1070        -0.0832
2             lower   -0.1720            -0.1050        0.0772         0.0703

    Non-Neuroticism
ID
1             0.190
2            -0.229
```

Finding a suitable senior colleague

[8]:
```
# List of personality traits scores of the target person
target_scores = [0.527886, 0.522337, 0.458468, 0.51761, 0.444649]

_b5._colleague_ranking(
    correlation_coefficients = df_correlation_coefficients,
    target_scores = target_scores,
    colleague = 'major',
    equal_coefficients = 0.5,
    out = False
)

_b5._save_logs(df = _b5.df_files_colleague_, name = 'major_colleague_ranking_mupta_ru',␣
↪out = True)

# Optional
df = _b5.df_files_colleague_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[8]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Match
1	speaker_01_center_83.mov	0.758	0.693	0.650	0.745	0.489	-0.052
10	speaker_27_center_83.mov	0.753	0.708	0.655	0.816	0.505	-0.054
8	speaker_23_center_83.mov	0.693	0.683	0.617	0.795	0.447	-0.057
7	speaker_19_center_83.mov	0.761	0.653	0.651	0.789	0.460	-0.063
4	speaker_10_center_83.mov	0.694	0.596	0.571	0.662	0.349	-0.210
3	speaker_07_center_83.mov	0.666	0.657	0.568	0.685	0.378	-0.214
5	speaker_11_center_83.mov	0.713	0.595	0.572	0.717	0.378	-0.222
6	speaker_15_center_83.mov	0.664	0.670	0.604	0.696	0.400	-0.223
9	speaker_24_center_83.mov	0.706	0.658	0.611	0.697	0.412	-0.229
2	speaker_06_center_83.mov	0.682	0.654	0.607	0.731	0.418	-0.232

Finding a suitable junior colleague

[9]:
```
# List of personality traits scores of the target person
target_scores = [0.527886, 0.522337, 0.458468, 0.51761, 0.444649]

_b5._colleague_ranking(
    correlation_coefficients = df_correlation_coefficients,
    target_scores = target_scores,
    colleague = 'minor',
    equal_coefficients = 0.5,
    out = False
)

_b5._save_logs(df = _b5.df_files_colleague_, name = 'minor_colleague_ranking_mupta_ru',␣
↪out = True)
```

(continues on next page)

(continued from previous page)

```
# Optional
df = _b5.df_files_colleague_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[9]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Match
2	speaker_06_center_83.mov	0.682	0.654	0.607	0.731	0.418	-0.008
6	speaker_15_center_83.mov	0.664	0.670	0.604	0.696	0.400	-0.013
9	speaker_24_center_83.mov	0.706	0.658	0.611	0.697	0.412	-0.016
5	speaker_11_center_83.mov	0.713	0.595	0.572	0.717	0.378	-0.019
3	speaker_07_center_83.mov	0.666	0.657	0.568	0.685	0.378	-0.020
4	speaker_10_center_83.mov	0.694	0.596	0.571	0.662	0.349	-0.025
8	speaker_23_center_83.mov	0.693	0.683	0.617	0.795	0.447	-0.190
7	speaker_19_center_83.mov	0.761	0.653	0.651	0.789	0.460	-0.199
10	speaker_27_center_83.mov	0.753	0.708	0.655	0.816	0.505	-0.212
1	speaker_01_center_83.mov	0.758	0.693	0.650	0.745	0.489	-0.213

MuPTA (en)

[10]:

```
import os
import pandas as pd

# Module import
from oceanai.modules.lab.build import Run

# Creating an instance of a class
_b5 = Run(lang = 'en')

corpus = 'fi'
lang = 'en'

# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

# Building audio models
res_load_model_hc = _b5.load_audio_model_hc()
res_load_model_nn = _b5.load_audio_model_nn()

# Loading audio model weights
url = _b5.weights_for_big5_['audio'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_audio_model_weights_hc(url = url)

url = _b5.weights_for_big5_['audio'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_audio_model_weights_nn(url = url)

# Building video models
```

(continues on next page)

(continued from previous page)

```
res_load_model_hc = _b5.load_video_model_hc(lang=lang)
res_load_model_deep_fe = _b5.load_video_model_deep_fe()
res_load_model_nn = _b5.load_video_model_nn()

# Loading video model weights
url = _b5.weights_for_big5_['video'][corpus]['hc']['sberdisk']
res_load_model_weights_hc = _b5.load_video_model_weights_hc(url = url)

url = _b5.weights_for_big5_['video'][corpus]['fe']['sberdisk']
res_load_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(url = url)

url = _b5.weights_for_big5_['video'][corpus]['nn']['sberdisk']
res_load_model_weights_nn = _b5.load_video_model_weights_nn(url = url)

# Loading a dictionary with hand-crafted features (text modality)
res_load_text_features = _b5.load_text_features()

# Building text models
res_setup_translation_model = _b5.setup_translation_model()
res_setup_translation_model = _b5.setup_bert_encoder()
res_load_text_model_hc_fi = _b5.load_text_model_hc(corpus=corpus)
res_load_text_model_nn_fi = _b5.load_text_model_nn(corpus=corpus)

# Loading text model weights
url = _b5.weights_for_big5_['text'][corpus]['hc']['sberdisk']
res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(url = url)

url = _b5.weights_for_big5_['text'][corpus]['nn']['sberdisk']
res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(url = url)

# Building model for multimodal information fusion
res_load_avt_model_b5 = _b5.load_avt_model_b5()

# Building model for multimodal information fusion
url = _b5.weights_for_big5_['avt'][corpus]['b5']['sberdisk']
res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(url = url)

PATH_TO_DIR = './video_MuPTA/'
PATH_SAVE_VIDEO = './video_MuPTA/test/'

_b5.path_to_save_ = PATH_SAVE_VIDEO

# Loading 10 test files from the MuPTA corpus
# URL: https://hci.nw.ru/en/pages/mupta-corpus
domain = 'https://download.sberdisk.ru/download/file/'
tets_name_files = [
    '477995979?token=2cvyk7CS0mHx2MJ&filename=speaker_06_center_83.mov',
    '477995980?token=jGPtBPS69uzFU6Y&filename=speaker_01_center_83.mov',
    '477995967?token=zCaRbNB6ht5wMPq&filename=speaker_11_center_83.mov',
    '477995966?token=B1rbinDYRQKrI3T&filename=speaker_15_center_83.mov',
    '477995978?token=dEpVDtZg1EQiEQ9&filename=speaker_07_center_83.mov',
    '477995961?token=o1hVjw8G45q9L9Z&filename=speaker_19_center_83.mov',
```

(continues on next page)

(continued from previous page)

```
    '477995964?token=5K220Aqf673VHPq&filename=speaker_23_center_83.mov',
    '477995965?token=v1LVD2KT1cU7Lpb&filename=speaker_24_center_83.mov',
    '477995962?token=tmaSGyyWLA6XCy9&filename=speaker_27_center_83.mov',
    '477995963?token=bTpo96qNDPcwGqb&filename=speaker_10_center_83.mov',
]

for curr_files in tets_name_files:
    _b5.download_file_from_url(url = domain + curr_files, out = True)

# Getting scores
_b5.path_to_dataset_ = PATH_TO_DIR # Dataset directory
_b5.ext_ = ['.mov'] # Search file extensions

# Full path to the file with ground truth scores for accuracy calculation
url_accuracy = _b5.true_traits_['mupta']['sberdisk']

_b5.get_avt_predictions(url_accuracy = url_accuracy, lang = lang)
```

[2023-12-16 19:40:25] Feature extraction (hand-crafted and deep) from text ...

[2023-12-16 19:40:28] Getting scores and accuracy calculation (multimodal fusion) ...

10 from 10 (100.0%) ... GitHub:nbsphinx-math:*OCEANAI*_guide:nbsphinx-math:*notebooks*_MuPTA:nbsphinx-math:*test*_27_center_83.mov
...

Person ID	Path	Openness	Conscientiousness	Extraversion	Agreeableness	Non-Neuroticism
1	speaker_01_center_83.mov	0.564985	0.539052	0.440615	0.59251	0.488763
2	speaker_06_center_83.mov	0.650774	0.663849	0.607308	0.643847	0.620627
3	speaker_07_center_83.mov	0.435976	0.486683	0.313828	0.415446	0.396618
4	speaker_10_center_83.mov	0.498542	0.511243	0.412592	0.468947	0.44399
5	speaker_11_center_83.mov	0.394776	0.341608	0.327082	0.427304	0.354936
6	speaker_15_center_83.mov	0.566107	0.543811	0.492766	0.587411	0.499433
7	speaker_19_center_83.mov	0.506271	0.438215	0.430894	0.456177	0.44075
8	speaker_23_center_83.mov	0.486463	0.521755	0.309894	0.432291	0.433601
9	speaker_24_center_83.mov	0.417404	0.473339	0.320714	0.445086	0.414649
10	speaker_27_center_83.mov	0.526112	0.661107	0.443167	0.558965	0.554224

[2023-12-16 19:40:28] Trait-wise accuracy ...

	Openness	Conscientiousness	Extraversion	Agreeableness	\

(continues on next page)

(continued from previous page)

```
Metrics
MAE         0.1727              0.1672        0.1661          0.2579
Accuracy    0.8273              0.8328        0.8339          0.7421

          Non-Neuroticism    Mean
Metrics
MAE                 0.107  0.1742
Accuracy            0.893  0.8258
```

[2023-12-16 19:40:28] Mean absolute errors: 0.1742, average accuracy: 0.8258 ...

Log files saved successfully ...

— Runtime: 377.119 sec. —

[10]:
```
True
```

To find the suitable colleague, you need to know two correlation coefficients for each of a personality traits. These coefficients should show how the trait score of one person changes when it is higher or lower than the same trait score of another person.

As an example, it is proposed to use the correlation coefficients between two people in the context of a manager-employee relationship presented in the article:

1) Kuroda S., Yamamoto I. Good boss, bad boss, workers' mental health and productivity: Evidence from Japan // Japan & The World Economy. – 2018. – vol. 48. – pp. 106-118.

The user can set their own correlation coefficients.

[11]:
```
# Loading dataframe with correlation coefficients
url = 'https://download.sberdisk.ru/download/file/478675819?token=LuB7L1QsEYOUuSs&
↪filename=colleague_ranking.csv'
df_correlation_coefficients = pd.read_csv(url)
df_correlation_coefficients = pd.DataFrame(
    df_correlation_coefficients.drop(['ID'], axis = 1)
)

df_correlation_coefficients.index.name = 'ID'
df_correlation_coefficients.index += 1
df_correlation_coefficients.index = df_correlation_coefficients.index.map(str)

df_correlation_coefficients
```

[11]:
```
   Score_comparison  Openness  Conscientiousness  Extraversion  Agreeableness  \
ID
1            higher   -0.0602             0.0471       -0.1070        -0.0832
2             lower   -0.1720            -0.1050        0.0772         0.0703

    Non-Neuroticism
ID
1             0.190
2            -0.229
```

Finding a suitable senior colleague

[12]:
```
# List of personality traits scores of the target person
target_scores = [0.527886, 0.522337, 0.458468, 0.51761, 0.444649]

_b5._colleague_ranking(
    correlation_coefficients = df_correlation_coefficients,
    target_scores = target_scores,
    colleague = 'major',
    equal_coefficients = 0.5,
    out = False
)

_b5._save_logs(df = _b5.df_files_colleague_, name = 'major_colleague_ranking_mupta_en',␣
↪out = True)

# Optional
df = _b5.df_files_colleague_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[12]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Match
1	speaker_01_center_83.mov	0.565	0.539	0.441	0.593	0.489	0.069
10	speaker_27_center_83.mov	0.526	0.661	0.443	0.559	0.554	0.034
2	speaker_06_center_83.mov	0.651	0.664	0.607	0.644	0.621	-0.009
6	speaker_15_center_83.mov	0.566	0.544	0.493	0.587	0.499	-0.015
5	speaker_11_center_83.mov	0.395	0.342	0.327	0.427	0.355	-0.130
9	speaker_24_center_83.mov	0.417	0.473	0.321	0.445	0.415	-0.160
3	speaker_07_center_83.mov	0.436	0.487	0.314	0.415	0.397	-0.163
7	speaker_19_center_83.mov	0.506	0.438	0.431	0.456	0.441	-0.169
4	speaker_10_center_83.mov	0.499	0.511	0.413	0.469	0.444	-0.176
8	speaker_23_center_83.mov	0.486	0.522	0.310	0.432	0.434	-0.183

Finding a suitable junior colleague

[13]:
```
# List of personality traits scores of the target person
target_scores = [0.527886, 0.522337, 0.458468, 0.51761, 0.444649]

_b5._colleague_ranking(
    correlation_coefficients = df_correlation_coefficients,
    target_scores = target_scores,
    colleague = 'minor',
    equal_coefficients = 0.5,
    out = False
)

_b5._save_logs(df = _b5.df_files_colleague_, name = 'minor_colleague_ranking_mupta_en',␣
↪out = True)
```

(continues on next page)

(continued from previous page)

```
# Optional
df = _b5.df_files_colleague_.rename(columns = {'Openness':'OPE', 'Conscientiousness':'CON
↪', 'Extraversion': 'EXT', 'Agreeableness': 'AGR', 'Non-Neuroticism': 'NNEU'})
columns_to_round = df.columns[1:]
df[columns_to_round] = df[columns_to_round].apply(lambda x: [round(i, 3) for i in x])
df
```

[13]:

Person ID	Path	OPE	CON	EXT	AGR	NNEU	Match
8	speaker_23_center_83.mov	0.486	0.522	0.310	0.432	0.434	0.009
9	speaker_24_center_83.mov	0.417	0.473	0.321	0.445	0.415	0.005
3	speaker_07_center_83.mov	0.436	0.487	0.314	0.415	0.397	0.004
4	speaker_10_center_83.mov	0.499	0.511	0.413	0.469	0.444	-0.005
7	speaker_19_center_83.mov	0.506	0.438	0.431	0.456	0.441	-0.010
5	speaker_11_center_83.mov	0.395	0.342	0.327	0.427	0.355	-0.011
6	speaker_15_center_83.mov	0.566	0.544	0.493	0.587	0.499	-0.189
2	speaker_06_center_83.mov	0.651	0.664	0.607	0.644	0.621	-0.232
10	speaker_27_center_83.mov	0.526	0.661	0.443	0.559	0.554	-0.236
1	speaker_01_center_83.mov	0.565	0.539	0.441	0.593	0.489	-0.271

Audio information processing

Formation of the neural network architecture of the model and downloading its weights to obtain features / scores based on hand-crafted features (audio modality)

- `_b5.audio_model_hc_` - Neural network model **tf.keras.Model** for obtaining features / scores based on hand-crafted features

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Interface language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-10 16:37:47] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a5 License: BSD License

Formation of the neural network architecture of the model

[4]:
```
res_load_audio_model_hc = _b5.load_audio_model_hc(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2022-12-11 12:20:55] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (audio modality) ...

— Runtime: 3.03 sec. —

Downloading the weights of the neural network model

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['audio']['fi']['hc']['sberdisk']

res_load_audio_model_weights_hc = _b5.load_audio_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 16:38:05] Downloading the weights of the neural network model to obtain scores by hand-crafted features (audio modality) ...

[2023-12-10 16:38:05] File download "weights_2022-05-05_11-27-55.h5" (100.0%) ...

— Runtime: 0.458 sec. —

Displaying the formed neural network architecture of the model

[6]:
```
_b5.audio_model_hc_.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 196, 25)]         0

 lstm (LSTM)                 (None, 196, 64)           23040

 dropout (Dropout)           (None, 196, 64)           0

 lstm_128_a_hc (LSTM)        (None, 128)               98816

 dropout_1 (Dropout)         (None, 128)               0

 dense (Dense)               (None, 5)                 645

=================================================================
Total params: 122,501
Trainable params: 122,501
Non-trainable params: 0
_________________________________________________________________
```

Formation of the neural network architecture of the model and downloading its weights to obtain features / scores based on deep features (audio modality)

- `_b5.audio_model_nn_` - Neural network model **tf.keras.Model** for obtaining features / scores based on deep features

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Interface language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
```

(continues on next page)

(continued from previous page)

```
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-10 16:45:19] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a5 License: BSD License

Formation of the neural network architecture of the model

[4]:
```
res_load_audio_model_nn = _b5.load_audio_model_nn(
    show_summary = False, # Display of the generated neural network архитектуры модели
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 16:45:19] Formation of a neural network architecture for obtaining scores by deep features (audio modality) ...

— Runtime: 1.221 sec. —

Downloading the weights of the neural network model

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['audio']['fi']['nn']['sberdisk']

res_load_audio_model_weights_nn = _b5.load_audio_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 16:45:23] Downloading the weights of the neural network model to obtain scores by deep features (audio modality) ...

[2023-12-10 16:45:27] File download "weights_2022-05-03_07-46-14.h5" (100.0%) ...

— Runtime: 4.175 sec. —

Displaying the formed neural network architecture of the model

[6]:
```
_b5.audio_model_nn_.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0

 block4_conv1 (Conv2D)       (None, 28, 28, 512)       1180160

 block4_conv2 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv3 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 14, 14, 512)       0

 block5_conv1 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv2 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv3 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 7, 7, 512)         0

 flatten (Flatten)           (None, 25088)             0

 dense (Dense)               (None, 512)               12845568
```

(continues on next page)

(continued from previous page)

```
 dropout (Dropout)           (None, 512)               0

 dense_256 (Dense)           (None, 256)               131328

 dense_1 (Dense)             (None, 5)                 1285

=================================================================
Total params: 27,692,869
Trainable params: 27,692,869
Non-trainable params: 0
_________________________________________________________________
```

Formation of neural network architectures of models and downloading their weights to obtain the personality traits scores (audio modality)

- `_b5.audio_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

```
[2023-12-14 11:10:51] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a16 License: BSD License
```

Formation of neural network architectures of models

[4]:
```
res_load_audio_models_b5 = _b5.load_audio_models_b5(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 11:10:51]Formation of neural network architectures of models for obtaining the personality traits scores (audio modality) ...

— Runtime: 0.157 sec. —

Downloading weights of neural network models

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url_openness = _b5.weights_for_big5_['audio']['fi']['b5']['openness']['sberdisk']
url_conscientiousness = _b5.weights_for_big5_['audio']['fi']['b5']['conscientiousness']['sberdisk']
url_extraversion = _b5.weights_for_big5_['audio']['fi']['b5']['extraversion']['sberdisk']
url_agreeableness = _b5.weights_for_big5_['audio']['fi']['b5']['agreeableness']['sberdisk']
url_non_neuroticism = _b5.weights_for_big5_['audio']['fi']['b5']['non_neuroticism']['sberdisk']

res_load_audio_models_weights_b5 = _b5.load_audio_models_weights_b5(
    url_openness = url_openness, # Openness
    url_conscientiousness = url_conscientiousness, # Conscientiousness
    url_extraversion = url_extraversion, # Extraversion
    url_agreeableness = url_agreeableness, # Agreeableness
    url_non_neuroticism = url_non_neuroticism, # Non-Neuroticism
    force_reload = True, # Forced download of a file with weights of a neural network model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 11:11:23] Downloading the weights of neural network models to obtain the personality traits scores (audio modality) ...

[2023-12-14 11:11:23] File download "weights_2022-06-15_16-16-20.h5" (100.0%) ... Openness

[2023-12-14 11:11:23] File download "weights_2022-06-15_16-21-57.h5" (100.0%) ... Conscientiousness

[2023-12-14 11:11:23] File download "weights_2022-06-15_16-26-41.h5" (100.0%) ... Extraversion

[2023-12-14 11:11:23] File download "weights_2022-06-15_16-32-51.h5" (100.0%) ... Agreeableness

[2023-12-14 11:11:24] File download "weights_2022-06-15_16-37-46.h5" (100.0%) ... Non-Neuroticism

— Runtime: 0.907 sec. —

Displaying the formed neural network architecture of the model

- Openness
- Conscientiousness
- Extraversion
- Agreeableness
- Non-neuroticism

[6]:
```
_b5.audio_models_b5_['openness'].summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32)]              0

 dense_1 (Dense)             (None, 1)                 33

 activ_1 (Activation)        (None, 1)                 0

=================================================================
Total params: 33 (132.00 Byte)
Trainable params: 33 (132.00 Byte)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Extracting features from an acoustic signal

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Interface language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
```

(continues on next page)

(continued from previous page)

```
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-10 16:35:36] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a2 License: BSD License

Acoustic feature extraction process

[5]:
```
# Core settings
sr = 44100 # Sampling frequency
# Path to the audio or video file
path = 'video_FI/test/_plk5k7PBEg.003.mp4'

hc_features, melspectrogram_features = _b5.get_acoustic_features(
    path = path, # Path to the audio or video file
    sr = sr, # Sampling frequency
    window = 2, # Signal segment window size (in seconds)
    step = 1, # Signal segment window shift step (in seconds)
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 16:36:06] Extraction of features (hand-crafted and mel-spectrograms) from an acoustic signal ...

[2023-12-10 16:36:11] Statistics of the features extracted from the acoustic signal: Total number of segments with: 1. hand-crafted features: 12 2. mel-spectrogram log: 12 Dimension of the matrix of hand-crafted features of one segment: 196 × 25 Dimension of the tensor with log mel-spectrograms of one segment: 224 × 224 × 3

— Runtime: 5.292 sec. —

Getting audio scores

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Interface language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-14 16:54:20] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a16 License: BSD License

Getting and displaying versions of installed libraries

- `_b5.df_pkgs_` - DataFrame with versions of installed libraries

[4]:
```
_b5.libs_vers(runtime = True, run = True)
```

	Package	Version
1	TensorFlow	2.15.0
2	Keras	2.15.0
3	OpenCV	4.8.1
4	MediaPipe	0.9.0
5	NumPy	1.26.2
6	SciPy	1.11.4
7	Pandas	2.1.3
8	Scikit-learn	1.3.2
9	OpenSmile	2.5.0
10	Librosa	0.10.1
11	AudioRead	3.0.1
12	IPython	8.18.1
13	PyMediaInfo	6.1.0
14	Requests	2.31.0
15	JupyterLab	4.0.9
16	LIWC	0.5.0
17	Transformers	4.36.0
18	Sentencepiece	0.1.99
19	Torch	2.0.1+cpu
20	Torchaudio	2.0.2+cpu

— Runtime: 0.005 sec. —

Formation of the neural network architecture of the model for obtaining scores by hand-crafted features

- `_b5.audio_model_hc_` - Neural network model **tf.keras.Model** for obtaining scores by hand-crafted features

[5]:
```
res_load_audio_model_hc = _b5.load_audio_model_hc(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 16:54:20] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (audio modality) ...

— Runtime: 0.335 sec. —

Downloading the weights of the neural network model to obtain scores by hand-crafted features

- `_b5.audio_model_hc_` - Neural network model **tf.keras.Model** for obtaining scores by hand-crafted features

[6]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['audio']['fi']['hc']['sberdisk']

res_load_audio_model_weights_hc = _b5.load_audio_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 16:54:21] Downloading the weights of the neural network model to obtain scores by hand-crafted features (audio modality) ...

[2023-12-14 16:54:21] File download "weights_2022-05-05_11-27-55.h5" (100.0%) ...

— Runtime: 0.323 sec. —

Formation of the neural network architecture of the model for obtaining scores by deep features

- `_b5.audio_model_nn_` - Neural network model **tf.keras.Model** for obtaining scores by deep features

[7]:
```
res_load_audio_model_nn = _b5.load_audio_model_nn(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 16:54:21] Formation of a neural network architecture for obtaining scores by deep features (audio modality) ...

— Runtime: 0.212 sec. —

Downloading the weights of the neural network model to obtain scores by deep features

- `_b5.audio_model_nn_` - Neural network model **tf.keras.Model** for obtaining scores by deep features

[8]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['audio']['fi']['nn']['sberdisk']

res_load_audio_model_weights_nn = _b5.load_audio_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 16:54:21] Downloading the weights of the neural network model to obtain scores for deep features (audio modality) ...

[2023-12-14 16:54:22] File download "weights_2022-05-03_07-46-14.h5"

— Runtime: 0.416 sec. —

Formation of neural network architectures of models for obtaining the personality traits scores

- `_b5.audio_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

[9]:
```
res_load_audio_models_b5 = _b5.load_audio_models_b5(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 16:54:22] Formation of neural network architectures of models for obtaining personality traits scores (audio modality) ...
— Runtime: 0.067 sec. —

Downloading the weights of neural network models to obtain the personality traits scores

- `_b5.audio_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

[10]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url_openness = _b5.weights_for_big5_['audio']['fi']['b5']['openness']['sberdisk']
url_conscientiousness = _b5.weights_for_big5_['audio']['fi']['b5']['conscientiousness'][
↪'sberdisk']
url_extraversion = _b5.weights_for_big5_['audio']['fi']['b5']['extraversion']['sberdisk']
url_agreeableness = _b5.weights_for_big5_['audio']['fi']['b5']['agreeableness']['sberdisk
↪']
url_non_neuroticism = _b5.weights_for_big5_['audio']['fi']['b5']['non_neuroticism'][
↪'sberdisk']

res_load_audio_models_weights_b5 = _b5.load_audio_models_weights_b5(
    url_openness = url_openness, # Openness
    url_conscientiousness = url_conscientiousness, # Conscientiousness
    url_extraversion = url_extraversion, # Extraversion
    url_agreeableness = url_agreeableness, # Agreeableness
    url_non_neuroticism = url_non_neuroticism, # Non-Neuroticism
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 16:54:22] Downloading the weights of neural network models to obtain the personality traits scores (audio modality) ...	
[2023-12-14 16:54:22] File download "weights_2022-06-15_16-16-20.h5" (100.0%) ...	Openness
[2023-12-14 16:54:22] File download "weights_2022-06-15_16-21-57.h5" (100.0%) ...	Conscientiousness
[2023-12-14 16:54:22] File download "weights_2022-06-15_16-26-41.h5" (100.0%) ...	Extraversion
[2023-12-14 16:54:22] File download "weights_2022-06-15_16-32-51.h5" (100.0%) ...	Agreeableness
[2023-12-14 16:54:22] File download "weights_2022-06-15_16-37-46.h5" (100.0%) ...	Non-Neuroticism
— Runtime: 0.807 sec. —	

Getting scores (audio modality)

- `_b5.df_files_` - DataFrame with data
- `_b5.df_accuracy_` - DataFrame with accuracy

[11]:
```
# Core settings
_b5.path_to_dataset_ = 'E:/Databases/FirstImpressionsV2/test' # Dataset directory
# Directories not included in the set
_b5.ignore_dirs_ = []
# Key names for DataFrame dataset
_b5.keys_dataset_ = ['Path', 'Openness', 'Conscientiousness', 'Extraversion',
↪'Agreeableness', 'Non-Neuroticism']
_b5.ext_ = ['.mp4'] # Search file extensions
_b5.path_to_logs_ = './logs' # Directory for saving LOG files

# Full path to the file containing the ground truth scores for the accuracy calculation
url_accuracy = _b5.true_traits_['fi']['sberdisk']

res_get_audio_union_predictions = _b5.get_audio_union_predictions(
    depth = 2,          # Hierarchy depth for receiving audio and video data
    recursive = False, # Recursive data search
    sr = 44100,         # Sampling frequency
    window = 2,         # Signal segment window size (in seconds)
    step = 1,           # Signal segment window shift step (in seconds)
    accuracy = True,    # Accuracy
    url_accuracy = url_accuracy,
    logs = True,       # If necessary, generate a LOG file
    out = True,         # Display
    runtime = True,     # Runtime count
    run = True          # Run blocking
)
```

[2023-12-14 17:59:22] Getting scores and accuracy calculation (audio modality) ...

2000 from 2000 (100.0%) ... test80_25_Q4wOgixh7E.004.mp4 ...

ID	Path	Openness	\
1	E:\Databases\FirstImpressionsV2\test\test80_01...	0.603529	
2	E:\Databases\FirstImpressionsV2\test\test80_01...	0.568246	
3	E:\Databases\FirstImpressionsV2\test\test80_01...	0.546209	
4	E:\Databases\FirstImpressionsV2\test\test80_01...	0.691056	
5	E:\Databases\FirstImpressionsV2\test\test80_01...	0.690808	
6	E:\Databases\FirstImpressionsV2\test\test80_01...	0.65728	
7	E:\Databases\FirstImpressionsV2\test\test80_01...	0.453781	
8	E:\Databases\FirstImpressionsV2\test\test80_01...	0.558594	
9	E:\Databases\FirstImpressionsV2\test\test80_01...	0.529081	
10	E:\Databases\FirstImpressionsV2\test\test80_01...	0.537279	
11	E:\Databases\FirstImpressionsV2\test\test80_01...	0.512779	
12	E:\Databases\FirstImpressionsV2\test\test80_01...	0.447102	
13	E:\Databases\FirstImpressionsV2\test\test80_01...	0.368372	
14	E:\Databases\FirstImpressionsV2\test\test80_01...	0.582539	
15	E:\Databases\FirstImpressionsV2\test\test80_01...	0.627705	
16	E:\Databases\FirstImpressionsV2\test\test80_01...	0.708798	

(continues on next page)

(continued from previous page)

```
17  E:\Databases\FirstImpressionsV2\test\test80_01...  0.583968
18  E:\Databases\FirstImpressionsV2\test\test80_01...  0.550836
19  E:\Databases\FirstImpressionsV2\test\test80_01...  0.626745
20  E:\Databases\FirstImpressionsV2\test\test80_01...  0.593014
21  E:\Databases\FirstImpressionsV2\test\test80_01...  0.545921
22  E:\Databases\FirstImpressionsV2\test\test80_01...  0.548432
23  E:\Databases\FirstImpressionsV2\test\test80_01...  0.486083
24  E:\Databases\FirstImpressionsV2\test\test80_01...  0.558323
25  E:\Databases\FirstImpressionsV2\test\test80_01...  0.473017
26  E:\Databases\FirstImpressionsV2\test\test80_01...  0.530967
27  E:\Databases\FirstImpressionsV2\test\test80_01...   0.61807
28  E:\Databases\FirstImpressionsV2\test\test80_01...   0.64703
29  E:\Databases\FirstImpressionsV2\test\test80_01...  0.571473
30  E:\Databases\FirstImpressionsV2\test\test80_01...  0.655007

   Conscientiousness Extraversion Agreeableness Non-Neuroticism
ID
1           0.556223     0.526545      0.579621        0.547629
2           0.465263     0.460744      0.541769        0.511338
3           0.603946     0.469445      0.589493        0.545716
4           0.623856     0.628851      0.614669        0.645813
5           0.589734     0.636104      0.606598         0.63479
6           0.681336     0.571412      0.596052        0.623451
7           0.438842     0.376464      0.520368        0.438252
8           0.598366     0.452183      0.618858        0.571653
9           0.502482     0.426603      0.488263        0.443719
10          0.508283     0.438888      0.579794        0.512117
11          0.447352     0.422968      0.559107        0.491406
12          0.451113     0.364429      0.513031        0.414412
13          0.391985     0.274865       0.42951        0.307666
14          0.432871     0.412363      0.441974        0.462192
15          0.801831     0.528622      0.692623        0.691908
16          0.654007     0.640547      0.632052        0.669044
17          0.644164      0.50463      0.633507         0.59208
18          0.539624     0.468092      0.594872        0.544016
19          0.563271     0.556561      0.561901        0.549236
20          0.421482     0.504798      0.534224        0.532807
21          0.479671     0.465769      0.571302        0.518793
22          0.480831     0.453319       0.52774         0.47759
23          0.467779     0.396113      0.444633        0.399402
24          0.537912     0.474172      0.563599         0.52937
25          0.542138     0.370228      0.550093        0.467068
26          0.460241     0.410618      0.507322        0.450027
27          0.506396     0.572248      0.574811        0.563796
28          0.577771     0.565869      0.575279         0.60631
29          0.529536      0.48662      0.535691        0.529022
30          0.606712     0.592804      0.570543        0.600349
```

[2023-12-14 17:59:22] Trait-wise accuracy ...

```
          Openness  Conscientiousness  Extraversion  Agreeableness  \
Metrics
MAE         0.0916             0.0925        0.0932         0.0918
```

(continues on next page)

(continued from previous page)

```
Accuracy    0.9084             0.9075        0.9068          0.9082

          Non-Neuroticism    Mean
Metrics
MAE                0.094  0.0926
Accuracy           0.906  0.9074
```

[2023-12-14 17:59:22] Mean absolute error: 0.0926, Accuracy: 0.9074 ...

Log files saved successfully ...

— Runtime: 3899.26 sec. —

Video information processing

Formation of the neural network architecture of the model and downloading its weights to obtain features / scores based on hand-crafted features (video modality)

- `_b5.video_model_hc_` - Neural network model **tf.keras.Model** for obtaining features / scores by hand-crafted features

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-10 17:11:13] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a5 License: BSD License

Formation of the neural network architecture of the model (FI V2)

[4]:
```
res_load_video_model_hc = _b5.load_video_model_hc(
    lang = 'en', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 17:11:13] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (video modality) ...

— Runtime: 0.789 sec. —

Downloading the weights of the neural network model

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['video']['fi']['hc']['sberdisk']

res_load_video_model_weights_hc = _b5.load_video_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 17:11:14] Downloading the weights of the neural network model to obtain scores by hand-crafted features (video modality) ...

[2023-12-10 17:11:14] File download "weights_2022-08-27_18-53-35.h5" (100.0%) ...

— Runtime: 0.226 sec. —

Displaying the formed neural network architecture of the model

[6]:
```
_b5.video_model_hc_.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 10, 115)]         0

 lstm (LSTM)                 (None, 10, 64)            46080

 dropout (Dropout)           (None, 10, 64)            0
```

(continues on next page)

(continued from previous page)

```
 lstm_128_v_hc (LSTM)        (None, 128)               98816

 dropout_1 (Dropout)         (None, 128)               0

 dense (Dense)               (None, 5)                 645

=================================================================
Total params: 145,541
Trainable params: 145,541
Non-trainable params: 0
_________________________________________________________________
```

Formation of the neural network architecture of the model (MuPTA)

[7]:
```
res_load_video_model_hc = _b5.load_video_model_hc(
    lang = 'ru', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 17:11:14] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (video modality) ...

— Runtime: 0.25 sec. —

Downloading the weights of the neural network model

[8]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['video']['mupta']['hc']['sberdisk']

res_load_video_model_weights_hc = _b5.load_video_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 17:11:14] Downloading the weights of the neural network model to obtain scores by hand-crafted features (video modality) ...

[2023-12-10 17:11:15] File download "weights_2022-08-27_18-53-35.h5" (100.0%) ...

— Runtime: 0.307 sec. —

Displaying the formed neural network architecture of the model

[9]:
```
_b5.video_model_hc_.summary()
```

```
Model: "model_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 10, 109)]         0

 lstm_1 (LSTM)               (None, 10, 64)            44544

 dropout_2 (Dropout)         (None, 10, 64)            0

 lstm_128_v_hc (LSTM)        (None, 128)               98816

 dropout_3 (Dropout)         (None, 128)               0

 dense_1 (Dense)             (None, 5)                 645

=================================================================
Total params: 144,005
Trainable params: 144,005
Non-trainable params: 0
_________________________________________________________________
```

Formation of the neural network architecture of the model and downloading its weights to obtain deep features (video modality)

- `_b5.video_model_deep_fe_` - Neural network model **tf.keras.Model** for obtaining deep features

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
```

(continues on next page)

(continued from previous page)

```
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-10 17:08:31] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a5 License: BSD License

Formation of the neural network architecture of the model (FI V2)

[4]:
```
res_load_video_model_deep_fe = _b5.load_video_model_deep_fe(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 17:08:31] Formation of neural network architecture for obtaining deep features (video modality) ...

— Runtime: 1.118 sec. —

Downloading the weights of the neural network model

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['video']['fi']['fe']['sberdisk']

res_load_video_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 17:08:32] Downloading weights of a neural network model to obtain deep features (video modality) ...

[2023-12-10 17:08:36] File download "weights_2022-11-01_12-27-07.h5" (100.0%) ...

— Runtime: 4.042 sec. —

Displaying the formed neural network architecture of the model

[6]:
```
_b5.video_model_deep_fe_.summary()
```

```
Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_1 (InputLayer)           [(None, 224, 224, 3  0           []
                                )]

 conv1/7x7_s2 (Conv2D)          (None, 112, 112, 64  9408        ['input_1[0][0]']
                                )

 conv1/7x7_s2/bn (BatchNormaliz  (None, 112, 112, 64  256        ['conv1/7x7_s2[0][0]']
 ation)                         )

 activation (Activation)        (None, 112, 112, 64  0           ['conv1/7x7_s2/bn[0][0]
                                )                                 ']

 max_pooling2d (MaxPooling2D)   (None, 55, 55, 64)   0           ['activation[0][0]']

 conv2_1_1x1_reduce (Conv2D)    (None, 55, 55, 64)   4096        ['max_pooling2d[0][0]']

 conv2_1_1x1_reduce/bn (BatchNo  (None, 55, 55, 64)  256         ['conv2_1_1x1_
 rmalization)                                                    reduce[0][0]']

 activation_1 (Activation)      (None, 55, 55, 64)   0           ['conv2_1_1x1_reduce/
                                                                 bn[0][0]']

 conv2_1_3x3 (Conv2D)           (None, 55, 55, 64)   36864       ['activation_1[0][0]']

 conv2_1_3x3/bn (BatchNormaliza  (None, 55, 55, 64)  256         ['conv2_1_3x3[0][0]']
 tion)

 activation_2 (Activation)      (None, 55, 55, 64)   0           ['conv2_1_3x3/bn[0][0]']

 conv2_1_1x1_increase (Conv2D)  (None, 55, 55, 256)  16384       ['activation_2[0][0]']

 conv2_1_1x1_proj (Conv2D)      (None, 55, 55, 256)  16384       ['max_pooling2d[0][0]']

 conv2_1_1x1_increase/bn (Batch  (None, 55, 55, 256)  1024       ['conv2_1_1x1_
 Normalization)                                                  increase[0][0]']

 conv2_1_1x1_proj/bn (BatchNorm  (None, 55, 55, 256)  1024       ['conv2_1_1x1_proj[0][0]
 alization)                                                      ']

 add (Add)                      (None, 55, 55, 256)  0           ['conv2_1_1x1_increase/
```

(continues on next page)

(continued from previous page)

```
↪bn[0][0]',
                                                                'conv2_1_1x1_proj/
↪bn[0][0]']

 activation_3 (Activation)      (None, 55, 55, 256)  0           ['add[0][0]']

 conv2_2_1x1_reduce (Conv2D)    (None, 55, 55, 64)   16384       ['activation_3[0][0]']

 conv2_2_1x1_reduce/bn (BatchNo  (None, 55, 55, 64)  256         ['conv2_2_1x1_
↪reduce[0][0]']
 rmalization)

 activation_4 (Activation)      (None, 55, 55, 64)   0           ['conv2_2_1x1_reduce/
↪bn[0][0]']

 conv2_2_3x3 (Conv2D)           (None, 55, 55, 64)   36864       ['activation_4[0][0]']

 conv2_2_3x3/bn (BatchNormaliza  (None, 55, 55, 64)  256         ['conv2_2_3x3[0][0]']
 tion)

 activation_5 (Activation)      (None, 55, 55, 64)   0           ['conv2_2_3x3/bn[0][0]']

 conv2_2_1x1_increase (Conv2D)  (None, 55, 55, 256)  16384       ['activation_5[0][0]']

 conv2_2_1x1_increase/bn (Batch  (None, 55, 55, 256)  1024       ['conv2_2_1x1_
↪increase[0][0]']
 Normalization)

 add_1 (Add)                    (None, 55, 55, 256)  0           ['conv2_2_1x1_increase/
↪bn[0][0]',
                                                                 'activation_3[0][0]']

 activation_6 (Activation)      (None, 55, 55, 256)  0           ['add_1[0][0]']

 conv2_3_1x1_reduce (Conv2D)    (None, 55, 55, 64)   16384       ['activation_6[0][0]']

 conv2_3_1x1_reduce/bn (BatchNo  (None, 55, 55, 64)  256         ['conv2_3_1x1_
↪reduce[0][0]']
 rmalization)

 activation_7 (Activation)      (None, 55, 55, 64)   0           ['conv2_3_1x1_reduce/
↪bn[0][0]']

 conv2_3_3x3 (Conv2D)           (None, 55, 55, 64)   36864       ['activation_7[0][0]']

 conv2_3_3x3/bn (BatchNormaliza  (None, 55, 55, 64)  256         ['conv2_3_3x3[0][0]']
 tion)

 activation_8 (Activation)      (None, 55, 55, 64)   0           ['conv2_3_3x3/bn[0][0]']

 conv2_3_1x1_increase (Conv2D)  (None, 55, 55, 256)  16384       ['activation_8[0][0]']
```

(continues on next page)

(continued from previous page)

```
conv2_3_1x1_increase/bn (Batch  (None, 55, 55, 256)  1024       ['conv2_3_1x1_
↪increase[0][0]']
Normalization)

add_2 (Add)                    (None, 55, 55, 256)  0          ['conv2_3_1x1_increase/
↪bn[0][0]',
                                                                 'activation_6[0][0]']

activation_9 (Activation)      (None, 55, 55, 256)  0          ['add_2[0][0]']

conv3_1_1x1_reduce (Conv2D)    (None, 28, 28, 128)  32768      ['activation_9[0][0]']

conv3_1_1x1_reduce/bn (BatchNo  (None, 28, 28, 128)  512       ['conv3_1_1x1_
↪reduce[0][0]']
rmalization)

activation_10 (Activation)     (None, 28, 28, 128)  0          ['conv3_1_1x1_reduce/
↪bn[0][0]']

conv3_1_3x3 (Conv2D)           (None, 28, 28, 128)  147456     ['activation_10[0][0]']

conv3_1_3x3/bn (BatchNormaliza  (None, 28, 28, 128)  512       ['conv3_1_3x3[0][0]']
tion)

activation_11 (Activation)     (None, 28, 28, 128)  0          ['conv3_1_3x3/bn[0][0]']

conv3_1_1x1_increase (Conv2D)  (None, 28, 28, 512)  65536      ['activation_11[0][0]']

conv3_1_1x1_proj (Conv2D)      (None, 28, 28, 512)  131072     ['activation_9[0][0]']

conv3_1_1x1_increase/bn (Batch  (None, 28, 28, 512)  2048      ['conv3_1_1x1_
↪increase[0][0]']
Normalization)

conv3_1_1x1_proj/bn (BatchNorm  (None, 28, 28, 512)  2048      ['conv3_1_1x1_proj[0][0]
↪']
alization)

add_3 (Add)                    (None, 28, 28, 512)  0          ['conv3_1_1x1_increase/
↪bn[0][0]',
                                                                 'conv3_1_1x1_proj/
↪bn[0][0]']

activation_12 (Activation)     (None, 28, 28, 512)  0          ['add_3[0][0]']

conv3_2_1x1_reduce (Conv2D)    (None, 28, 28, 128)  65536      ['activation_12[0][0]']

conv3_2_1x1_reduce/bn (BatchNo  (None, 28, 28, 128)  512       ['conv3_2_1x1_
↪reduce[0][0]']
rmalization)

activation_13 (Activation)     (None, 28, 28, 128)  0          ['conv3_2_1x1_reduce/
```

(continues on next page)

(continued from previous page)

```
↪bn[0][0]']

conv3_2_3x3 (Conv2D)           (None, 28, 28, 128)  147456      ['activation_13[0][0]']

conv3_2_3x3/bn (BatchNormaliza  (None, 28, 28, 128)  512        ['conv3_2_3x3[0][0]']
tion)

activation_14 (Activation)     (None, 28, 28, 128)  0           ['conv3_2_3x3/bn[0][0]']

conv3_2_1x1_increase (Conv2D)  (None, 28, 28, 512)  65536       ['activation_14[0][0]']

conv3_2_1x1_increase/bn (Batch  (None, 28, 28, 512)  2048       ['conv3_2_1x1_
↪increase[0][0]']
Normalization)

add_4 (Add)                    (None, 28, 28, 512)  0           ['conv3_2_1x1_increase/
↪bn[0][0]',
                                                                 'activation_12[0][0]']

activation_15 (Activation)     (None, 28, 28, 512)  0           ['add_4[0][0]']

conv3_3_1x1_reduce (Conv2D)    (None, 28, 28, 128)  65536       ['activation_15[0][0]']

conv3_3_1x1_reduce/bn (BatchNo  (None, 28, 28, 128)  512       ['conv3_3_1x1_
↪reduce[0][0]']
rmalization)

activation_16 (Activation)     (None, 28, 28, 128)  0           ['conv3_3_1x1_reduce/
↪bn[0][0]']

conv3_3_3x3 (Conv2D)           (None, 28, 28, 128)  147456      ['activation_16[0][0]']

conv3_3_3x3/bn (BatchNormaliza  (None, 28, 28, 128)  512       ['conv3_3_3x3[0][0]']
tion)

activation_17 (Activation)     (None, 28, 28, 128)  0           ['conv3_3_3x3/bn[0][0]']

conv3_3_1x1_increase (Conv2D)  (None, 28, 28, 512)  65536       ['activation_17[0][0]']

conv3_3_1x1_increase/bn (Batch  (None, 28, 28, 512)  2048       ['conv3_3_1x1_
↪increase[0][0]']
Normalization)

add_5 (Add)                    (None, 28, 28, 512)  0           ['conv3_3_1x1_increase/
↪bn[0][0]',
                                                                 'activation_15[0][0]']

activation_18 (Activation)     (None, 28, 28, 512)  0           ['add_5[0][0]']

conv3_4_1x1_reduce (Conv2D)    (None, 28, 28, 128)  65536       ['activation_18[0][0]']

conv3_4_1x1_reduce/bn (BatchNo  (None, 28, 28, 128)  512       ['conv3_4_1x1_
```

(continues on next page)

(continued from previous page)

```
↪reduce[0][0]']
rmalization)

activation_19 (Activation)     (None, 28, 28, 128)  0           ['conv3_4_1x1_reduce/
↪bn[0][0]']

conv3_4_3x3 (Conv2D)           (None, 28, 28, 128)  147456      ['activation_19[0][0]']

conv3_4_3x3/bn (BatchNormaliza  (None, 28, 28, 128)  512        ['conv3_4_3x3[0][0]']
tion)

activation_20 (Activation)     (None, 28, 28, 128)  0           ['conv3_4_3x3/bn[0][0]']

conv3_4_1x1_increase (Conv2D)  (None, 28, 28, 512)  65536       ['activation_20[0][0]']

conv3_4_1x1_increase/bn (Batch  (None, 28, 28, 512)  2048       ['conv3_4_1x1_
↪increase[0][0]']
Normalization)

add_6 (Add)                    (None, 28, 28, 512)  0           ['conv3_4_1x1_increase/
↪bn[0][0]',
                                                                 'activation_18[0][0]']

activation_21 (Activation)     (None, 28, 28, 512)  0           ['add_6[0][0]']

conv4_1_1x1_reduce (Conv2D)    (None, 14, 14, 256)  131072      ['activation_21[0][0]']

conv4_1_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024       ['conv4_1_1x1_
↪reduce[0][0]']
rmalization)

activation_22 (Activation)     (None, 14, 14, 256)  0           ['conv4_1_1x1_reduce/
↪bn[0][0]']

conv4_1_3x3 (Conv2D)           (None, 14, 14, 256)  589824      ['activation_22[0][0]']

conv4_1_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_1_3x3[0][0]']
tion)

activation_23 (Activation)     (None, 14, 14, 256)  0           ['conv4_1_3x3/bn[0][0]']

conv4_1_1x1_increase (Conv2D)  (None, 14, 14, 1024  262144      ['activation_23[0][0]']
                               )

conv4_1_1x1_proj (Conv2D)      (None, 14, 14, 1024  524288      ['activation_21[0][0]']
                               )

conv4_1_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_1_1x1_
↪increase[0][0]']
Normalization)                 )

conv4_1_1x1_proj/bn (BatchNorm  (None, 14, 14, 1024  4096       ['conv4_1_1x1_proj[0][0]
```

(continues on next page)

(continued from previous page)

```
↪']
alization)                      )

add_7 (Add)                     (None, 14, 14, 1024  0          ['conv4_1_1x1_increase/
↪bn[0][0]',
                                )                                'conv4_1_1x1_proj/
↪bn[0][0]']

activation_24 (Activation)      (None, 14, 14, 1024  0          ['add_7[0][0]']
                                )

conv4_2_1x1_reduce (Conv2D)     (None, 14, 14, 256)  262144     ['activation_24[0][0]']

conv4_2_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024       ['conv4_2_1x1_
↪reduce[0][0]']
rmalization)

activation_25 (Activation)      (None, 14, 14, 256)  0          ['conv4_2_1x1_reduce/
↪bn[0][0]']

conv4_2_3x3 (Conv2D)            (None, 14, 14, 256)  589824     ['activation_25[0][0]']

conv4_2_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_2_3x3[0][0]']
tion)

activation_26 (Activation)      (None, 14, 14, 256)  0          ['conv4_2_3x3/bn[0][0]']

conv4_2_1x1_increase (Conv2D)   (None, 14, 14, 1024  262144     ['activation_26[0][0]']
                                )

conv4_2_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_2_1x1_
↪increase[0][0]']
Normalization)                  )

add_8 (Add)                     (None, 14, 14, 1024  0          ['conv4_2_1x1_increase/
↪bn[0][0]',
                                )                                'activation_24[0][0]']

activation_27 (Activation)      (None, 14, 14, 1024  0          ['add_8[0][0]']
                                )

conv4_3_1x1_reduce (Conv2D)     (None, 14, 14, 256)  262144     ['activation_27[0][0]']

conv4_3_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024       ['conv4_3_1x1_
↪reduce[0][0]']
rmalization)

activation_28 (Activation)      (None, 14, 14, 256)  0          ['conv4_3_1x1_reduce/
↪bn[0][0]']

conv4_3_3x3 (Conv2D)            (None, 14, 14, 256)  589824     ['activation_28[0][0]']
```

(continues on next page)

(continued from previous page)

```
conv4_3_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_3_3x3[0][0]']
tion)

activation_29 (Activation)     (None, 14, 14, 256)  0          ['conv4_3_3x3/bn[0][0]']

conv4_3_1x1_increase (Conv2D)  (None, 14, 14, 1024  262144     ['activation_29[0][0]']
                               )

conv4_3_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_3_1x1_
↪increase[0][0]']
Normalization)                 )

add_9 (Add)                    (None, 14, 14, 1024  0          ['conv4_3_1x1_increase/
↪bn[0][0]',
                               )                                 'activation_27[0][0]']

activation_30 (Activation)     (None, 14, 14, 1024  0          ['add_9[0][0]']
                               )

conv4_4_1x1_reduce (Conv2D)    (None, 14, 14, 256)  262144     ['activation_30[0][0]']

conv4_4_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024       ['conv4_4_1x1_
↪reduce[0][0]']
rmalization)

activation_31 (Activation)     (None, 14, 14, 256)  0          ['conv4_4_1x1_reduce/
↪bn[0][0]']

conv4_4_3x3 (Conv2D)           (None, 14, 14, 256)  589824     ['activation_31[0][0]']

conv4_4_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_4_3x3[0][0]']
tion)

activation_32 (Activation)     (None, 14, 14, 256)  0          ['conv4_4_3x3/bn[0][0]']

conv4_4_1x1_increase (Conv2D)  (None, 14, 14, 1024  262144     ['activation_32[0][0]']
                               )

conv4_4_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_4_1x1_
↪increase[0][0]']
Normalization)                 )

add_10 (Add)                   (None, 14, 14, 1024  0          ['conv4_4_1x1_increase/
↪bn[0][0]',
                               )                                 'activation_30[0][0]']

activation_33 (Activation)     (None, 14, 14, 1024  0          ['add_10[0][0]']
                               )

conv4_5_1x1_reduce (Conv2D)    (None, 14, 14, 256)  262144     ['activation_33[0][0]']

conv4_5_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024       ['conv4_5_1x1_
```

(continues on next page)

(continued from previous page)

```
↪reduce[0][0]']
rmalization)

activation_34 (Activation)     (None, 14, 14, 256)  0           ['conv4_5_1x1_reduce/
↪bn[0][0]']

conv4_5_3x3 (Conv2D)           (None, 14, 14, 256)  589824      ['activation_34[0][0]']

conv4_5_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_5_3x3[0][0]']
tion)

activation_35 (Activation)     (None, 14, 14, 256)  0           ['conv4_5_3x3/bn[0][0]']

conv4_5_1x1_increase (Conv2D)  (None, 14, 14, 1024  262144      ['activation_35[0][0]']
                               )

conv4_5_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_5_1x1_
↪increase[0][0]']
Normalization)                 )

add_11 (Add)                   (None, 14, 14, 1024  0           ['conv4_5_1x1_increase/
↪bn[0][0]',
                               )                                 'activation_33[0][0]']

activation_36 (Activation)     (None, 14, 14, 1024  0           ['add_11[0][0]']
                               )

conv4_6_1x1_reduce (Conv2D)    (None, 14, 14, 256)  262144      ['activation_36[0][0]']

conv4_6_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024       ['conv4_6_1x1_
↪reduce[0][0]']
rmalization)

activation_37 (Activation)     (None, 14, 14, 256)  0           ['conv4_6_1x1_reduce/
↪bn[0][0]']

conv4_6_3x3 (Conv2D)           (None, 14, 14, 256)  589824      ['activation_37[0][0]']

conv4_6_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_6_3x3[0][0]']
tion)

activation_38 (Activation)     (None, 14, 14, 256)  0           ['conv4_6_3x3/bn[0][0]']

conv4_6_1x1_increase (Conv2D)  (None, 14, 14, 1024  262144      ['activation_38[0][0]']
                               )

conv4_6_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_6_1x1_
↪increase[0][0]']
Normalization)                 )

add_12 (Add)                   (None, 14, 14, 1024  0           ['conv4_6_1x1_increase/
↪bn[0][0]',
```

(continues on next page)

(continued from previous page)

```
                                )                                 'activation_36[0][0]']

 activation_39 (Activation)     (None, 14, 14, 1024  0           ['add_12[0][0]']
                                )

 conv5_1_1x1_reduce (Conv2D)    (None, 7, 7, 512)    524288      ['activation_39[0][0]']

 conv5_1_1x1_reduce/bn (BatchNo  (None, 7, 7, 512)   2048        ['conv5_1_1x1_
↪reduce[0][0]']
 rmalization)

 activation_40 (Activation)     (None, 7, 7, 512)    0           ['conv5_1_1x1_reduce/
↪bn[0][0]']

 conv5_1_3x3 (Conv2D)           (None, 7, 7, 512)    2359296     ['activation_40[0][0]']

 conv5_1_3x3/bn (BatchNormaliza  (None, 7, 7, 512)   2048        ['conv5_1_3x3[0][0]']
 tion)

 activation_41 (Activation)     (None, 7, 7, 512)    0           ['conv5_1_3x3/bn[0][0]']

 conv5_1_1x1_increase (Conv2D)  (None, 7, 7, 2048)   1048576     ['activation_41[0][0]']

 conv5_1_1x1_proj (Conv2D)      (None, 7, 7, 2048)   2097152     ['activation_39[0][0]']

 conv5_1_1x1_increase/bn (Batch  (None, 7, 7, 2048)  8192        ['conv5_1_1x1_
↪increase[0][0]']
 Normalization)

 conv5_1_1x1_proj/bn (BatchNorm  (None, 7, 7, 2048)  8192        ['conv5_1_1x1_proj[0][0]
↪']
 alization)

 add_13 (Add)                   (None, 7, 7, 2048)   0           ['conv5_1_1x1_increase/
↪bn[0][0]',
                                                                  'conv5_1_1x1_proj/
↪bn[0][0]']

 activation_42 (Activation)     (None, 7, 7, 2048)   0           ['add_13[0][0]']

 conv5_2_1x1_reduce (Conv2D)    (None, 7, 7, 512)    1048576     ['activation_42[0][0]']

 conv5_2_1x1_reduce/bn (BatchNo  (None, 7, 7, 512)   2048        ['conv5_2_1x1_
↪reduce[0][0]']
 rmalization)

 activation_43 (Activation)     (None, 7, 7, 512)    0           ['conv5_2_1x1_reduce/
↪bn[0][0]']

 conv5_2_3x3 (Conv2D)           (None, 7, 7, 512)    2359296     ['activation_43[0][0]']

 conv5_2_3x3/bn (BatchNormaliza  (None, 7, 7, 512)   2048        ['conv5_2_3x3[0][0]']
```

(continues on next page)

(continued from previous page)

```
tion)

activation_44 (Activation)     (None, 7, 7, 512)    0           ['conv5_2_3x3/bn[0][0]']

conv5_2_1x1_increase (Conv2D)  (None, 7, 7, 2048)   1048576     ['activation_44[0][0]']

conv5_2_1x1_increase/bn (Batch  (None, 7, 7, 2048)  8192        ['conv5_2_1x1_
↪increase[0][0]']
Normalization)

add_14 (Add)                   (None, 7, 7, 2048)   0           ['conv5_2_1x1_increase/
↪bn[0][0]',
                                                                 'activation_42[0][0]']

activation_45 (Activation)     (None, 7, 7, 2048)   0           ['add_14[0][0]']

conv5_3_1x1_reduce (Conv2D)    (None, 7, 7, 512)    1048576     ['activation_45[0][0]']

conv5_3_1x1_reduce/bn (BatchNo  (None, 7, 7, 512)   2048        ['conv5_3_1x1_
↪reduce[0][0]']
rmalization)

activation_46 (Activation)     (None, 7, 7, 512)    0           ['conv5_3_1x1_reduce/
↪bn[0][0]']

conv5_3_3x3 (Conv2D)           (None, 7, 7, 512)    2359296     ['activation_46[0][0]']

conv5_3_3x3/bn (BatchNormaliza  (None, 7, 7, 512)   2048        ['conv5_3_3x3[0][0]']
tion)

activation_47 (Activation)     (None, 7, 7, 512)    0           ['conv5_3_3x3/bn[0][0]']

conv5_3_1x1_increase (Conv2D)  (None, 7, 7, 2048)   1048576     ['activation_47[0][0]']

conv5_3_1x1_increase/bn (Batch  (None, 7, 7, 2048)  8192        ['conv5_3_1x1_
↪increase[0][0]']
Normalization)

add_15 (Add)                   (None, 7, 7, 2048)   0           ['conv5_3_1x1_increase/
↪bn[0][0]',
                                                                 'activation_45[0][0]']

activation_48 (Activation)     (None, 7, 7, 2048)   0           ['add_15[0][0]']

avg_pool (AveragePooling2D)    (None, 1, 1, 2048)   0           ['activation_48[0][0]']

global_average_pooling2d (Glob  (None, 2048)        0           ['avg_pool[0][0]']
alAveragePooling2D)

gaussian_noise (GaussianNoise)  (None, 2048)        0           ['global_average_
↪pooling2d[0][0]'
                                                                ]
```

(continues on next page)

(continued from previous page)

```
 dense_x (Dense)                (None, 512)          1049088     ['gaussian_noise[0][0]']

==================================================================================================
Total params: 24,610,240
Trainable params: 24,557,120
Non-trainable params: 53,120
--------------------------------------------------------------------------------------------------
↪---------
```

Formation of the neural network architecture of the model and downloading its weights to obtain features / scores based on deep features (video modality)

- `_b5s.video_model_nn_` - Neural network model **tf.keras.Model** for obtaining features / scores by deep features

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-10 17:12:11] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a5 License: BSD License

Formation of the neural network architecture of the model (FI V2)

[4]:
```
res_load_video_model_nn = _b5.load_video_model_nn(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2022-12-11 14:41:16] Formation of a neural network architecture for obtaining scores by deep features (video modality) ...

— Runtime: 1.559 sec. —

Downloading the weights of the neural network model

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['video']['fi']['nn']['sberdisk']

res_load_video_model_weights_nn = _b5.load_video_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 17:12:13] Downloading the weights of the neural network model to obtain scores by deep features (video modality) ...

[2023-12-10 17:12:14] File download "weights_2022-03-22_16-31-48.h5" (100.0%) ...

— Runtime: 1.053 sec. —

Displaying the formed neural network architecture of the model

[6]:
```
_b5.video_model_nn_.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 10, 512)]         0

 lstm_1024_v_nn (LSTM)       (None, 1024)              6295552

 dropout (Dropout)           (None, 1024)              0

 dense (Dense)               (None, 5)                 5125
```

(continues on next page)

(continued from previous page)

```
 activation (Activation)     (None, 5)                 0

=================================================================
Total params: 6,300,677
Trainable params: 6,300,677
Non-trainable params: 0
_________________________________________________________________
```

Formation of neural network architectures of models and downloading their weights to obtain the personality traits scores (video modality)

- `_b5.video_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

```
[2023-12-14 21:04:19] OCEANAI - personaly traits:  Authors:  Elena Ryumina
[ryumina_ev@mail.ru]  Dmitry Ryumin [dl_03.03.1991@mail.ru]  Alexey Karpov
[karpov@iias.spb.su]  Maintainers:  Elena Ryumina [ryumina_ev@mail.ru]  Dmitry Ryumin
[dl_03.03.1991@mail.ru]  Version: 1.0.0a16  License: BSD License
```

Formation of neural network architectures of models (FI V2)

[4]:
```
res_load_video_models_b5 = _b5.load_video_models_b5(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 21:04:19] Formation of neural network architectures of models for obtaining the personality traits scores (video modality) ...

— Runtime: 0.094 sec. —

Downloading weights of neural network models

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url_openness = _b5.weights_for_big5_['video']['fi']['b5']['openness']['sberdisk']
url_conscientiousness = _b5.weights_for_big5_['video']['fi']['b5']['conscientiousness'][
↪'sberdisk']
url_extraversion = _b5.weights_for_big5_['video']['fi']['b5']['extraversion']['sberdisk']
url_agreeableness = _b5.weights_for_big5_['video']['fi']['b5']['agreeableness']['sberdisk
↪']
url_non_neuroticism = _b5.weights_for_big5_['video']['fi']['b5']['non_neuroticism'][
↪'sberdisk']

res_load_video_models_weights_b5 = _b5.load_video_models_weights_b5(
    url_openness = url_openness, # Openness
    url_conscientiousness = url_conscientiousness, # Conscientiousness
    url_extraversion = url_extraversion, # Extraversion
    url_agreeableness = url_agreeableness, # Agreeableness
    url_non_neuroticism = url_non_neuroticism, # Non-Neuroticism
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 21:04:19] Downloading the weights of neural network models to obtain the personality traits scores (video modality) ...

[2023-12-14 21:04:19] File download "weights_2022-06-15_16-46-30.h5" (100.0%) ... Openness

[2023-12-14 21:04:20] File download "weights_2022-06-15_16-48-50.h5" (100.0%) ... Conscientiousness

[2023-12-14 21:04:20] File download "weights_2022-06-15_16-54-06.h5" (100.0%) ... Extraversion

[2023-12-14 21:04:20] File download "weights_2022-06-15_17-02-03.h5" (100.0%) ... Agreeableness

[2023-12-14 21:04:20] File download "weights_2022-06-15_17-06-15.h5" (100.0%) ... Non-Neuroticism

— Runtime: 0.998 sec. —

Displaying the formed neural network architecture of the model

- Openness
- Conscientiousness
- Extraversion
- Agreeableness
- Non-neuroticism

[6]:
```
_b5.video_models_b5_['openness'].summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32)]              0

 dense_1 (Dense)             (None, 1)                 33

 activ_1 (Activation)        (None, 1)                 0

=================================================================
Total params: 33 (132.00 Byte)
Trainable params: 33 (132.00 Byte)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Extracting features from a visual signal

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

```
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
```

(continues on next page)

(continued from previous page)

```
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2024-03-28 21:50:44] OCEANAI - персональные качества личности человека: Авторы: Рюмина Елена [ryumina_ev@mail.ru] Рюмин Дмитрий [dl_03.03.1991@mail.ru] Карпов Алексей [karpov@iias.spb.su] Сопровождающие: Рюмина Елена [ryumina_ev@mail.ru] Рюмин Дмитрий [dl_03.03.1991@mail.ru] Версия: 1.0.0a22 Лицензия: BSD License

Formation of the neural network architecture of the model

[4]:
```
res_load_video_model_deep_fe = _b5.load_video_model_deep_fe(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2024-03-28 21:50:46] Формирование нейросетевой архитектуры для получения нейросетевых признаков (видео модальность) ...

— Время выполнения: 1.001 сек. —

Downloading the weights of the neural network model

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['video']['fi']['fe']['sberdisk']

res_load_video_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2024-03-28 21:50:50] Загрузка весов нейросетевой модели для получения нейросетевых признаков (видео модальность) ...

[2024-03-28 21:50:56] Загрузка файла “weights_2022-11-01_12-27-07.h5” 100.0% ...

— Время выполнения: 6.461 сек. —

Process of extracting visual features (FI V2)

[6]:
```
# Core settings
# Path to video file
path = 'video_FI/test/_plk5k7PBEg.003.mp4'

hc_features, nn_features = _b5.get_visual_features(
    path = path, # Path to video file
    reduction_fps = 5, # Frame rate reduction
    window = 10, # Signal segment window size (in frames)
    step = 5, # Signal segment window shift step (frames)
    lang = 'en', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2024-03-28 21:50:58] Извлечение признаков (экспертных и нейросетевых) из визуального сигнала ...

[2024-03-28 21:51:22] Статистика извлеченных признаков из визуального сигнала: Общее количество сегментов с: 1. экспертными признаками: 16 2. нейросетевыми признаками: 16 Размерность матрицы экспертных признаков одного сегмента: 10 × 115 Размерность матрицы с нейросетевыми признаками одного сегмента: 10 × 512 Понижение кадровой частоты: с 30 до 5

— Время выполнения: 23.465 сек. —

Process of extracting visual features (MuPTA)

[7]:
```
# Core settings
# Path to video file
path = 'video_FI/test/_plk5k7PBEg.003.mp4'

hc_features, nn_features = _b5.get_visual_features(
    path = path, # Path to video file
    reduction_fps = 5, # Frame rate reduction
    window = 10, # Signal segment window size (in frames)
    step = 5, # Signal segment window shift step (frames)
    lang = 'ru', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2024-03-28 21:51:25] Извлечение признаков (экспертных и нейросетевых) из визуального сигнала ...

[2024-03-28 21:51:43] Статистика извлеченных признаков из визуального сигнала: Общее количество сегментов с: 1. экспертными признаками: 16 2. нейросетевыми признаками: 16 Размерность матрицы экспертных признаков одного сегмента: 10 × 109 Размерность матрицы

(continues on next page)

(continued from previous page)

с нейросетевыми признаками одного сегмента: 10 × 512 Понижение кадровой частоты: с 30 до 5
— Время выполнения: 18.659 сек. —

[ ]:

Getting video scores

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Interface language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-14 21:05:26] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a16 License: BSD License

Getting and displaying versions of installed libraries

- `_b5.df_pkgs_` - DataFrame with versions of installed libraries

[4]:
```
_b5.libs_vers(runtime = True, run = True)
```

	Package	Version
1	TensorFlow	2.15.0
2	Keras	2.15.0
3	OpenCV	4.8.1
4	MediaPipe	0.9.0
5	NumPy	1.26.2

(continues on next page)

(continued from previous page)

6	SciPy	1.11.4
7	Pandas	2.1.3
8	Scikit-learn	1.3.2
9	OpenSmile	2.5.0
10	Librosa	0.10.1
11	AudioRead	3.0.1
12	IPython	8.18.1
13	PyMediaInfo	6.1.0
14	Requests	2.31.0
15	JupyterLab	4.0.9
16	LIWC	0.5.0
17	Transformers	4.36.0
18	Sentencepiece	0.1.99
19	Torch	2.0.1+cpu
20	Torchaudio	2.0.2+cpu

— Runtime: 0.005 sec. —

Formation of the neural network architecture of the model for obtaining scores by hand-crafted features

- `_b5.video_model_hc_` - Neural network model **tf.keras.Model** for obtaining scores by hand-crafted features

[5]:
```
res_load_video_model_hc = _b5.load_video_model_hc(
    lang = 'en', # Language selection for models trained on First Impressions V2'en' and␣
↪models trained on for MuPTA 'ru'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 21:05:26] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (video modality) ...

— Runtime: 0.321 sec. —

Downloading the weights of the neural network model to obtain scores by hand-crafted features

- `_b5.video_model_hc_` - Neural network model **tf.keras.Model** for obtaining scores by hand-crafted features

[6]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in one step

url = _b5.weights_for_big5_['video']['fi']['hc']['sberdisk']

res_load_video_model_weights_hc = _b5.load_video_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
```

(continues on next page)

(continued from previous page)

```
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 21:05:27] Downloading the weights of the neural network model to obtain scores by hand-crafted features (video modality) ...

[2023-12-14 21:05:27] File download "weights_2022-08-27_18-53-35.h5" (100.0%) ...

— Runtime: 0.249 sec. —

Formation of neural network architecture for obtaining neural network features

- `_b5.video_model_deep_fe_` - Neural network model **tf.keras.Model** for obtaining deep features

[7]:
```
res_load_video_model_deep_fe = _b5.load_video_model_deep_fe(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 21:05:27] Formation of neural network architecture for obtaining deep features (video modality) ...

— Runtime: 0.823 sec. —

Downloading weights of a neural network model to obtain neural network features

- `_b5.video_model_deep_fe_` - Neural network model **tf.keras.Model** for obtaining deep features

[8]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in one step

url = _b5.weights_for_big5_['video']['fi']['fe']['sberdisk']

res_load_video_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 21:05:28] Downloading weights of a neural network model to obtain deep features (video modality) ...

[2023-12-14 21:05:31] File download "weights_2022-11-01_12-27-07.h5" (100.0%) ...

— Runtime: 3.342 sec. —

Formation of the neural network architecture of the model for obtaining scores by deep features

- `_b5.video_model_nn_` - Neural network model **tf.keras.Model** for obtaining scores by deep features

[9]:
```
res_load_video_model_nn = _b5.load_video_model_nn(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 21:05:31] Formation of a neural network architecture for obtaining scores by deep features (video modality) ...

— Runtime: 0.708 sec. —

Downloading the weights of the neural network model to obtain scores for deep features

- `_b5.video_model_nn_` - Neural network model **tf.keras.Model** for obtaining scores by deep features

[10]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in one step

url = _b5.weights_for_big5_['video']['fi']['nn']['sberdisk']

res_load_video_model_weights_nn = _b5.load_video_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 21:05:32] Downloading the weights of the neural network model to obtain scores for deep features (video modality) ...

[2023-12-14 21:05:32] File downloading “weights_2022-03-22_16-31-48.h5”

— Runtime: 0.209 sec. —

Formation of neural network architectures of models for obtaining the personality traits scores

- `_b5.video_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

[11]:
```
res_load_video_models_b5 = _b5.load_video_models_b5(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 21:05:32] Formation of neural network architectures of models for obtaining the personality traits acores (video modality) ...

— Runtime: 0.046 sec. —

Downloading the weights of neural network models to obtain the personality traits scores

- `_b5.video_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

[12]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in one step

url_openness = _b5.weights_for_big5_['video']['fi']['b5']['openness']['sberdisk']
url_conscientiousness = _b5.weights_for_big5_['video']['fi']['b5']['conscientiousness'][
↪'sberdisk']
url_extraversion = _b5.weights_for_big5_['video']['fi']['b5']['extraversion']['sberdisk']
url_agreeableness = _b5.weights_for_big5_['video']['fi']['b5']['agreeableness']['sberdisk
↪']
url_non_neuroticism = _b5.weights_for_big5_['video']['fi']['b5']['non_neuroticism'][
↪'sberdisk']

res_load_video_models_weights_b5 = _b5.load_video_models_weights_b5(
    url_openness = url_openness, # Openness
    url_conscientiousness = url_conscientiousness, # Conscientiousness
    url_extraversion = url_extraversion, # Extraversion
    url_agreeableness = url_agreeableness, # Agreeableness
    url_non_neuroticism = url_non_neuroticism, # Non-Neuroticism
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 21:05:32] Downloading the weights of neural network models to obtain the personality traits scores (video modality) ...

[2023-12-14 21:05:32] File download "weights_2022-06-15_16-46-30.h5" Openness

[2023-12-14 21:05:32] File download "weights_2022-06-15_16-48-50.h5" Conscientiousness

[2023-12-14 21:05:33] File download "weights_2022-06-15_16-54-06.h5" Extraversion
[2023-12-14 21:05:33] File download "weights_2022-06-15_17-02-03.h5" Agreeableness
[2023-12-14 21:05:33] File download "weights_2022-06-15_17-06-15.h5" Non-Neuroticism
— Runtime: 1.013 sec. —

Getting scores (video modality)

- `_b5.df_files_` - DataFrame with data
- `_b5.df_accuracy_` - DataFrame with accuracy

[13]:
```
# Core settings
_b5.path_to_dataset_ = '/Users/dl/GitHub/oceanai/oceanai/dataset/First_Impression' #␣
↪Dataset directory
# Directories not included in the selection
_b5.ignore_dirs_ = []
# Key names for DataFrame dataset
_b5.keys_dataset_ = ['Path', 'Openness', 'Conscientiousness', 'Extraversion',
↪'Agreeableness', 'Non-Neuroticism']
_b5.ext_ = ['.mp4'] # Search file extensions
_b5.path_to_logs_ = './logs' # Directory for saving LOG files

# Full path to the file containing the ground truth scores for the accuracy calculation

url_accuracy = _b5.true_traits_['fi']['sberdisk']

res_get_video_union_predictions = _b5.get_video_union_predictions(
    depth = 2,          # Hierarchy depth for receiving audio and video data
    recursive = False, # Recursive data search
    reduction_fps = 5, # Frame rate reduction
    window = 10,        # PSignal segment window size (in seconds)
    step = 5,           # Signal segment window shift step (in seconds)
    lang = 'en', # Language selection for models trained on First Impressions V2'en' and␣
↪models trained on for MuPTA 'ru'
    accuracy = True,    # Accuracy calculation
    url_accuracy = url_accuracy,
    logs = True,        # If necessary, generate a LOG file
    out = True,         # Display
    runtime = True,     # Runtime count
    run = True          # Run blocking
)
```

[2023-12-14 22:24:55] Getting scores and accuracy calculation (video modality) ...

2000 from 2000 (100.0%) ... test80_25_Q4wOgixh7E.004.mp4 ...

```
                                                Path  Openness  \
ID
1   E:\Databases\FirstImpressionsV2\test\test80_01...  0.526971
2   E:\Databases\FirstImpressionsV2\test\test80_01...  0.559385
3   E:\Databases\FirstImpressionsV2\test\test80_01...  0.466969
4   E:\Databases\FirstImpressionsV2\test\test80_01...  0.626113
```

(continues on next page)

(continued from previous page)

```
5   E:\Databases\FirstImpressionsV2\test\test80_01...    0.5925
6   E:\Databases\FirstImpressionsV2\test\test80_01...  0.671855
7   E:\Databases\FirstImpressionsV2\test\test80_01...  0.411555
8   E:\Databases\FirstImpressionsV2\test\test80_01...  0.583696
9   E:\Databases\FirstImpressionsV2\test\test80_01...  0.551353
10  E:\Databases\FirstImpressionsV2\test\test80_01...  0.575084
11  E:\Databases\FirstImpressionsV2\test\test80_01...  0.559182
12  E:\Databases\FirstImpressionsV2\test\test80_01...   0.50948
13  E:\Databases\FirstImpressionsV2\test\test80_01...  0.330026
14  E:\Databases\FirstImpressionsV2\test\test80_01...  0.649351
15  E:\Databases\FirstImpressionsV2\test\test80_01...  0.651914
16  E:\Databases\FirstImpressionsV2\test\test80_01...  0.523986
17  E:\Databases\FirstImpressionsV2\test\test80_01...  0.575113
18  E:\Databases\FirstImpressionsV2\test\test80_01...  0.566349
19  E:\Databases\FirstImpressionsV2\test\test80_01...  0.672282
20  E:\Databases\FirstImpressionsV2\test\test80_01...  0.684442
21  E:\Databases\FirstImpressionsV2\test\test80_01...  0.550788
22  E:\Databases\FirstImpressionsV2\test\test80_01...  0.525446
23  E:\Databases\FirstImpressionsV2\test\test80_01...  0.473489
24  E:\Databases\FirstImpressionsV2\test\test80_01...  0.667829
25  E:\Databases\FirstImpressionsV2\test\test80_01...  0.469207
26  E:\Databases\FirstImpressionsV2\test\test80_01...  0.625514
27  E:\Databases\FirstImpressionsV2\test\test80_01...  0.568821
28  E:\Databases\FirstImpressionsV2\test\test80_01...  0.696397
29  E:\Databases\FirstImpressionsV2\test\test80_01...  0.578405
30  E:\Databases\FirstImpressionsV2\test\test80_01...  0.637576

   Conscientiousness Extraversion Agreeableness Non-Neuroticism
ID
1           0.460063     0.422793      0.502726        0.450519
2           0.432843     0.504231      0.578673        0.513424
3            0.51701     0.331863      0.451395        0.406188
4           0.597363     0.564068      0.574056        0.589245
5           0.507246     0.505394      0.585405        0.493066
6           0.642559     0.614689      0.613508        0.619511
7           0.394029     0.329323      0.488684         0.39105
8           0.568682     0.505574      0.625314        0.587337
9           0.450333     0.449763      0.495501        0.438009
10          0.517972      0.46315      0.582468        0.537961
11          0.398618     0.433806      0.480592        0.492383
12          0.432549       0.3319      0.495221        0.486891
13          0.322635     0.235595      0.369766         0.25056
14          0.550074     0.502858      0.526621        0.566755
15           0.83048     0.535514      0.695223        0.734383
16          0.435594     0.382946       0.41001        0.466265
17          0.678301     0.468646      0.602139        0.626021
18          0.558975     0.462116      0.606252        0.569516
19            0.6552     0.656699      0.627328        0.663199
20          0.602593     0.680469      0.635343        0.652304
21          0.492015     0.404885      0.562745        0.478233
22          0.469039     0.428517      0.491442         0.45359
23          0.442729     0.353017      0.447929        0.358706
```

(continues on next page)

(continued from previous page)

```
24          0.655159     0.603695      0.630121        0.614812
25          0.594029     0.364701      0.522734        0.481228
26          0.641622     0.514204      0.547718         0.54766
27          0.524382     0.475687      0.520644        0.531275
28          0.665074      0.70902      0.655993        0.689747
29          0.577321     0.487293      0.557221         0.52153
30          0.587702     0.614512      0.637398        0.613861
```

[2023-12-14 22:24:55] Trait-wise accuracy ...

```
          Openness  Conscientiousness  Extraversion  Agreeableness  \
Metrics
MAE         0.0873              0.082        0.0805          0.087
Accuracy    0.9127              0.918        0.9195          0.913

          Non-Neuroticism    Mean
Metrics
MAE                0.0872  0.0848
Accuracy           0.9128  0.9152
```

[2023-12-14 22:24:55] Mean absolute error: 0.0848, Accuracy: 0.9152 ...

Log files saved successfully ...

— Runtime: 4762.254 sec. —

Text information processing

Formation of the neural network architecture of the model and downloading its weights to obtain features / scores based on hand-crafted features (text modality)

- `_b5.text_model_hc_` - Neural network model **tf.keras.Model** for obtaining features / scores by hand-crafted features

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
```

(continues on next page)

(continued from previous page)

```
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-10 17:11:13] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a5 License: BSD License

Formation of the neural network architecture of the model (FI V2

[4]:
```
res_load_text_model_hc_fi = _b5.load_text_model_hc(
    corpus = "fi", # Corpus selection for models trained on First Impressions V2 'fi' and␣
↪models trained on for MuPTA 'mupta'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blockin
)
```

[2023-12-10 17:11:13] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (text modality) ...

— Runtime: 1.886 sec. —

Downloading the weights of the neural network model

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['text']['fi']['hc']['sberdisk']

res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,      # Display
    runtime = True, # Runtime count
    run = True       # Run blocking
)
```

[2023-12-10 16:54:00] Downloading the weights of a neural network model to obtain hand-crafted features (text modality) ...

[2023-12-10 16:54:01] File download "weights_2023-07-15_10-52-15.h5" 100.0% ...

— Runtime: 0.311 sec. —

Displaying the formed neural network architecture of the model

[6]:
```
_b5.text_model_hc_.summary()
```

```
Model: "model"
_________________________________________________________________________________________
 Layer (type)                    Output Shape         Param #     Connected to
=========================================================================================
 model_hc/input (InputLayer)     [(None, 89, 64)]     0           []

 model_hc/bilstm_1 (Bidirection  (None, 89, 64)       24832       ['model_hc/input[0][0]']
 al)

 model_hc/dence_2 (Dense)        (None, 89, 64)       4160        ['model_hc/input[0][0]']

 model_hc/attention (Attention)  (None, 89, 64)       0           ['model_hc/bilstm_
↪1[0][0]',
                                                                  'model_hc/bilstm_
↪1[0][0]']

 model_hc/bilstm_2 (Bidirection  (None, 89, 64)       24832       ['model_hc/dence_2[0][0]
↪']
 al)

 add (Add)                       (None, 89, 64)       0           ['model_hc/bilstm_
↪1[0][0]',
                                                                  'model_hc/
↪attention[0][0]',
                                                                  'model_hc/bilstm_
↪2[0][0]']

 model_hc/add (Addition)         (None, 128)          0           ['add[0][0]']

 dense (Dense)                   (None, 5)            645         ['model_hc/add[0][0]']

=========================================================================================
Total params: 54,469
Trainable params: 54,469
Non-trainable params: 0
_________________________________________________________________________________________
```

Formation of the neural network architecture of the model (MuPTA)

[7]:
```
res_load_text_model_hc_mupta = _b5.load_text_model_hc(
    corpus = "mupta", # Corpus selection for models trained on First Impressions V2 'fi'␣
↪and models trained on for MuPTA 'mupta'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 16:54:06] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (text modality) ...

— Runtime: 0.577 sec. —

Downloading the weights of the neural network model

[8]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['text']['mupta']['hc']['sberdisk']

res_load_text_model_weights_hc_mupta = _b5.load_text_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 16:54:19] Downloading the weights of a neural network model to obtain hand-crafted features (text modality) ...

[2023-12-10 16:54:19] File download "weights_2023-07-15_10-53-38.h5" 100.0% ...

— Runtime: 0.264 sec. —

Displaying the formed neural network architecture of the model

[9]:
```
_b5.text_model_hc_.summary()
```

```
Model: "model_1"
_________________________________________________________________________________________
↪_________
 Layer (type)                   Output Shape         Param #     Connected to
=========================================================================================
 model_hc/input (InputLayer)    [(None, 365, 64)]    0           []

 model_hc/bilstm_1 (Bidirection  (None, 365, 64)     24832       ['model_hc/input[0][0]']
 al)
```

(continues on next page)

(continued from previous page)

```
 model_hc/dence_2 (Dense)       (None, 365, 64)      4160        ['model_hc/input[0][0]']

 model_hc/attention (Attention)  (None, 365, 64)     0           ['model_hc/bilstm_
↪1[0][0]',
                                                                  'model_hc/bilstm_
↪1[0][0]']

 model_hc/bilstm_2 (Bidirection  (None, 365, 64)     24832       ['model_hc/dence_2[0][0]
↪']
 al)

 add_1 (Add)                    (None, 365, 64)      0           ['model_hc/bilstm_
↪1[0][0]',
                                                                  'model_hc/
↪attention[0][0]',
                                                                  'model_hc/bilstm_
↪2[0][0]']

 model_hc/add (Addition)        (None, 128)          0           ['add_1[0][0]']

 dense_1 (Dense)                (None, 5)            645         ['model_hc/add[0][0]']

==================================================================================================
Total params: 54,469
Trainable params: 54,469
Non-trainable params: 0
--------------------------------------------------------------------------------------------------
↪---------
```

Formation of the neural network architecture of the model and downloading its weights to obtain features / scores based on deep features (text modality)

- `_b5s.text_model_nn_` - Neural network model **tf.keras.Model** for obtaining features / scores by deep features

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

```
[2023-12-10 17:12:11] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a5 License: BSD License
```

Formation of the neural network architecture of the model (FI V2)

[4]:
```
res_load_text_model_nn_fi = _b5.load_text_model_nn(
    corpus = "fi", # Corpus selection for models trained on First Impressions V2 'fi' and␣
↪models trained on for MuPTA 'mupta'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

```
[2023-12-10 16:55:40] Formation of a neural network architecture for obtaining scores by deep features (text modality) ...
```

```
— Runtime: 1.03 sec. —
```

Downloading the weights of the neural network model

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['text']['fi']['nn']['sberdisk']

res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,      # Display
    runtime = True, # Runtime count
    run = True       # Run blocking
)
```

[2023-12-10 16:55:45] Downloading the weights of a neural network model to obtain deep features (text modality) ...

[2023-12-10 16:55:45] File download "weights_2023-07-03_15-01-08.h5" 100.0% ...

— Runtime: 0.393 sec. —

Displaying the formed neural network architecture of the model

[6]:
```
_b5.text_model_nn_.summary()
```

```
Model: "model"
_________________________________________________________________________________________
↪_________
 Layer (type)                   Output Shape         Param #     Connected to
=========================================================================================================
 model_nn/input (InputLayer)    [(None, 104, 768)]   0           []

 model_nn/bilstm_1 (Bidirection  (None, 104, 64)     205056      ['model_nn/input[0][0]']
 al)

 model_nn/attention (Attention)  (None, 104, 64)     0           ['model_nn/bilstm_
↪1[0][0]',
                                                                  'model_nn/bilstm_
↪1[0][0]']

 model_nn/dence_2 (Dense)       (None, 104, 128)     8320        ['model_nn/
↪attention[0][0]']

 model_nn/add (Addition)        (None, 256)          0           ['model_nn/dence_2[0][0]
↪']

 model_nn/dence_3 (Dense)       (None, 128)          32896       ['model_nn/add[0][0]']

 dense (Dense)                  (None, 5)            645         ['model_nn/dence_3[0][0]
↪']
```

(continues on next page)

(continued from previous page)

```
================================================================================================================
Total params: 246,917
Trainable params: 246,917
Non-trainable params: 0
----------------------------------------------------------------------------------------------------------------
```

Formation of the neural network architecture of the model (MuPTA)

[7]:
```
res_load_text_model_nn_mupta = _b5.load_text_model_nn(
    corpus = "mupta", # Corpus selection for models trained on First Impressions V2 'fi' and models trained on for MuPTA 'mupta'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 16:55:49] Formation of the neural network architecture of the model for obtaining scores by deep features (text modality) ...

— Runtime: 0.264 sec. —

Downloading the weights of the neural network model

[8]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['text']['mupta']['nn']['sberdisk']

res_load_text_model_weights_nn_mupta = _b5.load_text_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 16:55:51] Downloading the weights of a neural network model to obtain deep features (text modality) ...

[2023-12-10 16:55:52] File download “weights_2023-07-16_18-12-01.h5” 100.0% ...

— Runtime: 0.373 sec. —

Displaying the formed neural network architecture of the model

[9]:
```
_b5.text_model_nn_.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 model_nn/input (InputLayer)  [(None, 414, 768)]   0

 model_nn/bilstm_1 (Bidirection  (None, 414, 64)     205056
 al)

 model_nn/attention (Attention)  (None, 414, 64)     0
1[0][0]',
                                                     'model_nn/bilstm_
1[0][0]']

 model_nn/dence_2 (Dense)    (None, 414, 128)     8320
attention[0][0]']

 model_nn/add (Addition)     (None, 256)          0
']

 model_nn/dence_3 (Dense)    (None, 128)          32896

 dense_1 (Dense)             (None, 5)            645
']

=================================================================
Total params: 246,917
Trainable params: 246,917
Non-trainable params: 0
_________________________________________________________________
```

Formation of the neural network architecture of the model and downloading its weights to obtain personality traits scores (text modality)

- `_b5.text_model_b5_` - Neural network model **tf.keras.Model** for obtaining the personality traits scores

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

```
[2023-12-10 17:03:46] OCEANAI - personaly traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a16 License: BSD License
```

Formation of neural network architectures of models

[4]:
```
res_load_text_model_b5 = _b5.load_text_model_b5(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

```
[2023-12-10 17:03:46] Formation of neural network architectures of models for obtaining the personality traits scores (text modality) ...
```

```
— Runtime: 0.539 sec. —
```

Downloading weights of neural network models

FI V2

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['text']['fi']['b5']['sberdisk']

res_load_text_model_weights_b5 = _b5.load_text_model_weights_b5(
    url = url,
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 17:03:46] Downloading the weights of neural network models to obtain the personality traits scores (text modality) ...

[2023-12-14 21:04:19] File download "ft_fi_2023-12-09_14-25-13.h5"

— Runtime: 0.144 sec. —

MuPTA

[6]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # Размер загрузки файла из сети за 1 шаг

url = _b5.weights_for_big5_['text']['mupta']['b5']['sberdisk']

res_load_text_model_weights_b5 = _b5.load_text_model_weights_b5(
    url = url,
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-10 17:03:47] Downloading the weights of neural network models to obtain the personality traits scores (text modality) ...

[2023-12-10 17:03:47] File download "ft_mupta_2023-12-09_14-25-13.h5"

— Runtime: 0.137 sec. —

Displaying the formed neural network architecture of the model

[7]:
```
_b5.text_model_b5_.summary()
```

```
Model: "model"
_________________________________________________________________________________________
 Layer (type)                Output Shape              Param #     Connected to
=========================================================================================
 input_1 (InputLayer)        [(None, 5)]               0           []

 input_2 (InputLayer)        [(None, 5)]               0           []

 tf.concat (TFOpLambda)      (None, 10)                0           ['input_1[0][0]',
                                                                    'input_2[0][0]']

 dense (Dense)               (None, 5)                 55          ['tf.concat[0][0]']

=========================================================================================
Total params: 55
Trainable params: 55
Non-trainable params: 0
_________________________________________________________________________________________
```

Extracting features from a text

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

```
2023-12-03 00:29:47.655916: I tensorflow/core/platform/cpu_feature_guard.cc:193] This␣
↪TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use␣
↪the following CPU instructions in performance-critical operations:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler␣
↪flags.
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#FFF', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
```

(continues on next page)

(continued from previous page)

```
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-03 00:29:57] OCEANAI - personality traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a5 License: BSD License

Loading a dictionary with hand-crafted features

[4]:
```
# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

res_load_text_features = _b5.load_text_features(
    force_reload = True,       # Forced download file
    out = True,                # Display
    runtime = True,            # Runtime calculation
    run = True                 # Run blocking
)
```

[2023-12-03 00:29:57] Loading a dictionary with hand-crafted features ...

[2023-12-03 00:30:00] Loading the "LIWC2007.txt" file 100.0% ...

— Runtime: 3.073 sec. —

Building tokenizer and translation model (RU -¿ EN)

[5]:
```
res_setup_translation_model = _b5.setup_translation_model(
    out = True,      # Display
    runtime = True, # Runtime calculation
    run = True       # Run blocking
)
```

[2023-12-03 00:30:00] Building tokenizer and translation model ...

— Runtime: 3.098 sec. —

Building tokenizer and BERT model (for word encoding)

[6]:
```
# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

res_setup_translation_model = _b5.setup_bert_encoder(
    force_reload = True,       # Forced download file
    out = True,                # Display
    runtime = True,            # Runtime calculation
```

(continues on next page)

(continued from previous page)

```
    run = True                  # Run blocking
)
```

[2023-12-03 00:30:04] Building tokenizer and BERT model ...
[2023-12-03 00:30:07] Loading the “bert-base-multilingual-cased.zip” file**
[2023-12-03 00:30:04] Building tokenizer and BERT model ...
[2023-12-03 00:30:07] Loading the “bert-base-multilingual-cased.zip” file**
[2023-12-03 00:30:07] Unzipping an archive “bert-base-multilingual-cased.zip” ...
— Runtime: 14.752 sec. —

Process of extracting text features

Example 1 (Analyzing a video file (EN) with manual transcription)

[7]:
```
# Video file path
path = '/Users/dl/GitHub/OCEANAI/docs/source/user_guide/notebooks/glgfB3vFewc.004.mp4'

hc_features, nn_features = _b5.get_text_features(
    path = path, # Video file path
    asr = False, # Using a model for ASR
    lang = 'en', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    show_text = True, # Text display
    out = True,       # Display
    runtime = True,   # Runtime calculation
    run = True        # Run blocking
)
```

[2023-12-03 00:30:18] Extraction of features (hand-crafted and deep) from a text ...
**[2023-12-03 00:30:19] Statistics of extracted features from the text: ** Dimension of the matrix of hand-crafted features: 89 × 64 Dimension of the matrix of deep features: 104 × 768 Text: during those times i feel sad i feel confused and
— Runtime: 0.343 sec. —

Example 2 (Analyzing a video file (EN) without manual transcription)

[8]:
```
# Video file path
path = '/Users/dl/GitHub/OCEANAI/docs/source/user_guide/notebooks/glgfB3vFewc.004.mp4'

hc_features, nn_features = _b5.get_text_features(
    path = path, # Video file path
    asr = True, # Using a model for ASR
    lang = 'en', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    show_text = True, # Text display
    out = True,       # Display
    runtime = True,   # Runtime calculation
```

(continues on next page)

(continued from previous page)

```
    run = True          # Run blocking
)
```

[2023-12-03 00:30:19] Extraction of features (hand-crafted and deep) from a text ...
**[2023-12-03 00:30:25] Statistics of extracted features from the text: ** Dimension of the matrix of hand-crafted features: 89 × 64 Dimension of the matrix of deep features: 104 × 768 Text: during those times i feel sad i feel confused and- the school and introduce them to our administrators and the different faculty that work throughout the school and the library and the gym and so on and then they can get comfortable if theyre in a new school as well
— Runtime: 6.398 sec. —

Example 3 (Analyzing a video file (RU) without manual transcription)

[9]:
```
# Video file path
path = '/Users/dl/GitHub/OCEANAI/docs/source/user_guide/notebooks/center_42.mov'

hc_features, nn_features = _b5.get_text_features(
    path = path, # Video file path
    asr = False, # Using a model for ASR
    lang = 'ru', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    show_text = True, # Text display
    out = True,       # Display
    runtime = True,   # Runtime calculation
    run = True        # Run blocking
)
```

[2023-12-03 00:30:25] Extraction of features (hand-crafted and deep) from a text ...
**[2023-12-03 00:30:43] Statistics of extracted features from the text: ** Dimension of the matrix of hand-crafted features: 365 × 64 Dimension of the matrix of deep features: 414 × 768 Text: на картинке изображены скорее всего друзья которые играют в груз мечом это скорее всего происходит где-то в америке возможно в калифорнии на пляже девушка в топе и в шортах пытается словить мяч также двое парней смотрят одинаково думает как перехватить следующую подачу меча на заднем фоне видны высокие пальмы стоят дома неба голубое песок чистой чётко написки отображаются силой этой людей у парня в дали одеты солнце защитные очки он также в шортах и в майке в близи не видно головы человека он одет в темные шорты и в серую фортболку
— Runtime: 18.045 sec. —

Example 4 (Text Analysis - ``RU")

[10]:
```
# Text
path = '''
На картинке изображены скорее всего друзья, которые играют в игру с мячом.
Это скорее всего происходит где-то в Америке, возможно, в Калифорнии на пляже.
Девушка в топе и в шортах пытается словить мяч. Также двое парней смотрят, один активно␣
↪думает,
как перехватить следующую подачу мяча. На заднем фоне видны высокие пальмы. Стоят дома.
Небо голубое. Песок чистый. Чётко на песке отображаются силуэты людей. У парня вдали␣
```

(continues on next page)

(continued from previous page)

```
↪одеты солнцезащитные очки,
он также в шортах и в майке. Вблизи не видно головы человека. Он одет в тёмные шорты и в␣
↪серую футболку.
'''

hc_features, nn_features = _b5.get_text_features(
    path = path, # Text
    asr = False, # Using a model for ASR
    lang = 'ru', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    show_text = True, # Text display
    out = True,       # Display
    runtime = True,   # Runtime calculation
    run = True        # Run blocking
)
```

[2023-12-03 00:30:43] Extraction of features (hand-crafted and deep) from a text ...

**[2023-12-03 00:30:52] Statistics of extracted features from the text: ** Dimension of the matrix of hand-crafted features: 365 × 64 Dimension of the matrix of deep features: 414 × 768 Text: на картинке изображены скорее всего друзья которые играют в игру с мячом это скорее всего происходит где-то в америке возможно в калифорнии на пляже девушка в топе и в шортах пытается словить мяч также двое парней смотрят один активно думает как перехватить следующую подачу мяча на заднем фоне видны высокие пальмы стоят дома небо голубое песок чистый чётко на песке отображаются силуэты людей у парня вдали одеты солнцезащитные очки он также в шортах и в майке вблизи не видно головы человека он одет в тёмные шорты и в серую футболку

— Runtime: 9.227 sec. —

Example 5 (Text Analysis - ``EN")

[11]:
```
# Text
path = '''
today says they to for that but right now i am just watching super girl a new images be␣
↪catching up
and some shows a good say you guys
'''

hc_features, nn_features = _b5.get_text_features(
    path = path, # Text
    asr = False, # Using a model for ASR
    lang = 'en', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    show_text = True, # Text display
    out = True,       # Display
    runtime = True,   # Runtime calculation
    run = True        # Run blocking
)
```

[2023-12-03 00:30:52] Extraction of features (hand-crafted and deep) from a text ...

**[2023-12-03 00:30:53] Statistics of extracted features from the text: ** Dimension of the matrix of hand-crafted features: 89 × 64 Dimension of the matrix of deep features: 104 × 768 Text: today

(continues on next page)

(continued from previous page)

says they to for that but right now i am just watching super girl a new images be catching up and some shows a good say you guys

— Runtime: 0.247 sec. —

Example 5 (Analysing a text file - ``EN")

[12]:
```
# Text
path = '/Users/dl/GitHub/OCEANAI/docs/source/user_guide/notebooks/glgfB3vFewc.004.txt'

hc_features, nn_features = _b5.get_text_features(
    path = path, # Text
    asr = False, # Using a model for ASR
    lang = 'en', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    show_text = True, # Text display
    out = True,       # Display
    runtime = True,   # Runtime calculation
    run = True        # Run blocking
)
```

[2023-12-03 00:30:53] Extraction of features (hand-crafted and deep) from a text ...

**[2023-12-03 00:30:53] Statistics of extracted features from the text: ** Dimension of the matrix of hand-crafted features: 89 × 64 Dimension of the matrix of deep features: 104 × 768 Text: during those times i feel sad i feel confused and

— Runtime: 0.204 sec. —

Getting text scores

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Inference language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
```

(continues on next page)

(continued from previous page)

```
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-14 18:07:43] OCEANAI - personaly traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a16 License: BSD License

Getting and displaying versions of installed libraries

- `_b5.df_pkgs_` - DataFrame with versions of installed libraries

[4]:
```
_b5.libs_vers(runtime = True, run = True)
```

	Package	Version
1	TensorFlow	2.15.0
2	Keras	2.15.0
3	OpenCV	4.8.1
4	MediaPipe	0.9.0
5	NumPy	1.26.2
6	SciPy	1.11.4
7	Pandas	2.1.3
8	Scikit-learn	1.3.2
9	OpenSmile	2.5.0
10	Librosa	0.10.1
11	AudioRead	3.0.1
12	IPython	8.18.1
13	PyMediaInfo	6.1.0
14	Requests	2.31.0
15	JupyterLab	4.0.9
16	LIWC	0.5.0
17	Transformers	4.36.0
18	Sentencepiece	0.1.99
19	Torch	2.0.1+cpu
20	Torchaudio	2.0.2+cpu

— Runtime: 0.006 sec. —

Loading a dictionary with hand-crafted features

[5]:
```
# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

res_load_text_features = _b5.load_text_features(
    force_reload = True,       # Forced download file
    out = True,                # Display
    runtime = True,            # Runtime calculation
    run = True                 # Run blocking
)
```

[2023-12-14 18:07:43] Loading a dictionary with hand-crafted features ...
[2023-12-03 00:30:00] Loading the "LIWC2007.txt" file 100.0% ...
— Runtime: 0.232 sec. —

Building tokenizer and translation model (RU -¿ EN)

[6]:
```
res_setup_translation_model = _b5.setup_translation_model(
    out = True,      # Display
    runtime = True, # Runtime calculation
    run = True       # Run blocking
)
```

[2023-12-14 18:07:43] Building tokenizer and translation model ...
— Runtime: 1.71 sec. —

Building tokenizer and BERT model (for word encoding)

[7]:
```
# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

res_setup_translation_model = _b5.setup_bert_encoder(
    force_reload = True,       # Forced download file
    out = True,                # Display
    runtime = True,            # Runtime calculation
    run = True                 # Run blocking
)
```

[2023-12-14 18:07:45] Building tokenizer and BERT model ...
[2023-12-14 18:07:47] Loading the "bert-base-multilingual-cased.zip" file
[2023-12-14 18:07:45] Building tokenizer and BERT model ...
[2023-12-14 18:07:47] Loading the "bert-base-multilingual-cased.zip" file
[2023-12-14 18:07:47] Unzipping an archive "bert-base-multilingual-cased.zip" ...
— Runtime: 4.188 sec. —

FI V2

Formation of neural network architectures of models for obtaining scores by hand-crafted features

- `_b5.text_model_hc_` - Neural network model **tf.keras.Model** for obtaining scores by hand-crafted features

[8]:
```
res_load_text_model_hc_mupta = _b5.load_text_model_hc(
    corpus = "fi", # Corpus selection for models trained on First Impressions V2 'fi' and␣
↪models trained on for MuPTA 'mupta'
    show_summary = False, # Displaying the formed neural network architecture of the model
```

(continues on next page)

(continued from previous page)

```
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 18:07:49] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (text modality) ...

— Runtime: 0.647 sec. —

Downloading the weights of the neural network model for obtaining scores by hand-crafted features

- `_b5.text_model_hc_` - Neural network model **tf.keras.Model** for obtaining scores by hand-crafted features

[9]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['text']['fi']['hc']['sberdisk']

res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,      # Display
    runtime = True, # Runtime count
    run = True       # Run blocking
)
```

[2023-12-14 18:07:50] Downloading the weights of a neural network model to obtain hand-crafted features (text modality) ...

[2023-12-14 18:07:50] File download "weights_2023-07-15_10-52-15.h5" 100.0% ...

— Runtime: 0.289 sec. —

Formation of the neural network architecture of the model to obtain scores by deep features

- `_b5s.text_model_nn_` - Neural network model **tf.keras.Model** for obtaining scores by deep features

[10]:
```
res_load_text_model_nn_fi = _b5.load_text_model_nn(
    corpus = "fi", # Corpus selection for models trained on First Impressions V2 'fi' and␣
↪models trained on for MuPTA 'mupta'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 18:07:50] Formation of a neural network architecture for obtaining scores by deep features (text modality) ...

— Runtime: 0.279 sec. —

Downloading the weights of the neural network model for obtaining scores by deep features

- `_b5s.text_model_nn_` - Neural network model **tf.keras.Model** for obtaining scores by deep features

[11]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['text']['fi']['nn']['sberdisk']

res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,      # Display
    runtime = True, # Runtime count
    run = True       # Run blocking
)
```

[2023-12-14 18:07:50] Downloading the weights of a neural network model to obtain deep features (text modality) ...

[2023-12-14 18:07:51] File download "weights_2023-07-03_15-01-08.h5" 100.0% ...

— Runtime: 0.337 sec. —

Formation of the neural network architecture of the model to obtain personality traits scores

- `_b5.text_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

[12]:
```
res_load_text_model_b5 = _b5.load_text_model_b5(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 18:07:51] Formation of neural network architectures of models for obtaining the personality traits scores (text modality) ...

— Runtime: 0.015 sec. —

Downloading weights of neural network models for obtaining the personality traits scores

- `_b5.text_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

[13]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['text']['fi']['b5']['sberdisk']

res_load_text_model_weights_b5 = _b5.load_text_model_weights_b5(
    url = url,
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 18:07:51] Downloading the weights of neural network models to obtain the personality traits scores (text modality) ...

[2023-12-14 18:07:51] File download "ft_fi_2023-12-09_14-25-13.h5"

— Runtime: 0.163 sec. —

Getting scores (text modality)

- `_b5.df_files_` - DataFrame with data
- `_b5.df_accuracy_` - DataFrame with accuracy

[14]:
```
# Core settings
_b5.path_to_dataset_ = 'E:/Databases/FirstImpressionsV2/test' # Dataset directory
# Directories not included in the selection
_b5.ignore_dirs_ = []
# HKey names for DataFrame dataset
_b5.keys_dataset_ = ['Path', 'Openness', 'Conscientiousness', 'Extraversion',
↪'Agreeableness', 'Non-Neuroticism']
_b5.ext_ = ['.mp4'] # Search file extensions
_b5.path_to_logs_ = './logs' # Directory for saving LOG files

# Full path to the file containing the ground truth scores for the accuracy calculation
url_accuracy = _b5.true_traits_['fi']['sberdisk']

res_get_text_union_predictions = _b5.get_text_union_predictions(
    depth = 1,          # Hierarchy depth for receiving video
    recursive = False, # Recursive data search
    asr = True,         # Using a model for ASR
    lang = 'en', # Language selection for models trained on First Impressions V2 'en' and␣
↪models trained on for MuPTA 'ru'
    accuracy = True,    # Accuracy calculation
    url_accuracy = url_accuracy,
```

(continues on next page)

(continued from previous page)

```
    logs = True,        # If necessary, generate a LOG file
    out = True,         # Display
    runtime = True,     # Runtime count
    run = True          # Run blocking
)
```

[2023-12-14 19:00:14] Feature extraction (hand-crafted and deep) from text ...

[2023-12-14 19:00:15] Getting scores and accuracy calculation (text modality) ...

2000 from 2000 (100.0%) ... test80_25_Q4wOgixh7E.004.mp4 ...

```
                                                 Path  Openness  \
ID
1   E:\Databases\FirstImpressionsV2\test\test80_01...  0.624434
2   E:\Databases\FirstImpressionsV2\test\test80_01...  0.518305
3   E:\Databases\FirstImpressionsV2\test\test80_01...  0.516165
4   E:\Databases\FirstImpressionsV2\test\test80_01...  0.653522
5   E:\Databases\FirstImpressionsV2\test\test80_01...  0.672823
6   E:\Databases\FirstImpressionsV2\test\test80_01...  0.571563
7   E:\Databases\FirstImpressionsV2\test\test80_01...  0.579048
8   E:\Databases\FirstImpressionsV2\test\test80_01...  0.547369
9   E:\Databases\FirstImpressionsV2\test\test80_01...  0.630611
10  E:\Databases\FirstImpressionsV2\test\test80_01...  0.643665
11  E:\Databases\FirstImpressionsV2\test\test80_01...  0.610431
12  E:\Databases\FirstImpressionsV2\test\test80_01...  0.501841
13  E:\Databases\FirstImpressionsV2\test\test80_01...  0.516751
14  E:\Databases\FirstImpressionsV2\test\test80_01...  0.625826
15  E:\Databases\FirstImpressionsV2\test\test80_01...  0.506065
16  E:\Databases\FirstImpressionsV2\test\test80_01...  0.638552
17  E:\Databases\FirstImpressionsV2\test\test80_01...   0.51764
18  E:\Databases\FirstImpressionsV2\test\test80_01...  0.581101
19  E:\Databases\FirstImpressionsV2\test\test80_01...  0.545621
20  E:\Databases\FirstImpressionsV2\test\test80_01...  0.619155
21  E:\Databases\FirstImpressionsV2\test\test80_01...   0.58491
22  E:\Databases\FirstImpressionsV2\test\test80_01...  0.504319
23  E:\Databases\FirstImpressionsV2\test\test80_01...  0.587255
24  E:\Databases\FirstImpressionsV2\test\test80_01...    0.6448
25  E:\Databases\FirstImpressionsV2\test\test80_01...  0.575514
26  E:\Databases\FirstImpressionsV2\test\test80_01...  0.561977
27  E:\Databases\FirstImpressionsV2\test\test80_01...  0.522762
28  E:\Databases\FirstImpressionsV2\test\test80_01...  0.642535
29  E:\Databases\FirstImpressionsV2\test\test80_01...  0.615789
30  E:\Databases\FirstImpressionsV2\test\test80_01...  0.620333

   Conscientiousness Extraversion Agreeableness Non-Neuroticism
ID
1           0.588915      0.53729      0.601771        0.587032
2           0.405696     0.440837      0.486431         0.42919
3           0.482939     0.419187      0.520959         0.46346
4           0.645953       0.5613       0.63864        0.635908
5           0.563164     0.597474      0.618239        0.627377
6            0.49441     0.477624      0.548336        0.509708
7           0.590844     0.470888      0.580203        0.545247
```

(continues on next page)

(continued from previous page)

```
8           0.540064     0.441378       0.55407        0.52564
9           0.546466     0.548925      0.592785       0.576801
10          0.650126     0.561841       0.63202       0.636658
11          0.509742     0.532337      0.563182       0.548405
12          0.438787     0.408134      0.493867       0.433236
13          0.521908     0.412392      0.535759       0.475492
14          0.595756     0.545166      0.608196       0.601571
15          0.466968     0.428299      0.497129       0.451425
16          0.564402     0.561068      0.599493       0.594701
17          0.588128     0.392461      0.569938       0.512308
18          0.516556     0.489761      0.557651       0.521073
19          0.467661      0.46827      0.518607       0.478676
20          0.529129     0.535892       0.58141       0.571938
21          0.489063     0.500084      0.538159       0.525135
22          0.449576     0.427531      0.488319       0.441239
23          0.591969      0.50329      0.578679       0.566444
24           0.58204     0.558367       0.61345        0.60149
25          0.517498     0.481397      0.548056       0.514953
26          0.594428     0.456222      0.562595       0.536081
27          0.468697     0.426084      0.510566       0.451157
28          0.538425     0.564254      0.602641       0.595872
29           0.54139     0.522493      0.585496       0.570682
30          0.522955     0.543902      0.569043       0.559107
```

[2023-12-14 19:00:16] Trait-wise accuracy ...

```
          Openness  Conscientiousness  Extraversion  Agreeableness  \
Metrics
MAE         0.1097              0.114         0.115         0.1019
Accuracy    0.8903              0.886         0.885         0.8981

          Non-Neuroticism    Mean
Metrics
MAE                0.1154  0.1112
Accuracy           0.8846  0.8888
```

[2023-12-14 19:00:16] Mean absolute errors: 0.1112, average accuracy: 0.8888 ...

Log files saved successfully ...

— Runtime: 3131.846 sec. —

Multimodal information processing

Formation of neural network architectures of models and downloading their weights to obtain the personality traits scores (audio and video fusion)

- `_b5.av_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en',               # Inference language
    color_simple = '#333',     # Plain text color (hexadecimal code)
    color_info = '#1776D2',    # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000',     # Error text color (hexadecimal code)
    color_true = '#008001',    # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True,          # Bold text
    num_to_df_display = 30,    # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True            # Displaying information about library
)
```

```
[2023-12-14 22:44:38] OCEANAI - personal traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a16 License: BSD License
```

Formation of neural network architectures of models

[4]:
```
res_load_av_models_b5 = _b5.load_av_models_b5(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

```
[2023-12-14 22:44:38] Formation of neural network architectures of models for obtaining the personality traits scores (multimodal fusion) ...
— Runtime: 0.095 sec. —
```

Downloading weights of neural network models

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url_openness = _b5.weights_for_big5_['av']['fi']['b5']['openness']['sberdisk']
url_conscientiousness = _b5.weights_for_big5_['av']['fi']['b5']['conscientiousness'][
↪'sberdisk']
```

(continues on next page)

(continued from previous page)

```
url_extraversion = _b5.weights_for_big5_['av']['fi']['b5']['extraversion']['sberdisk']
url_agreeableness = _b5.weights_for_big5_['av']['fi']['b5']['agreeableness']['sberdisk']
url_non_neuroticism = _b5.weights_for_big5_['av']['fi']['b5']['non_neuroticism']['
↪'sberdisk']

res_load_av_models_weights_b5 = _b5.load_av_models_weights_b5(
    url_openness = url_openness, # Openness
    url_conscientiousness = url_conscientiousness, # Conscientiousness
    url_extraversion = url_extraversion, # Extraversion
    url_agreeableness = url_agreeableness, # Agreeableness
    url_non_neuroticism = url_non_neuroticism, # Non-Neuroticism
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-14 22:44:53] Downloading the weights of neural network models to obtain the personality traits scores (multimodal fusion) ...
[2023-12-14 22:44:53] File download "weights_2022-08-28_11-14-35.h5" (100.0%) ... Openness
[2023-12-14 22:44:54] File download "weights_2022-08-28_11-08-10.h5" (100.0%) ... Conscientiousness
[2023-12-14 22:44:54] File download "weights_2022-08-28_11-17-57.h5" (100.0%) ... Extraversion
[2023-12-14 22:44:54] File download "weights_2022-08-28_11-25-11.h5" (100.0%) ... Agreeableness
[2023-12-14 22:44:54] File download "weights_2022-06-14_21-44-09.h5" (100.0%) ... Non-Neuroticism
— Runtime: 0.914 sec. —

Displaying the formed neural network architecture of the model

- Openness
- Conscientiousness
- Extraversion
- Agreeableness
- Non-Neuroticism

[6]:
```
_b5.av_models_b5_['openness'].summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 64)]              0

 dense_1 (Dense)             (None, 1)                 65

 activ_1 (Activation)        (None, 1)                 0

```

(continues on next page)

(continued from previous page)

```
=================================================================
Total params: 65 (260.00 Byte)
Trainable params: 65 (260.00 Byte)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Formation of neural network architectures of models and downloading their weights to obtain the personality traits scores (audio, video and tex fusion)

- `_b5.avt_model_b5_` - Neural network model **tf.keras.Model** for obtaining the personality traits scores

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en',              # Inference language
    color_simple = '#333',    # Plain text color (hexadecimal code)
    color_info = '#1776D2',   # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000',    # Error text color (hexadecimal code)
    color_true = '#008001',   # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True,         # Bold text
    num_to_df_display = 30,   # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True           # Displaying information about library
)
```

```
[2023-12-14 22:44:38] OCEANAI - personal traits:  Authors:  Elena Ryumina
[ryumina_ev@mail.ru]  Dmitry Ryumin [dl_03.03.1991@mail.ru]  Alexey Karpov
[karpov@iias.spb.su]  Maintainers:  Elena Ryumina [ryumina_ev@mail.ru]  Dmitry Ryumin
[dl_03.03.1991@mail.ru]  Version: 1.0.0a5  License: BSD License
```

Formation of neural network architectures of models

[4]:
```
res_load_avt_model_b5 = _b5.load_avt_model_b5(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

```
[2023-12-11 09:46:45] Formation of neural network architectures of models for obtaining the personality
traits scores (multimodal fusion) ...
```

— Runtime: 0.814 sec. —

Downloading weights of neural network models

[5]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['avt']['fi']['b5']['sberdisk']

res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(
    url = url,
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-11 09:46:46] Downloading the weights of neural network models to obtain the personality traits scores (multimodal fusion) ...

[2023-12-11 09:46:46] File download "avt_fi_2023-12-03_11-36-51.h5"

— Runtime: 0.218 sec. —

Displaying the formed neural network architecture of the model

[6]:
```
_b5.avt_model_b5_.summary()
```

```
Model: "model"
__________________________________________________________________________________________
↪_________
 Layer (type)                Output Shape              Param #     Connected to
==========================================================================================
 hc_t (InputLayer)           [(None, 128)]             0           []

 hc_a (InputLayer)           [(None, 256)]             0           []

 nn_t (InputLayer)           [(None, 128)]             0           []

 nn_a (InputLayer)           [(None, 512)]             0           []

 hc_v (InputLayer)           [(None, 256)]             0           []

 nn_v (InputLayer)           [(None, 2048)]            0           []

 ln_hc_t (LayerNormalization)  (None, 128)             256         ['hc_t[0][0]']

 ln_hc_a (LayerNormalization)  (None, 256)             512         ['hc_a[0][0]']

 ln_nn_t (LayerNormalization)  (None, 128)             256         ['nn_t[0][0]']
```

(continues on next page)

(continued from previous page)

```
ln_nn_a (LayerNormalization)   (None, 512)          1024        ['nn_a[0][0]']

ln_hc_v (LayerNormalization)   (None, 256)          512         ['hc_v[0][0]']

ln_nn_v (LayerNormalization)   (None, 2048)         4096        ['nn_v[0][0]']

gata (GFL)                     (None, 64)           131072      ['ln_hc_t[0][0]',
                                                                 'ln_hc_a[0][0]',
                                                                 'ln_nn_t[0][0]',
                                                                 'ln_nn_a[0][0]']

gatv (GFL)                     (None, 64)           327680      ['ln_hc_t[0][0]',
                                                                 'ln_hc_v[0][0]',
                                                                 'ln_nn_t[0][0]',
                                                                 'ln_nn_v[0][0]']

gaav (GFL)                     (None, 64)           393216      ['ln_hc_a[0][0]',
                                                                 'ln_hc_v[0][0]',
                                                                 'ln_nn_a[0][0]',
                                                                 'ln_nn_v[0][0]']

tf.concat (TFOpLambda)         (None, 192)          0           ['gata[0][0]',
                                                                 'gatv[0][0]',
                                                                 'gaav[0][0]']

dense (Dense)                  (None, 50)           9650        ['tf.concat[0][0]']

dence_cl (Dense)               (None, 5)            255         ['dense[0][0]']

==================================================================================================
Total params: 868,529
Trainable params: 868,529
Non-trainable params: 0
__________________________________________________________________________________________
↪_________
```

Multimodal fusion to obtain scores by audio and video FI V2

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en',               # Inference language
    color_simple = '#333',     # Plain text color (hexadecimal code)
    color_info = '#1776D2',    # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000',     # Error text color (hexadecimal code)
    color_true = '#008001',    # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True,          # Bold text
    num_to_df_display = 30,    # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True            # Displaying information about library
)
```

[2023-12-14 22:46:31] OCEANAI - personal traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a16 License: BSD License

Getting and displaying versions of installed libraries

- `_b5.df_pkgs_` - DataFrame with versions of installed libraries

[4]:
```
_b5.libs_vers(runtime = True, run = True)
```

	Package	Version
1	TensorFlow	2.15.0
2	Keras	2.15.0
3	OpenCV	4.8.1
4	MediaPipe	0.9.0
5	NumPy	1.26.2
6	SciPy	1.11.4
7	Pandas	2.1.3
8	Scikit-learn	1.3.2
9	OpenSmile	2.5.0
10	Librosa	0.10.1
11	AudioRead	3.0.1
12	IPython	8.18.1
13	PyMediaInfo	6.1.0
14	Requests	2.31.0
15	JupyterLab	4.0.9
16	LIWC	0.5.0
17	Transformers	4.36.0
18	Sentencepiece	0.1.99

(continues on next page)

(continued from previous page)

```
19          Torch  2.0.1+cpu
20     Torchaudio  2.0.2+cpu
```

— Runtime: 0.006 sec. —

Analysing audio information (forming model and loading model weights)

Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (audio modality)

- `_b5.audio_model_hc_` - Neural network model **tf.keras.Model** for obtaining scores by hand-crafted features

[5]:
```
res_load_audio_model_hc = _b5.load_audio_model_hc(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-14 22:46:31] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (audio modality) ...

— Runtime: 0.322 sec. —

Downloading the weights of the neural network model to obtain scores by hand-crafted features (audio modality)

- `_b5.audio_model_hc_` - Neural network model **tf.keras.Model** for obtaining scores by hand-crafted features

[6]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['audio']['fi']['hc']['sberdisk']

res_load_audio_model_weights_hc = _b5.load_audio_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,          # Display
    runtime = True,      # Runtime count
    run = True           # Run blocking
)
```

[2023-12-14 22:46:31] Downloading the weights of the neural network model to obtain scores by hand-crafted features (audio modality) ...

[2023-12-14 22:46:32] File download "weights_2022-05-05_11-27-55.h5" (100.0%) ...

— Runtime: 0.277 sec. —

Formation of the neural network architecture of the model for obtaining scores by deep features (audio modality)

- `_b5.audio_model_nn_` - Neural network model **tf.keras.Model** for obtaining scores by deep features

[7]:
```
res_load_audio_model_nn = _b5.load_audio_model_nn(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-14 22:46:32] Formation of a neural network architecture for obtaining scores by deep features (audio modality) ...

— Runtime: 0.244 sec. —

Downloading the weights of the neural network model to obtain scores for deep features (audio modality)

- `_b5.audio_model_nn_` - Neural network model **tf.keras.Model** for obtaining scores by deep features

[8]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['audio']['fi']['nn']['sberdisk']

res_load_audio_model_weights_nn = _b5.load_audio_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-14 22:46:32] Downloading the weights of the neural network model to obtain scores for deep features (audio modality) ...

[2023-12-14 22:46:32] File download "weights_2022-05-03_07-46-14.h5"

— Runtime: 0.389 sec. —

Analysing video information (forming model and loading model weights)

Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (video modality)

- `_b5.video_model_hc_` - Neural network model **tf.keras.Model** for obtaining scores by hand-crafted features

[9]:
```
res_load_video_model_hc = _b5.load_video_model_hc(
    lang = 'en', # Language selection for models trained on First Impressions V2'en' and␣
```

(continues on next page)

(continued from previous page)

```
↪models trained on for MuPTA 'ru'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-14 22:46:32] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (video modality) ...

— Runtime: 0.257 sec. —

Downloading the weights of the neural network model to obtain scores by hand-crafted features (video modality)

- `_b5.video_model_hc_` - Neural network model **tf.keras.Model** for obtaining scores by hand-crafted features

[10]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['video']['fi']['hc']['sberdisk']

res_load_video_model_weights_hc = _b5.load_video_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-14 22:46:32] Downloading the weights of the neural network model to obtain scores by hand-crafted features (video modality) ...

[2023-12-14 22:46:33] File download "weights_2022-08-27_18-53-35.h5" (100.0%) ...

— Runtime: 0.226 sec. —

Formation of neural network architecture for obtaining deep features

- `_b5.video_model_deep_fe_` - Neural network model **tf.keras.Model** for obtaining deep features

[11]:
```
res_load_video_model_deep_fe = _b5.load_video_model_deep_fe(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-14 22:46:34] Formation of neural network architecture for obtaining deep features (video modality) ...

— Runtime: 0.783 sec. —

Downloading weights of a neural network model to obtain deep features

- `_b5.video_model_deep_fe_` - Neural network model **tf.keras.Model** for obtaining deep features

[12]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['video']['fi']['fe']['sberdisk']

res_load_video_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-14 22:46:35] Downloading weights of a neural network model to obtain deep features (video modality) ...

[2023-12-14 22:46:40] File download "weights_2022-11-01_12-27-07.h5" (100.0%) ...

— Runtime: 4.311 sec. —

Formation of the neural network architecture of the model for obtaining scores by deep features (video modality)

- `_b5.video_model_nn_` - Neural network model **tf.keras.Model** for obtaining scores by deep features

[13]:
```
res_load_video_model_nn = _b5.load_video_model_nn(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-14 22:46:40] Formation of a neural network architecture for obtaining scores by deep features (video modality) ...

— Runtime: 0.724 sec. —

Downloading the weights of the neural network model to obtain scores for deep features (video modality)

- `_b5.video_model_nn_` - Neural network model **tf.keras.Model** for obtaining scores by deep features

[14]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['video']['fi']['nn']['sberdisk']

res_load_video_model_weights_nn = _b5.load_video_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-14 22:46:40] Downloading the weights of the neural network model to obtain scores by deep features (video modality) ...

[2023-12-14 22:46:42] File downloading "weights_2022-03-22_16-31-48.h5"

— Runtime: 1.355 sec. —

Analysing multimodal information (forming model, loading model weights, obtaining personality traits scores)

Formation of neural network architectures of models for obtaining the personality traits scores (multimodal fusion)

- `_b5.av_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

[15]:
```
res_load_av_models_b5 = _b5.load_av_models_b5(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-14 22:46:42] Formation of neural network architectures of models for obtaining the personality traits scores (multimodal fusion) ...

— Runtime: 0.048 sec. —

Downloading the weights of neural network models to obtain the personality traits scores (multimodal fusion)

- `_b5.av_models_b5_` - Neural network models **tf.keras.Model** for obtaining the personality traits scores

[16]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url_openness = _b5.weights_for_big5_['av']['fi']['b5']['openness']['sberdisk']
url_conscientiousness = _b5.weights_for_big5_['av']['fi']['b5']['conscientiousness'][
↪'sberdisk']
url_extraversion = _b5.weights_for_big5_['av']['fi']['b5']['extraversion']['sberdisk']
url_agreeableness = _b5.weights_for_big5_['av']['fi']['b5']['agreeableness']['sberdisk']
url_non_neuroticism = _b5.weights_for_big5_['av']['fi']['b5']['non_neuroticism'][
↪'sberdisk']

res_load_av_models_weights_b5 = _b5.load_av_models_weights_b5(
    url_openness = url_openness,                    # Openness
    url_conscientiousness = url_conscientiousness, # Conscientiousness
    url_extraversion = url_extraversion,            # Extraversion
    url_agreeableness = url_agreeableness,          # Agreeableness
    url_neuroticism = url_neuroticism,              # Non-Neuroticism
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,          # Display
    runtime = True,      # Runtime count
    run = True           # Run blocking
)
```

[2023-12-14 22:46:47] Downloading the weights of neural network models to obtain the personality traits scores (multimodal fusion) ...
[2023-12-14 22:46:47] File download "weights_2022-08-28_11-14-35.h5" Openness
[2023-12-14 22:46:47] File download "weights_2022-08-28_11-08-10.h5" Conscientiousness
[2023-12-14 22:46:47] File download "weights_2022-08-28_11-17-57.h5" Extraversion
[2023-12-14 22:46:47] File download "weights_2022-08-28_11-25-11.h5" Agreeableness
[2023-12-14 22:46:47] File download "weights_2022-06-14_21-44-09.h5" Non-Neuroticism
— Runtime: 0.785 sec. —

Getting scores (multimodal fusion)

- `_b5.df_files_` - DataFrame with data
- `_b5.df_accuracy_` - DataFrame with accuracy

[17]:
```
# Core settings
_b5.path_to_dataset_ = 'E:/Databases/FirstImpressionsV2/test' # Dataset directory
# Directories not included in the selection
_b5.ignore_dirs_ = []
# Key names for DataFrame dataset
_b5.keys_dataset_ = ['Path', 'Openness', 'Conscientiousness', 'Extraversion',
```

(continues on next page)

(continued from previous page)

```
↪'Agreeableness', 'Non-Neuroticism']
_b5.ext_ = ['.mp4'] # Search file extensions

# Full path to the file containing the ground truth scores for the accuracy calculation
url_accuracy = _b5.true_traits_['fi']['sberdisk']

_b5.get_av_union_predictions(
    depth = 2,          # Hierarchy depth for receiving audio and video data
    recursive = False, # Recursive data search
    sr = 44100,         # Sampling frequency
    window_audio = 2,  # Audio segment window size (in seconds)
    step_audio = 1,     # Audio segment window shift step (in seconds)
    reduction_fps = 5, # Frame rate reduction
    window_video = 10, # Video segment window size (in seconds)
    step_video = 5,     # Video segment window shift step (in seconds)
    lang = 'en',        # Language selection for models trained on First Impressions V2'en
↪' and models trained on for MuPTA 'ru'
    accuracy = True,    # Accuracy
    url_accuracy = url_accuracy,
    logs = True,        # If necessary, generate a LOG file
    out = True,         # Display
    runtime = True,     # Runtime count
    run = True          # Run blocking
)
```

[2023-12-15 01:11:04] Getting scores and accuracy calculation (multimodal fusion) ...

2000 from 2000 (100.0%) ... test80_25_Q4wOgixh7E.004.mp4 ...

ID	Path	Openness	\
1	E:\Databases\FirstImpressionsV2\test\test80_01...	0.554249	
2	E:\Databases\FirstImpressionsV2\test\test80_01...	0.558823	
3	E:\Databases\FirstImpressionsV2\test\test80_01...	0.477549	
4	E:\Databases\FirstImpressionsV2\test\test80_01...	0.662656	
5	E:\Databases\FirstImpressionsV2\test\test80_01...	0.645876	
6	E:\Databases\FirstImpressionsV2\test\test80_01...	0.67497	
7	E:\Databases\FirstImpressionsV2\test\test80_01...	0.39908	
8	E:\Databases\FirstImpressionsV2\test\test80_01...	0.577705	
9	E:\Databases\FirstImpressionsV2\test\test80_01...	0.543675	
10	E:\Databases\FirstImpressionsV2\test\test80_01...	0.54876	
11	E:\Databases\FirstImpressionsV2\test\test80_01...	0.546634	
12	E:\Databases\FirstImpressionsV2\test\test80_01...	0.459302	
13	E:\Databases\FirstImpressionsV2\test\test80_01...	0.309097	
14	E:\Databases\FirstImpressionsV2\test\test80_01...	0.643403	
15	E:\Databases\FirstImpressionsV2\test\test80_01...	0.65016	
16	E:\Databases\FirstImpressionsV2\test\test80_01...	0.598313	
17	E:\Databases\FirstImpressionsV2\test\test80_01...	0.571537	
18	E:\Databases\FirstImpressionsV2\test\test80_01...	0.552433	
19	E:\Databases\FirstImpressionsV2\test\test80_01...	0.658695	
20	E:\Databases\FirstImpressionsV2\test\test80_01...	0.660076	
21	E:\Databases\FirstImpressionsV2\test\test80_01...	0.543881	
22	E:\Databases\FirstImpressionsV2\test\test80_01...	0.537325	

(continues on next page)

(continued from previous page)

```
23  E:\Databases\FirstImpressionsV2\test\test80_01...  0.464761
24  E:\Databases\FirstImpressionsV2\test\test80_01...  0.633951
25  E:\Databases\FirstImpressionsV2\test\test80_01...    0.4517
26  E:\Databases\FirstImpressionsV2\test\test80_01...  0.602848
27  E:\Databases\FirstImpressionsV2\test\test80_01...  0.586638
28  E:\Databases\FirstImpressionsV2\test\test80_01...  0.689552
29  E:\Databases\FirstImpressionsV2\test\test80_01...  0.583505
30  E:\Databases\FirstImpressionsV2\test\test80_01...  0.642695

   Conscientiousness Extraversion Agreeableness Non-Neuroticism
ID
1           0.506548     0.440194      0.540235         0.48605
2           0.442357      0.50397      0.558767        0.521587
3           0.568616     0.333939      0.491873        0.458966
4           0.621852      0.58996      0.599038        0.636035
5           0.532378     0.551939      0.589174        0.552269
6           0.666972     0.617604      0.610567        0.641452
7           0.397298     0.335823      0.497966         0.39729
8           0.597157     0.498064      0.640584        0.600152
9           0.451197     0.449555      0.482371        0.415256
10           0.51097     0.433856      0.579709        0.536171
11          0.398485     0.443701      0.518107        0.492343
12          0.427114     0.315686      0.495817        0.457954
13          0.317028     0.218514      0.372315        0.241697
14          0.509414     0.483608      0.503154        0.550979
15          0.840148     0.535299      0.710939        0.743357
16          0.520505     0.450767      0.486345        0.561532
17          0.673989     0.472203      0.615608        0.621064
18          0.568787     0.457108      0.613188        0.570902
19          0.625194     0.634877      0.612277        0.626052
20          0.544358      0.64178      0.604572        0.628259
21          0.477881     0.407731      0.555772        0.499664
22           0.46375     0.419255      0.499785        0.455146
23          0.434816     0.346836      0.428429        0.358087
24           0.63333     0.584644      0.615227        0.608006
25          0.574346     0.350136      0.526873        0.468283
26          0.592382     0.494679      0.539232        0.505865
27          0.521421     0.485391      0.530296        0.535499
28          0.643902     0.695799      0.646209        0.686243
29          0.564313     0.502263      0.554502        0.539899
30          0.588222     0.617706      0.615312        0.626649
```

[2023-12-15 01:11:04] Trait-wise accuracy ...

```
          Openness  Conscientiousness  Extraversion  Agreeableness  \
Metrics
MAE         0.0845             0.0802        0.0793         0.0858
Accuracy    0.9155             0.9198        0.9207         0.9142

          Non-Neuroticism    Mean
Metrics
MAE                0.0847  0.0829
Accuracy           0.9153  0.9171
```

[2023-12-15 01:11:04] Mean absolute error: 0.0829, Accuracy: 0.9171 ...
Log files saved successfully ...
— Runtime: 8654.754 sec. —

[17]:
```
True
```

Multimodal fusion to obtain scores by audio, video and text FI V2

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en',               # Inference language
    color_simple = '#333',     # Plain text color (hexadecimal code)
    color_info = '#1776D2',    # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000',     # Error text color (hexadecimal code)
    color_true = '#008001',    # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True,          # Bold text
    num_to_df_display = 30,    # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True            # Displaying information about library
)
```

[2023-12-15 07:01:44] OCEANAI - personal traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a16 License: BSD License

Getting and displaying versions of installed libraries

- `_b5.df_pkgs_` - DataFrame with versions of installed libraries

[4]:
```
_b5.libs_vers(runtime = True, run = True)
```

```
          Package    Version
1      TensorFlow     2.15.0
2           Keras     2.15.0
3          OpenCV      4.8.1
4       MediaPipe      0.9.0
5           NumPy     1.26.2
```

(continues on next page)

(continued from previous page)

```
6           SciPy     1.11.4
7          Pandas      2.1.3
8    Scikit-learn      1.3.2
9       OpenSmile      2.5.0
10        Librosa     0.10.1
11      AudioRead      3.0.1
12        IPython     8.18.1
13    PyMediaInfo      6.1.0
14       Requests     2.31.0
15     JupyterLab      4.0.9
16           LIWC      0.5.0
17   Transformers     4.36.0
18  Sentencepiece     0.1.99
19          Torch  2.0.1+cpu
20     Torchaudio  2.0.2+cpu
```

— Runtime: 0.004 sec. —

Analysing audio information (forming model and loading model weights)

Formation of the neural network architecture of the model for obtaining features / scores by hand-crafted features (audio modality)

- `_b5.audio_model_hc_` - Neural network model **tf.keras.Model** for obtaining features / scores by hand-crafted features

[5]:
```
res_load_audio_model_hc = _b5.load_audio_model_hc(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-15 07:01:44] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (audio modality) ...

— Runtime: 0.326 sec. —

Downloading the weights of the neural network model to obtain features / scores by hand-crafted features (audio modality)

- `_b5.audio_model_hc_` - Neural network model **tf.keras.Model** for obtaining features / scores by hand-crafted features

[6]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step
corpus = 'fi'
lang = 'en'

url = _b5.weights_for_big5_['audio'][corpus]['hc']['sberdisk']
```

(continues on next page)

(continued from previous page)

```
res_load_audio_model_weights_hc = _b5.load_audio_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[[2023-12-15 07:01:45] Downloading the weights of the neural network model to obtain scores by hand-crafted features (audio modality) ...

[2023-12-15 07:01:45] File download "weights_2022-05-05_11-27-55.h5" (100.0%) ...

— Runtime: 0.226 sec. —

Formation of the neural network architecture of the model for obtaining features / scores by deep features (audio modality)

- `_b5.audio_model_nn_` - Neural network model **tf.keras.Model** for obtaining features / scores by deep features

[7]:
```
res_load_audio_model_nn = _b5.load_audio_model_nn(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-15 07:01:45] Formation of a neural network architecture for obtaining scores by deep features (audio modality) ...

— Runtime: 0.219 sec. —

Downloading the weights of the neural network model to obtain features / scores for deep features

- `_b5.audio_model_nn_` - Neural network model **tf.keras.Model** for obtaining features / scores by deep features

[8]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['audio'][corpus]['nn']['sberdisk']

res_load_audio_model_weights_nn = _b5.load_audio_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,           # Display
    runtime = True,       # Runtime count
```

(continues on next page)

(continued from previous page)

```
    run = True              # Run blocking
)
```

[2023-12-15 07:01:45] Downloading the weights of the neural network model to obtain scores for deep features (audio modality) ...

[2023-12-15 07:01:45] File download "weights_2022-05-03_07-46-14.h5"

— Runtime: 0.328 sec. —

Analysing video information (forming model and loading model weights)

Formation of the neural network architecture of the model for obtaining features / scores by hand-crafted features (audio modality)

- `_b5.video_model_hc_` - Neural network model **tf.keras.Model** for obtaining features / scores by hand-crafted features

[9]:
```
res_load_video_model_hc = _b5.load_video_model_hc(
    lang = lang, # Language selection for models trained on First Impressions V2'en' and␣
↪models trained on for MuPTA 'ru'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-15 07:01:45] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (video modality) ...

— Runtime: 0.252 sec. —

Downloading the weights of the neural network model to obtain features / scores by hand-crafted features (audio modality)

- `_b5.video_model_hc_` - Neural network model **tf.keras.Model** for obtaining features / scores by hand-crafted features

[10]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['video'][corpus]['hc']['sberdisk']

res_load_video_model_weights_hc = _b5.load_video_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-15 07:01:46] Downloading the weights of the neural network model to obtain scores by hand-crafted features (video modality) ...
[2023-12-15 07:01:46] File download "weights_2022-08-27_18-53-35.h5" (100.0%) ...
— Runtime: 0.24 sec. —

Formation of neural network architecture for obtaining deep features

- `_b5.video_model_deep_fe_` - Neural network model **tf.keras.Model** for obtaining deep features

[11]:
```
res_load_video_model_deep_fe = _b5.load_video_model_deep_fe(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-15 07:01:46] Formation of neural network architecture for obtaining deep features (video modality) ...
— Runtime: 0.794 sec. —

Downloading weights of a neural network model to obtain deep features (video modality)

- `_b5.video_model_deep_fe_` - Neural network model **tf.keras.Model** for obtaining deep features

[12]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

url = _b5.weights_for_big5_['video'][corpus]['fe']['sberdisk']

res_load_video_model_weights_deep_fe = _b5.load_video_model_weights_deep_fe(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-15 07:01:47] Downloading weights of a neural network model to obtain deep features (video modality) ...
[2023-12-15 07:01:50] File download "weights_2022-11-01_12-27-07.h5" (100.0%) ...

— Runtime: 3.937 sec. —

Formation of the neural network architecture of the model for obtaining features / scores by deep features (audio modality)

- `_b5.video_model_nn_` - Neural network model **tf.keras.Model** for obtaining features / scores by deep features

[13]:
```
res_load_video_model_nn = _b5.load_video_model_nn(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-15 07:01:51] Formation of a neural network architecture for obtaining scores by deep features (video modality) ...

— Runtime: 0.707 sec. —

Downloading the weights of the neural network model to obtain features / scores for deep features

- `_b5.video_model_nn_` - Neural network model **tf.keras.Model** for obtaining features / scores by deep features

[14]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['video'][corpus]['nn']['sberdisk']

res_load_video_model_weights_nn = _b5.load_video_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = False, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-15 07:01:51] Downloading the weights of the neural network model to obtain scores by deep features (video modality) ...

[2023-12-15 07:01:51] File downloading "weights_2022-03-22_16-31-48.h5"

— Runtime: 0.166 sec. —

Analysing text information (forming model and loading model weights)

Loading a dictionary with hand-crafted features

[15]:
```
# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

res_load_text_features = _b5.load_text_features(
    force_reload = True,       # Forced download file
    out = True,                # Display
    runtime = True,            # Runtime calculation
    run = True                 # Run blocking
)
```

[2023-12-15 07:01:51] Loading a dictionary with hand-crafted features ...

[2023-12-15 07:01:52] Loading the "LIWC2007.txt" file 100.0% ...

— Runtime: 0.166 sec. —

Building tokenizer and translation model (RU -¿ EN)

[16]:
```
res_setup_translation_model = _b5.setup_translation_model(
    out = True,      # Display
    runtime = True, # Runtime calculation
    run = True       # Run blocking
)
```

[2023-12-15 07:01:52] Building tokenizer and translation model ...

— Runtime: 1.763 sec. —

Building tokenizer and BERT model (for word encoding)

[17]:
```
# Core setup
_b5.path_to_save_ = './models' # Directory to save the models
_b5.chunk_size_ = 2000000      # File download size from network in one step

res_setup_translation_model = _b5.setup_bert_encoder(
    force_reload = True,       # Forced download file
    out = True,                # Display
    runtime = True,            # Runtime calculation
    run = True                 # Run blocking
)
```

[2023-12-15 07:01:53] Building tokenizer and BERT model ...

[2023-12-15 07:01:55] Loading the "bert-base-multilingual-cased.zip" file

[2023-12-15 07:01:53] Building tokenizer and BERT model ...

[2023-12-15 07:01:55] Loading the "bert-base-multilingual-cased.zip" file

[2023-12-15 07:01:55] Unzipping an archive "bert-base-multilingual-cased.zip" ...

— Runtime: 5.269 sec. —

Formation of the neural network architecture of the model for obtaining features / scores by hand-crafted features (audio modality)

- `_b5.text_model_hc_` - Neural network model **tf.keras.Model** for obtaining features / scores by hand-crafted features

[18]:
```
res_load_text_model_hc_mupta = _b5.load_text_model_hc(
    corpus = corpus, # Corpus selection for models trained on First Impressions V2 'fi'␣
↪and models trained on for MuPTA 'mupta'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[22023-12-15 07:01:59] Formation of the neural network architecture of the model for obtaining scores by hand-crafted features (text modality) ...

— Runtime: 0.701 sec. —

Downloading the weights of the neural network model to obtain features / scores by hand-crafted features (audio modality)

- `_b5.text_model_hc_` - Neural network model **tf.keras.Model** for obtaining features / scores by hand-crafted features

[19]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['text'][corpus]['hc']['sberdisk']

res_load_text_model_weights_hc_fi = _b5.load_text_model_weights_hc(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,      # Display
    runtime = True, # Runtime count
    run = True       # Run blocking
)
```

[2023-12-15 07:01:59] Downloading the weights of a neural network model to obtain scores by hand-crafted features (text modality) ...

[2023-12-15 07:02:00] File download "weights_2023-07-15_10-52-15.h5" 100.0% ...

— Runtime: 0.278 sec. —

Formation of the neural network architecture of the model for obtaining features / scores by deep features (audio modality)

- `_b5s.text_model_nn_` - Neural network model **tf.keras.Model** for obtaining features / scores by deep features

[20]:
```
res_load_text_model_nn_fi = _b5.load_text_model_nn(
    corpus = corpus, # Corpus selection for models trained on First Impressions V2 'fi'␣
↪and models trained on for MuPTA 'mupta'
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True, # Display
    runtime = True, # Runtime count
    run = True # Run blocking
)
```

[2023-12-15 07:02:00] Formation of a neural network architecture for obtaining scores by deep features (text modality) ...

— Runtime: 0.286 sec. —

Downloading the weights of the neural network model to obtain features / scores for deep features

- `_b5s.text_model_nn_` - Neural network model **tf.keras.Model** for obtaining features / scores by deep features

[21]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['text'][corpus]['nn']['sberdisk']

res_load_text_model_weights_nn_fi = _b5.load_text_model_weights_nn(
    url = url, # Full path to the file with weights of the neural network model
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,      # Display
    runtime = True, # Runtime count
    run = True       # Run blocking
)
```

[2023-12-15 07:02:00] Downloading the weights of a neural network model to obtain deep features (text modality) ...

[2023-12-15 07:02:00] File download "weights_2023-07-03_15-01-08.h5" 100.0% ...

— Runtime: 0.42 sec. —

Analysing multimodal information (forming model, loading model weights, obtaining personality traits scores)

Formation of neural network architectures of models for obtaining the personality traits scores

- `_b5.avt_model_b5_` - Neural network model **tf.keras.Model** for obtaining the personality traits scores

[22]:
```
res_load_avt_model_b5 = _b5.load_avt_model_b5(
    show_summary = False, # Displaying the formed neural network architecture of the model
    out = True,           # Display
    runtime = True,       # Runtime count
    run = True            # Run blocking
)
```

[2023-12-15 07:02:00] Formation of neural network architectures of models for obtaining the personality traits scores (multimodal fusion) ...

— Runtime: 0.212 sec. —

Downloading the weights of neural network models to obtain the personality traits scores (multimodal fusion)

- `_b5.avt_model_b5_` - Neural network model **tf.keras.Model** for obtaining the personality traits scores

[23]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000      # File download size from network in 1 step

url = _b5.weights_for_big5_['avt'][corpus]['b5']['sberdisk']

res_load_avt_model_weights_b5 = _b5.load_avt_model_weights_b5(
    url = url,
    force_reload = True, # Forced download of a file with weights of a neural network␣
↪model from the network
    out = True,          # Display
    runtime = True,      # Runtime count
    run = True           # Run blocking
)
```

[2023-12-15 07:02:01] Downloading the weights of neural network models to obtain the personality traits scores (multimodal fusion) ...

[2023-12-15 07:02:01] File download "avt_fi_2023-12-03_11-36-51.h5"

— Runtime: 0.295 sec. —

Getting scores (multimodal fusion)

- `_b5.df_files_` - DataFrame with data
- `_b5.df_accuracy_` - DataFrame with accuracy

[24]:
```
# Core settings
_b5.path_to_dataset_ = 'E:/Databases/FirstImpressionsV2/test' # Dataset directory
# Directories not included in the selection
_b5.ignore_dirs_ = []
# Key names for DataFrame dataset
_b5.keys_dataset_ = ['Path', 'Openness', 'Conscientiousness', 'Extraversion',
↪'Agreeableness', 'Non-Neuroticism']
_b5.ext_ = ['.mp4'] # Search file extensions

# Full path to the file containing the ground truth scores for the accuracy calculation
url_accuracy = _b5.true_traits_[corpus]['sberdisk']

_b5.get_avt_predictions(
    depth = 1,          # ГHierarchy depth for receiving audio and video data
    recursive = False, # Recursive data search
    sr = 44100,         # Sampling frequency
    window_audio = 2,   # Audio segment window size (in seconds)
    step_audio = 1,     # Audio segment window shift step (in seconds)
    reduction_fps = 5, # Frame rate reduction
    window_video = 10, # Video segment window size (in seconds)
    step_video = 5,     # Video segment window shift step (in seconds)
    asr = False,        # Using a model for ASR
    lang = lang,        # Language selection for models trained on First Impressions V2'en
↪' and models trained on for MuPTA 'ru'
    accuracy = True,    # Accuracy
    url_accuracy = url_accuracy,
    logs = True,        # If necessary, generate a LOG file
    out = True,         # Display
    runtime = True,     # Runtime count
    run = True          # Run blocking
)
```

[2023-12-15 10:22:11] Feature extraction (hand-crafted and deep) from text ...

[2023-12-15 10:22:14] Getting scores and accuracy calculation (multimodal fusion) ...

2000 from 2000 (100.0%) ... test80_25_Q4wOgixh7E.004.mp4 ...

ID	Path	Openness	\
1	E:\Databases\FirstImpressionsV2\test\test80_01...	0.545377	
2	E:\Databases\FirstImpressionsV2\test\test80_01...	0.520572	
3	E:\Databases\FirstImpressionsV2\test\test80_01...	0.450715	
4	E:\Databases\FirstImpressionsV2\test\test80_01...	0.665193	
5	E:\Databases\FirstImpressionsV2\test\test80_01...	0.669463	
6	E:\Databases\FirstImpressionsV2\test\test80_01...	0.632529	
7	E:\Databases\FirstImpressionsV2\test\test80_01...	0.489579	
8	E:\Databases\FirstImpressionsV2\test\test80_01...	0.59544	
9	E:\Databases\FirstImpressionsV2\test\test80_01...	0.559325	
10	E:\Databases\FirstImpressionsV2\test\test80_01...	0.509495	

(continues on next page)

(continued from previous page)

```
11  E:\Databases\FirstImpressionsV2\test\test80_01...  0.599391
12  E:\Databases\FirstImpressionsV2\test\test80_01...  0.458006
13  E:\Databases\FirstImpressionsV2\test\test80_01...  0.377578
14  E:\Databases\FirstImpressionsV2\test\test80_01...  0.563649
15  E:\Databases\FirstImpressionsV2\test\test80_01...    0.7302
16  E:\Databases\FirstImpressionsV2\test\test80_01...  0.620163
17  E:\Databases\FirstImpressionsV2\test\test80_01...  0.603495
18  E:\Databases\FirstImpressionsV2\test\test80_01...  0.543104
19  E:\Databases\FirstImpressionsV2\test\test80_01...  0.624445
20  E:\Databases\FirstImpressionsV2\test\test80_01...  0.658763
21  E:\Databases\FirstImpressionsV2\test\test80_01...  0.562814
22  E:\Databases\FirstImpressionsV2\test\test80_01...  0.472688
23  E:\Databases\FirstImpressionsV2\test\test80_01...   0.43985
24  E:\Databases\FirstImpressionsV2\test\test80_01...  0.638308
25  E:\Databases\FirstImpressionsV2\test\test80_01...  0.506815
26  E:\Databases\FirstImpressionsV2\test\test80_01...  0.517949
27  E:\Databases\FirstImpressionsV2\test\test80_01...  0.570406
28  E:\Databases\FirstImpressionsV2\test\test80_01...  0.637813
29  E:\Databases\FirstImpressionsV2\test\test80_01...  0.572268
30  E:\Databases\FirstImpressionsV2\test\test80_01...  0.658128

   Conscientiousness Extraversion Agreeableness Non-Neuroticism
ID
1           0.523155     0.456685      0.533811        0.516093
2           0.396216     0.478419      0.528622        0.459169
3           0.491121      0.36674      0.510387        0.414304
4           0.648017     0.640581      0.580625        0.596675
5           0.606313     0.619956      0.653291        0.618665
6           0.722035     0.583922       0.63653        0.603358
7           0.453927     0.373339      0.486156        0.421787
8           0.615519     0.514064      0.627394        0.601345
9            0.50692     0.442211      0.537979        0.499341
10          0.526581     0.406979      0.565923         0.54616
11          0.516418     0.516382      0.589003        0.558064
12          0.496319     0.345605       0.48779        0.448027
13          0.410694     0.283698      0.384478        0.313993
14          0.499573     0.445833      0.454925        0.463903
15          0.784698      0.51636      0.698729        0.713016
16          0.564576     0.556421      0.563072        0.543618
17          0.644997     0.440616      0.603712        0.578639
18          0.489751     0.452691      0.566111        0.520961
19          0.574276     0.609165      0.582815        0.560111
20          0.545697     0.627865       0.61989        0.609391
21          0.493076     0.430422      0.539134        0.502142
22          0.417943     0.423233      0.472491        0.392815
23          0.429655     0.319237      0.420569        0.414306
24          0.632067     0.580016      0.642938        0.603159
25           0.57838     0.367448      0.523856        0.481819
26          0.562723     0.383299      0.483178        0.467141
27          0.441804     0.454944      0.530368        0.512669
28          0.611132     0.607629      0.636313        0.620745
29          0.532781     0.504937      0.575169        0.518609
```

(continues on next page)

(continued from previous page)

```
30          0.598394      0.59656      0.621783        0.612908
```

[2023-12-15 10:22:14] Trait-wise accuracy ...

```
          Openness  Conscientiousness  Extraversion  Agreeableness  \
Metrics
MAE         0.0758             0.0716        0.0688         0.0752
Accuracy    0.9242             0.9284        0.9312         0.9248

          Non-Neuroticism    Mean
Metrics
MAE                0.0731  0.0729
Accuracy           0.9269  0.9271
```

[2023-12-15 10:22:14] Mean absolute error: 0.0729, Accuracy: 0.9271 ...

Log files saved successfully ...

— Runtime: 12013.03 sec. —

[24]:
```
True
```

Additional capability

Downloading a file from a URL

Import required packages

[2]:
```
from oceanai.modules.lab.build import Run
```

Build

[3]:
```
_b5 = Run(
    lang = 'en', # Interface language
    color_simple = '#333', # Plain text color (hexadecimal code)
    color_info = '#1776D2', # The color of the text containing the information␣
↪(hexadecimal code)
    color_err = '#FF0000', # Error text color (hexadecimal code)
    color_true = '#008001', # Text color containing positive information (hexadecimal␣
↪code)
    bold_text = True, # Bold text
    num_to_df_display = 30, # Number of rows to display in tables
    text_runtime = 'Runtime', # Runtime text
    metadata = True # Displaying information about library
)
```

[2023-12-10 16:49:03] OCEANAI - personal traits: Authors: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Alexey Karpov [karpov@iias.spb.su] Maintainers: Elena Ryumina [ryumina_ev@mail.ru] Dmitry Ryumin [dl_03.03.1991@mail.ru] Version: 1.0.0a2 License: BSD License

Download process

[4]:
```
# Core settings
_b5.path_to_save_ = './models' # Directory to save the file
_b5.chunk_size_ = 2000000 # File download size from network in 1 step

res_download_file_from_url = _b5.download_file_from_url(
    url = 'https://download.sberdisk.ru/download/file/400635799?token=MMRrak8fMsyzxLE&
↪filename=weights_2022-05-05_11-27-55.h5',
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[2023-12-10 16:49:04] File download "weights_2022-05-05_11-27-55.h5" (100.0%) ...

— Runtime: 0.214 sec. —

[5]:
```
res_download_file_from_url
```

[5]:
```
200
```

4.2.2 Modules

Custom exceptions

exception `oceanai.modules.core.exceptions.`CustomException

Bases: `Exception`

Class for all custom exceptions

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.exceptions import CustomException

message = 'Custom Exception'

try: raise CustomException(message)
except CustomException as ex: print(ex)
```

[1]:
```
Custom Exception
```

exception `oceanai.modules.core.exceptions.`InvalidContentLength

Bases: *CustomException*

Upload file size not defined

Example

True – 1 –

```
In [1]:
from oceanai.modules.core.exceptions import InvalidContentLength

message = 'Upload file size not defined'

try: raise InvalidContentLength(message)
except InvalidContentLength as ex: print(ex)
```

```
[1]:
Upload file size not defined
```

exception oceanai.modules.core.exceptions.IsSmallWindowSizeError

Bases: *CustomException*

Signal segment window size specified too small

Example

True – 1 –

```
In [1]:
from oceanai.modules.core.exceptions import IsSmallWindowSizeError

message = 'Signal segment window size specified too small'

try: raise IsSmallWindowSizeError(message)
except IsSmallWindowSizeError as ex: print(ex)
```

```
[1]:
Signal segment window size specified too small
```

Language detection

class oceanai.modules.core.language.Language(*lang: str = 'ru'*)

Bases: object

Class for internationalization (I18N) and localization (L10N)

Parameters
lang (*str*) – Language

¨get˙languages() → List[Optional[str]]

Get supported languages

Note: private method

Returns
List of supported languages

Return type
List[Optional[str]]

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'en')
language._Language__get_languages()
```

[1]:
```
['ru', 'en']
```

¨get˙locales() → Dict[str, method]

Get language packs

Note: private method

Returns
Dictionary with language packs

Return type
Dict[str, MethodType]

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'en')
language._Language__get_locales()
```

[1]:
```
{
    'ru': <bound method GNUTranslations.gettext of <gettext.GNUTranslations␣
↪object at 0x14680ce50>>,
    'en': <bound method GNUTranslations.gettext of <gettext.GNUTranslations␣
↪object at 0x1460ddbb0>>
}
```

¨set˙locale(*lang: str = ''*) → method

Language setting

Note: private method

Parameters
lang (*str*) – Language

Returns
MethodType of translating strings into one of the supported languages if the method is launched via the constructor

Return type
MethodType

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'ru')
print(language.lang_)
```

[1]:
```
ru
```

– 2 –

In [2]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'ru')
language._Language__set_locale('en')
print(language.lang_)
```

[2]:
```
en
```

lang: str = 'ru'

Language options available:

- "ru" - Russian language (default)
- "en" - English language

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.language import Language

language = Language()
print(language.lang, language.lang_)
```

[1]:
```
ru ru
```

– 2 –

In [2]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'ru')
print(language.lang, language.lang_)
```

[2]:

```
ru ru
```

– 3 –

In [3]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'en')
print(language.lang, language.lang_)
```

[3]:
```
en en
```

Better not to do that – 1 –

In [4]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'es')
print(language.lang, language.lang_)
```

[4]:
```
es ru
```

– 2 –

In [5]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 1)
print(language.lang, language.lang_)
```

[5]:
```
1 ru
```

Type
str

property lang': str

Getting the current language

Returns
Language

Return type
str

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.language import Language

language = Language()
print(language.lang_)
```

[1]:

```
ru
```

– 2 –

In [2]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'ru')
print(language.lang_)
```

[2]:
```
ru
```

– 3 –

In [3]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'en')
print(language.lang_)
```

[3]:
```
en
```

Better not to do that – 1 –

In [4]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'es')
print(language.lang_)
```

[4]:
```
ru
```

– 2 –

In [5]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 1)
print(language.lang_)
```

[5]:
```
ru
```

property locales˙: List[str]

Get supported languages

Returns
List of supported languages

Return type
List[str]

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'en')
print(language.locales_)
```

[1]:
```
['ru', 'en']
```

property path_to_locales: str

Get directory with language packs

Returns
Directory with language packs

Return type
str

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.language import Language

language = Language(lang = 'en')
# Each user has their own path
print(language.path_to_locales_)
```

[1]:
```
/Users/dl/GitHub/OCEANAI/oceanai/modules/locales
```

Messages

class oceanai.modules.core.messages.Messages(*lang: str = 'ru'*)

Bases: *Language*

Class for messages

Parameters
lang (*str*) – See *lang*

Settings

class oceanai.modules.core.settings.Settings(*lang: str = 'ru'*, *color_simple: str = '#666'*, *color_info: str = '#1776D2'*, *color_err: str = '#FF0000'*, *color_true: str = '#008001'*, *bold_text: bool = True*, *text_runtime: str = ''*, *num_to_df_display: int = 30*)

Bases: *Messages*

Class for settings

Parameters

- lang (*str*) – See *lang*
- color_simple (*str*) – Plain text color (hexadecimal code)
- color_info (*str*) – The color of the text containing the information (hexadecimal code)
- color_err (*str*) – Error text color (hexadecimal code)
- color_true (*str*) – Text color containing positive information (hexadecimal code)
- bold_text (*bool*) – Bold text
- num_to_df_display (*int*) – Number of rows to display in tables
- text_runtime (*str*) – Runtime text

bold_text: bool = True

Bold text

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en')
print(settings.bold_text, settings.bold_text_)
```

[1]:
```
True True
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', bold_text = True)
print(settings.bold_text, settings.bold_text_)
```

[2]:
```
True True
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', bold_text = False)
print(settings.bold_text, settings.bold_text_)
```

[3]:
```
False False
```

Better not to do that – 1 –

In [4]:

```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', bold_text = 1)
print(settings.bold_text, settings.bold_text_)
```

[4]:

```
True True
```

– 2 –

In [5]:

```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', bold_text = 'some_text')
print(settings.bold_text, settings.bold_text_)
```

[5]:

```
True True
```

Type
bool

property bold_text: bool

Getting and setting bold text

Parameters
(bool) – **True** or **False**

Returns
True or **False**

Return type
bool

Examples

True – 1 –

In [1]:

```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', bold_text = True)
print(settings.bold_text_)
```

[1]:

```
True
```

– 2 –

In [2]:

```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', bold_text = True)
settings.bold_text_ = False
print(settings.bold_text_)
```

[2]:

```
False
```

Better not to do that – 1 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', bold_text = False)
settings.bold_text_ = 1
print(settings.bold_text_)
```

[3]:
```
False
```

– 2 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', bold_text = True)
settings.bold_text_ = 'some_text'
print(settings.bold_text_)
```

[4]:
```
True
```

property chunk˙size˙: int

Getting/setting file download size from network in 1 step

Parameters
(int) – File download size from network in 1 step

Returns
File download size from network in 1 step

Return type
int

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.chunk_size_)
```

[1]:
```
1000000
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.chunk_size_ = 2000000
print(settings.chunk_size_)
```

[2]:
```
2000000
```

Better not to do that – 1 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.chunk_size_ = -1
print(settings.chunk_size_)
```

[3]:
```
1000000
```

– 2 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.chunk_size_ = False
print(settings.chunk_size_)
```

[4]:
```
1000000
```

– 3 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.chunk_size_ = 'some_text'
print(settings.chunk_size_)
```

[5]:
```
1000000
```

color˙err: str = '#FF0000'

Error text color (hexadecimal code)

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.color_err, settings.color_err_)
```

[1]:
```
#FF0000 #FF0000
```

– 2 –

In [2]:

```
from oceanai.modules.core.settings import Settings

settings = Settings(color_err = 'FF0000')
print(settings.color_err, settings.color_err_)
```

[2]:
```
#FF0000 #FF0000
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_err = '#FF0')
print(settings.color_err, settings.color_err_)
```

[3]:
```
#FF0 #FF0
```

Better not to do that – 1 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_err = 1)
print(settings.color_err, settings.color_err_)
```

[4]:
```
#FF0000 #FF0000
```

– 2 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_err = [])
print(settings.color_err, settings.color_err_)
```

[5]:
```
#FF0000 #FF0000
```

Type
str

property color˙err˙: str

Getting/setting the color of the text containing the error

Parameters
(str) – Hex code

Returns
Hex code

Return type
str

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_err = '#C22931')
print(settings.color_err_)
```

[1]:
```
#C22931
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_err_ = '#FF0'
print(settings.color_err_)
```

[2]:
```
#FF0
```

Better not to do that – 1 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_err_ = 1
print(settings.color_err_)
```

[3]:
```
#FF0000
```

– 2 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_err_ = {}
print(settings.color_err_)
```

[4]:
```
#FF0000
```

color˙info: str = '#1776D2'

The color of the text containing the information (hexadecimal code)

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.color_info, settings.color_info_)
```

[1]:
```
#1776D2 #1776D2
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_info = '#1776D2')
print(settings.color_info, settings.color_info_)
```

[2]:
```
#1776D2 #1776D2
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_info = '#42F')
print(settings.color_info, settings.color_info_)
```

[3]:
```
#42F #42F
```

Better not to do that – 1 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_info = 1)
print(settings.color_info, settings.color_info_)
```

[4]:
```
#1776D2 #1776D2
```

– 2 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_info = [])
print(settings.color_info, settings.color_info_)
```

[5]:
```
#1776D2 #1776D2
```

Type
str

property color˙info˙: str

Getting/setting the color of the text containing the information

Parameters
(str) – Hex code

Returns
Hex code

Return type
str

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_info = '#1776D2')
print(settings.color_info_)
```

[1]:
```
#1776D2
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_info_ = '#42F'
print(settings.color_info_)
```

[2]:
```
#42F
```

Better not to do that – 1 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_info_ = 1
print(settings.color_info_)
```

[3]:
```
#1776D2
```

– 2 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_info_ = ()
print(settings.color_info_)
```

[4]:

```
#1776D2
```

color_simple: str = '#666'

Plain text color (hexadecimal code)

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.color_simple, settings.color_simple_)
```

[1]:
```
#666 #666
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_simple = '#666')
print(settings.color_simple, settings.color_simple_)
```

[2]:
```
#666 #666
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_simple = '#222')
print(settings.color_simple, settings.color_simple_)
```

[3]:
```
#222 #222
```

Better not to do that – 1 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_simple = 1)
print(settings.color_simple, settings.color_simple_)
```

[4]:
```
#666 #666
```

– 2 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_simple = {1, 2, 3})
print(settings.color_simple, settings.color_simple_)
```

[5]:
```
#666 #666
```

Type
str

property color˙simple˙: str

Getting/setting plain text color

Parameters
(str) – Hex code

Returns
Hex code

Return type
str

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_simple = '#111')
print(settings.color_simple_)
```

[1]:
```
#111
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_simple_ = '#444'
print(settings.color_simple_)
```

[2]:
```
#444
```

Better not to do that – 1 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_simple_ = 1
print(settings.color_simple_)
```

[3]:
```
#666
```

– 2 –

In [4]:

```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_simple_ = ()
print(settings.color_simple_)
```

[4]:
```
#666
```

color˙true: str = '#008001'

Text color containing positive information (hexadecimal code)

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.color_true, settings.color_true_)
```

[1]:
```
#008001 #008001
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_true = '#003332')
print(settings.color_true, settings.color_true_)
```

[2]:
```
#003332 #003332
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_true = '#42F')
print(settings.color_true, settings.color_true_)
```

[3]:
```
#42F #42F
```

Better not to do that – 1 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_true = 1)
print(settings.color_true, settings.color_true_)
```

[4]:
```
#008001 #008001
```

– 2 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_true = [])
print(settings.color_true, settings.color_true_)
```

[5]:
```
#008001 #008001
```

Type
str

property color˙true˙: str

Getting/setting the color of text containing positive information

Parameters
(str) – Hex code

Returns
Hex code

Return type
str

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(color_true = '#008001')
print(settings.color_true_)
```

[1]:
```
#008001
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_true_ = '#42F'
print(settings.color_true_)
```

[2]:
```
#42F
```

Better not to do that – 1 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_true = 1
print(settings.color_true)
```

[3]:
```
#008001
```

– 2 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.color_true_ = ()
print(settings.color_true_)
```

[4]:
```
#008001
```

property ext˙: List[str]

Getting/installing the extensions of searched files

Parameters
(List[str]) – List with search file extensions

Returns
List with search file extensions

Return type
List[str]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.ext_)
```

[1]:
```
[]
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.ext_ = ['.mp4']
print(settings.ext_)
```

[2]:
```
['.mp4']
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.ext_ = ['.mp3', '.wav']
print(settings.ext_)
```

[3]:
```
['.mp3', '.wav']
```

– 4 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.ext_ = []
print(settings.ext_)
```

[4]:
```
[]
```

Better not to do that – 1 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.ext_ = [2, []]
print(settings.ext_)
```

[5]:
```
[]
```

property ignore˙dirs˙: List[str]

Getting/installing a list with directories not included in the selection

Parameters
(List[str]) – List with directories

Returns
List with directories

Return type
List[str]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.ignore_dirs_)
```

[1]:
```
[]
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.ignore_dirs_ = ['test', 'test_2']
print(settings.ignore_dirs_)
```

[2]:
```
['test', 'test_2']
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.ignore_dirs_ = []
print(settings.ignore_dirs_)
```

[3]:
```
[]
```

– 4 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.ext_ = ['1_a', '2_b']
print(settings.ext_)
```

[4]:
```
['1_a', '2_b']
```

Better not to do that – 1 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.ignore_dirs_ = [2, []]
print(settings.ignore_dirs_)
```

[5]:
```
[]
```

property keys˙dataset˙

Getting/setting dataset key names

Parameters
(List[str]) – List with dataset key names

Returns
List with dataset key names

Return type
List[str]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.keys_dataset_)
```

[1]:
```
[
    'Path',
    'Openness',
    'Conscientiousness',
    'Extraversion',
    'Agreeableness',
    'Non-Neuroticism'
]
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.keys_dataset_ = ['P', 'O', 'C', 'E', 'A', 'N']
print(settings.keys_dataset_)
```

[2]:
```
['P', 'O', 'C', 'E', 'A', 'N']
```

Better not to do that – 1 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.keys_dataset_ = [{}, [], 1]
print(settings.keys_dataset_)
```

[3]:
```
[
    'Path',
    'Openness',
    'Conscientiousness',
    'Extraversion',
    'Agreeableness',
    'Non-Neuroticism'
]
```

– 2 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.keys_dataset_ = ['P', 'O']
print(settings.keys_dataset_)
```

[4]:
```
[
    'Path',
    'Openness',
    'Conscientiousness',
    'Extraversion',
    'Agreeableness',
    'Non-Neuroticism'
]
```

– 3 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.keys_dataset_ = []
print(settings.keys_dataset_)
```

[5]:
```
[
    'Path',
    'Openness',
    'Conscientiousness',
    'Extraversion',
    'Agreeableness',
    'Non-Neuroticism'
]
```

num_to_df_display: int = 30

Number of rows to display in tables

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.num_to_df_display, settings.num_to_df_display_)
```

[1]:
```
30 30
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(num_to_df_display = 30)
print(settings.num_to_df_display, settings.num_to_df_display_)
```

[2]:
```
30 30
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(num_to_df_display = 50)
print(settings.num_to_df_display, settings.num_to_df_display_)
```

[3]:
```
50 50
```

Better not to do that – 1 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(num_to_df_display = 0)
print(settings.num_to_df_display, settings.num_to_df_display_)
```

[4]:
```
30 30
```

– 2 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(num_to_df_display = 'some_text')
print(settings.num_to_df_display, settings.num_to_df_display_)
```

[5]:
```
30 30
```

Type
int

property num_to_df_display_: int

Getting/setting the number of rows to display in tables

Parameters
(int) – Number of lines

Returns
Number of lines

Return type
int

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(num_to_df_display = 30)
print(settings.num_to_df_display_)
```

[1]:
```
30
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.num_to_df_display_ = 50
print(settings.num_to_df_display_)
```

[2]:
```
50
```

Better not to do that – 1 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.num_to_df_display_ = 0
print(settings.num_to_df_display_)
```

[3]:
```
30
```

– 2 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.num_to_df_display_ = ()
print(settings.num_to_df_display_)
```

[4]:
```
30
```

property path˙to˙dataset˙: str

Getting/setting the dataset directory

Parameters
(str) – Dataset directory

Returns
Dataset directory

Return type
str

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.path_to_dataset_)
```

[1]:

```
.
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_dataset_ = './dataset'
print(settings.path_to_dataset_)
```

[2]:
```
dataset
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_dataset_ = ''
print(settings.path_to_dataset_)
```

[3]:
```
.
```

Better not to do that – 1 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_dataset_ = [2, []]
print(settings.path_to_dataset_)
```

[4]:
```
.
```

– 2 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_dataset_ = 1
print(settings.path_to_dataset_)
```

[5]:
```
.
```

property path˙to˙logs˙: str

Getting/setting directory for saving LOG files

Parameters
(str) – Directory for saving LOG files

Returns
Directory for saving LOG files

Return type
str

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.path_to_logs_)
```

[1]:
```
logs
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_logs_ = './logs/DF'
print(settings.path_to_logs_)
```

[2]:
```
logs/DF
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_logs_ = ''
print(settings.path_to_logs_)
```

[3]:
```
.
```

Better not to do that – 1 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_logs_ = [2, []]
print(settings.path_to_logs_)
```

[4]:
```
logs
```

– 2 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
```

(continues on next page)

(continued from previous page)

```
settings.path_to_logs_ = {'a': 1, 'b': 2}
print(settings.path_to_logs_)
```

[5]:

```
logs
```

property path˙to˙save˙: str

Getting/setting directory to save data

Parameters
(str) – Directory for saving data

Returns
Directory for saving data

Return type
str

Examples

True – 1 –

In [1]:

```
from oceanai.modules.core.settings import Settings

settings = Settings()
print(settings.path_to_save_)
```

[1]:

```
models
```

– 2 –

In [2]:

```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_save_ = './models/Audio'
print(settings.path_to_save_)
```

[2]:

```
models/Audio
```

– 3 –

In [3]:

```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_save_ = ''
print(settings.path_to_save_)
```

[3]:

```
.
```

Better not to do that – 1 –

In [4]:

```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_save_ = [2, []]
print(settings.path_to_save_)
```

[4]:
```
models
```

– 2 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings()
settings.path_to_save_ = {'a': 1, 'b': 2}
print(settings.path_to_save_)
```

[5]:
```
models
```

text˙runtime: str = ''

Runtime text

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en')
print(settings.text_runtime, settings.text_runtime_)
```

[1]:
```
Runtime Runtime
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', text_runtime = 'Code executed in')
print(settings.text_runtime, settings.text_runtime_)
```

[2]:
```
Code executed in Code executed in
```

– 3 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', text_runtime = 'Runtime')
print(settings.text_runtime, settings.text_runtime_)
```

[3]:

```
Runtime Runtime
```

Better not to do that – 1 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', text_runtime = 1)
print(settings.text_runtime, settings.text_runtime_)
```

[4]:
```
Runtime Runtime
```

– 2 –

In [5]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', text_runtime = {1, 2, 3})
print(settings.text_runtime, settings.text_runtime_)
```

[5]:
```
Runtime Runtime
```

Type
str

property text˙runtime˙: str

Getting/setting the runtime text

Parameters
(str) – Text

Returns
Text

Return type
str

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en', text_runtime = 'Runtime')
print(settings.text_runtime_)
```

[1]:
```
Runtime
```

– 2 –

In [2]:
```
from oceanai.modules.core.settings import Settings

```

(continues on next page)

(continued from previous page)

```
settings = Settings(lang = 'en')
settings.text_runtime_ = 'Code executed in'
print(settings.text_runtime_)
```

[2]:
```
Code executed in
```

Better not to do that – 1 –

In [3]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en')
settings.text_runtime_ = 1
print(settings.text_runtime_)
```

[3]:
```
Runtime
```

– 2 –

In [4]:
```
from oceanai.modules.core.settings import Settings

settings = Settings(lang = 'en')
settings.text_runtime_ = ()
print(settings.text_runtime_)
```

[4]:
```
Runtime
```

Core

class oceanai.modules.core.core.CoreMessages(*lang: str = 'ru', color_simple: str = '#666', color_info: str = '#1776D2', color_err: str = '#FF0000', color_true: str = '#008001', bold_text: bool = True, text_runtime: str = '', num_to_df_display: int = 30*)

Bases: *Settings*

Class for messages

Parameters

- lang (***str***) – See *lang*
- color˙simple (***str***) – See *color_simple*
- color˙info (***str***) – See *color_info*
- color˙err (***str***) – See *color_err*
- color˙true (***str***) – See *color_true*
- bold˙text (***bool***) – See *bold_text*
- num˙to˙df˙display (***int***) – See *num_to_df_display*
- text˙runtime (***str***) – See *text_runtime*

class oceanai.modules.core.core.Core(*lang: str = 'ru', color_simple: str = '#666', color_info: str = '#1776D2', color_err: str = '#FF0000', color_true: str = '#008001', bold_text: bool = True, text_runtime: str = '', num_to_df_display: int = 30*)

Bases: *CoreMessages*

Core class of modules

Parameters

- lang (***str***) – See *lang*
- color˙simple (***str***) – See *color_simple*
- color˙info (***str***) – See *color_info*
- color˙err (***str***) – See *color_err*
- color˙true (***str***) – See *color_true*
- bold˙text (***bool***) – See *bold_text*
- num˙to˙df˙display (***int***) – See *num_to_df_display*
- text˙runtime (***str***) – See *text_runtime*

¨is˙notebook() → bool

Determining how to run a library in Jupyter or similar

Note: private method

Returns

True if the library is run in Jupyter or similar, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()
core._Core__is_notebook()
```

[1]:
```
True
```

– 2 –

In [2]:
```
from oceanai.modules.core.core import Core

Core._Core__is_notebook()
```

[2]:
```
True
```

˙add˙last˙el˙notebook˙history˙output(*message: str*) → None

Adding text to the latest message from the message output history in a Jupyter cell

---

Note: protected method

Parameters
message (*str*) – Message

Returns
None

Return type
None

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._add_last_el_notebook_history_output(message = '...')

core._add_notebook_history_output(
    message = 'Message 1', last = False
)
core._add_last_el_notebook_history_output(message = '...')

core.show_notebook_history_output()
```

[1]:
```
...
Message 1 ...
```

˙add˙notebook˙history˙output(*message: str*, *last: bool = False*) → None

Adding message output history to Jupyter cell

Note: protected method

Parameters

- message (*str*) – Message
- last (*bool*) – Replacing the last message

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._add_notebook_history_output(
    message = 'Message 1', last = False
)
core._add_notebook_history_output(
    message = 'Message 2', last = False
)
core._add_notebook_history_output(
    message = 'Replacing the last message', last = True
)

core.show_notebook_history_output()
```

[1]:
```
Message 1
Replacing the last message
```

– 2 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

for message, last in zip(
    [
        'Message 1',
        'Message 2',
        'Replacing the last message'
    ],
    [False, False, True]
):
    core._add_notebook_history_output(
        message = message, last = last
    )

core.show_notebook_history_output()
```

[2]:
```
Message 1
Replacing the last message
```

append to list of accuracy(*preds: List[Optional[float]]*, *out: bool = True*) → bool

Adding values to the dictionary for a DataFrame with precision results

Note: protected method

Parameters

- preds (*List[Optional[float]]*) – Personality traits scores
- out (*bool*) – Display

Returns

True if values have been added to the dictionary for the DataFrame, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()

core.keys_dataset_ = ['O', 'C', 'E', 'A', 'N']

core._append_to_list_of_accuracy(
    preds = [0.5, 0.6, 0.2, 0.1, 0.8],
    out = True
)

core._append_to_list_of_accuracy(
    preds = [0.4, 0.5, 0.1, 0, 0.7],
    out = True
)

core.dict_of_accuracy_
```

[1]:
```
{
    'O': [0.5, 0.4],
    'C': [0.6, 0.5],
    'E': [0.2, 0.1],
    'A': [0.1, 0],
    'N': [0.8, 0.7]
}
```

Error – 1 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core()

core.keys_dataset_ = ['O', 'C', 'E', 'A', 'N']

core._append_to_list_of_accuracy(
    preds = [0.5, 0.6, 0.2, 0.1, 0.8],
    out = True
)
```

(continues on next page)

(continued from previous page)

```
core.keys_dataset_ = ['O2', 'C2', 'E2', 'A2', 'N2']

core._append_to_list_of_accuracy(
    preds = [0.4, 0.5, 0.1, 0, 0.7],
    out = True
)

core.dict_of_accuracy_
```

– 2 –

[2]:

```
[2022-12-03 23:08:15] Ouch! Something went wrong ... see kernel settings  and␣
↪chain of action ...

    File: /Users/dl/GitHub/OCEANAI.private/oceanai/modules/core/core.py
    Line: 2669
    Method: _append_to_list_of_accuracy
    Error type: KeyError

{
    'O': [0.5, 0.4],
    'C': [0.6, 0.5],
    'E': [0.2, 0.1],
    'A': [0.1, 0],
    'N': [0.8, 0.7]
}
```

append_to_list_of_files(*path: str*, *preds: List[Optional[float]]*, *out: bool = True*) → bool

Adding values to a dictionary for a DataFrame with data

Note: protected method

Parameters

- path (*str*) – The path to the file
- preds (*List[Optional[float]]*) – Personality traits scores
- out (*bool*) – Display

Returns

True if values have been added to the dictionary for the DataFrame, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()

core.keys_dataset_ = ['P', 'O', 'C', 'E', 'A', 'N']

core._append_to_list_of_files(
    path = './6V807Mf_gHM.003.mp4',
    preds = [0.5, 0.6, 0.2, 0.1, 0.8],
    out = True
)

core._append_to_list_of_files(
    path = './6V807Mf_gHM.004.mp4',
    preds = [0.4, 0.5, 0.1, 0, 0.7],
    out = True
)

core.dict_of_files_
```

[1]:
```
{
    'P': ['./6V807Mf_gHM.003.mp4', './6V807Mf_gHM.004.mp4'],
    'O': [0.5, 0.4],
    'C': [0.6, 0.5],
    'E': [0.2, 0.1],
    'A': [0.1, 0],
    'N': [0.8, 0.7]
}
```

Error – 1 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core.keys_dataset_ = ['P', 'O', 'C', 'E', 'A', 'N']

core._append_to_list_of_files(
    path = './6V807Mf_gHM.003.mp4',
    preds = [0.5, 0.6, 0.2, 0.1, 0.8],
    out = True
)

core.keys_dataset_ = ['P2', 'O2', 'C2', 'E2', 'A2', 'N2']

core._append_to_list_of_files(
    path = './6V807Mf_gHM.004.mp4',
    preds = [0.4, 0.5, 0.1, 0, 0.7],
```

(continues on next page)

(continued from previous page)

```
    out = True
)

core.dict_of_files_
```

– 2 –

[2]:

```
[2022-10-13 18:22:40] Ouch! Something went wrong ... see kernel settings and␣
↪chain of action ...

    File: /Users/dl/GitHub/oceanai/oceanai/modules/core/core.py
    Line: 1105
    Method: _append_to_list_of_files
    Error type: KeyError

{
    'P': ['./6V807Mf_gHM.003.mp4'],
    'O': [0.5],
    'C': [0.6],
    'E': [0.2],
    'A': [0.1],
    'N': [0.8]
}
```

˙bold˙wrapper(*message: str*) → str

Wrapped message with bold text

Note: protected method

Parameters
message (*str*) – Message

Returns
Wrapped message with bold text

Return type
str

Example

True – 1 –

In [1]:

```
from oceanai.modules.core.core import Core

core = Core(lang = 'en', bold_text = False)
print(core._bold_wrapper(
    'Wrapped message without bold text'
))

core.bold_text = True
```

(continues on next page)

(continued from previous page)

```
print(core._bold_wrapper(
    'Wrapped message with bold text'
))
```

[1]:

```
<span style="color:#FF0000">Wrapped message without bold text</span>
<span style="color:#FF4545">Wrapped message with bold text</span>
```

candidate_ranking(*df_files: Optional[DataFrame] = None*, *weigths_openness: int = 0*, *weigths_conscientiousness: int = 0*, *weigths_extraversion: int = 0*, *weigths_agreeableness: int = 0*, *weigths_non_neuroticism: int = 0*, *out: bool = True*) → DataFrame

Ranking candidates by professional responsibilities

Note: protected method

Parameters

- df_files (*pd.DataFrame*) – **DataFrame** with data
- weigths_openness (*int*) – Weight for ranking personality trait (openness)
- weigths_conscientiousness (*int*) – Weight for ranking personality trait (conscientiousness)
- weigths_extraversion (*int*) – Weight for ranking personality trait (extraversion)
- weigths_agreeableness (*int*) – Weight for ranking personality trait (agreeableness)
- weigths_non_neuroticism (*int*) – Weight for ranking personality trait (non-neuroticism)
- out (*bool*) – Display

Returns
DataFrame with ranking data

Return type
pd.DataFrame

clear_notebook_history_output() → None

Clearing message output history in a Jupyter cell

Note: protected method

Returns
None

Return type
None

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._add_notebook_history_output(
    message = 'Message 1', last = False
)
core._add_notebook_history_output(
    message = 'Message 2', last = False
)

core._clear_notebook_history_output()

core.show_notebook_history_output()
```

[1]:
```

```

˙colleague˙personality˙desorders(*df_files: Optional[DataFrame] = None*, *correlation_coefficients_mbti: Optional[DataFrame] = None*, *correlation_coefficients_disorders: Optional[DataFrame] = None*, *personality_desorder_number: int = 3*, *col_name_ocean: str = 'Trait'*, *threshold: float = 0.55*, *out: bool = True*) → DataFrame

Определение приоритетных профессиональных растройств по версии MBTI

Note: protected method

Parameters

- df˙files (*pd.DataFrame*) – **DataFrame** with data
- correlation˙coefficients˙mbti (*pd.DataFrame*) – **DataFrame** с коэффициентами корреляции для MBTI
- correlation˙coefficients˙disorders (*pd.DataFrame*) – **DataFrame** с коэффициентами корреляции для расстройств
- target˙scores (*List[float]*) – List with the names of personality traits scores
- personality˙desorder˙number (*int*) – Количество приоритетных расстройств
- threshold (*float*) – Threshold for scores of traits polarity (e.g., introvert < 0.55, extrovert > 0.55)
- out (*bool*) – Display
- col˙name˙ocean (*str*) –

Returns

DataFrame с приоритетными расстройствами

Return type

pd.DataFrame

˙colleague˙personality˙type˙match(*df_files: Optional[DataFrame] = None*, *correlation_coefficients: Optional[DataFrame] = None*, *target_scores: List[float] = [0.47, 0.63, 0.35, 0.58, 0.51]*, *col_name_ocean: str = 'Trait'*, *threshold: float = 0.55*, *out: bool = True*) → DataFrame

Поиск коллег по совместимости персональных типов по версии MBTI

Note: protected method

Parameters

- df˙files (*pd.DataFrame*) – **DataFrame** with data
- correlation˙coefficients (*pd.DataFrame*) – **DataFrame** with correlation coefficients
- target˙scores (*List[float]*) – List with the names of personality traits scores
- threshold (*float*) – Threshold for scores of traits polarity (e.g., introvert < 0.55, extrovert > 0.55)
- out (*bool*) – Display
- col˙name˙ocean (*str*) –

Returns

DataFrame с совместимостью коллег по персональным типам по версии MBTI

Return type

pd.DataFrame

˙colleague˙ranking(*df_files: Optional[DataFrame] = None*, *correlation_coefficients: Optional[DataFrame] = None*, *target_scores: List[float] = [0.47, 0.63, 0.35, 0.58, 0.51]*, *colleague: str = 'major'*, *equal_coefficients: float = 0.5*, *out: bool = True*) → DataFrame

Finding a suitable colleague

Note: protected method

Parameters

- df˙files (*pd.DataFrame*) – **DataFrame** with data
- correlation˙coefficients (*pd.DataFrame*) – **DataFrame** with correlation coefficients
- target˙scores (*List[float]*) – List with the names of personality traits scores
- colleague (*str*) – Rank of compatibility colleague
- equal˙coefficients (*float*) – Coefficient applied to scores in case of equality of scores of two people
- out (*bool*) – Display

Returns

DataFrame with ranked colleagues

Return type

pd.DataFrame

˙compatibility˙percentage(*type1*, *type2*)

˙create˙folder˙for˙logs(*out: bool = True*)

Creating a directory for saving LOG files

Note: protected method

Parameters
out (*bool*) – Display

Returns
True if the directory is created or exists, otherwise **False**

Return type
bool

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()

core.path_to_logs_ = './logs'

core._create_folder_for_logs(out = True)
```

[1]:
```
true
```

˙del˙last˙el˙notebook˙history˙output() → None

Removing the last message from the message output history in a Jupyter cell

---

Note: protected method

Returns
None

Return type
None

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._add_notebook_history_output(
    message = 'Message 1', last = False
)
core._add_notebook_history_output(
    message = 'Message 2', last = False
)

core._del_last_el_notebook_history_output()

core.show_notebook_history_output()
```

[1]:
```
Message 1
```

error(*message: str*, *last: bool = False*, *out: bool = True*) → None

Error message

Note: protected method

Parameters

- message (*str*) – Message
- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._error(
    message = 'Error message 1',
    last = False, out = True
)
```

(continues on next page)

(continued from previous page)

```
core.color_simple_ = '#FFF'
core.color_err_ = 'FF0000'
core.bold_text_ = False

core._error(
    message = 'Error message 2',
    last = True, out = True
)
```

[1]:
```
[2022-10-12 15:21:00] Error message 1
[2022-10-12 15:21:00] Error message 2
```

Error – 1 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._error(
    message = '',
    last = False, out = True
)
```

[2]:
```
[2022-10-12 17:06:04] Invalid argument types or values in "Core._error" ...
```

˙error˙wrapper(*message: str*) → str

Wrapped error message

Note: protected method

Parameters
message (*str*) – Message

Returns
Wrapped error message

Return type
str

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
print(core._error_wrapper(
    'Wrapped error message 1'
```

(continues on next page)

(continued from previous page)

```
))

core.color_err_ = '#FF4545'
print(core._error_wrapper(
    'Wrapped error message 2'
))
```

[1]:

```
<span style="color:#FF0000">Wrapped error message 1</span>
<span style="color:#FF4545">Wrapped error message 2</span>
```

get_paths(*path: Iterable*, *depth: int = 1*, *out: bool = True*) → Union[List[str], bool]

Getting directories where data is stored

Note: protected method

Parameters

- path (*Iterable*) – Dataset directory
- depth (*int*) – Hierarchy depth for class extraction
- out (*bool*) – Display

Returns

False if the argument check fails or a list of directories

Return type

Union[List[str], bool]

Examples

True – 1 –

In [1]:

```
core = Core()
core._get_paths(
    path = '/Users/dl/GitHub/oceanai/oceanai/dataset',
    depth = 1, out = True
)
```

[1]:

```
[
    '/Users/dl/GitHub/oceanai/oceanai/dataset/test80_01',
    '/Users/dl/GitHub/oceanai/oceanai/dataset/1',
    '/Users/dl/GitHub/oceanai/oceanai/dataset/test80_17'
]
```

Errors – 1 –

In [2]:

```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
```

(continues on next page)

(continued from previous page)

```
core._get_paths(
    path = '',
    depth = 1, out = True
)
```

[2]:

```
[2022-10-12 16:36:16] Invalid argument types or values in "Core._get_paths" ...
False
```

– 2 –

In [3]:

```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core._get_paths(
    path = '/Users/dl/GitHub/oceanai/oceanai/folder',
    depth = 1, out = True
)
```

[3]:

```
[2022-10-13 18:37:11] Ouch! Something went wrong ... directory "/Users/dl/
↪GitHub/oceanai/oceanai/folder" not found ...

    File: /Users/dl/GitHub/oceanai/oceanai/modules/core/core.py
    Line: 1023
    Method: _get_paths
    Error type: FileNotFoundError

False
```

˙info(*message: str*, *last: bool = False*, *out: bool = True*) → None

Announcement

Note: protected method

Parameters

- message (*str*) – Message
- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns

None

Return type

None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._info(
    message = 'Announcement 1',
    last = False, out = True
)

core.color_simple_ = '#FFF'
core.color_info_ = '#0B45B9'
core.bold_text_ = False

core._info(
    message = 'Announcement 2',
    last = True, out = True
)
```

[1]:
```
[2022-10-14 11:35:00] Announcement 1
[2022-10-14 11:35:00] Announcement 2
```

Error – 1 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._info(
    message = '',
    last = False, out = True
)
```

[2]:
```
[2022-10-14 11:43:00] Invalid argument types or values in "Core._info" ...
```

˙info˙true(*message: str*, *last: bool = False*, *out: bool = True*) → None

True information

Note: protected method

Parameters

- message (*str*) – Message
- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns

None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._info_true(
    message = 'Informational true message 1',
    last = False, out = True
)

core.color_true_ = '#008001'
core.bold_text_ = False

core._info_true(
    message = 'Informational true message 2',
    last = True, out = True
)
```

[1]:
```
Informational true message 1

Informational true message 2
```

Error – 1 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._info_true(
    message = '',
    last = False, out = True
)
```

[2]:
```
[2022-10-22 16:46:56] Invalid argument types or values in "Core._info_true" ...
```

˙info˙wrapper(*message: str*) → str
Wrapped announcement

Note: protected method

Parameters
message (*str*) – Message

Returns
Wrapped announcement

Return type
str

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
print(core._info_wrapper('Wrapped announcement 1'))

core.color_info_ = '#0B45B9'
print(core._info_wrapper('Wrapped announcement 2'))
```

[1]:
```
<span style="color:#1776D2">Wrapped announcement 1</span>
<span style="color:#0B45B9">Wrapped announcement 2</span>
```

˙inv˙args(*class_ name: str*, *build_ name: str*, *last: bool = False*, *out: bool = True*) → None
Message about specifying invalid argument types

Note: protected method

Parameters

- class˙name (*str*) – Class name
- build˙name (*str*) – Function method/name
- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core._inv_args(
    Core.__name__, core._info.__name__,
    last = False, out = True
)
```

[1]:

```
[2022-10-14 11:58:04] Invalid argument types or values in "Core._info" ...
```

Error – 1 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core._inv_args(1, '', last = False, out = True)
```

[2]:
```
[2022-10-14 11:58:04] Invalid argument types or values in "Core._inv_args" ...
```

˙metadata˙info(*last: bool = False*, *out: bool = True*) → None

Library Information

Note: protected method

Parameters

- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core._metadata_info(last = False, out = True)
```

[1]:
```
[2022-10-14 13:05:54] oceanai - personality traits:
    Authors:
        Elena Ryumina [ryumina_ev@mail.ru]
        Dmitry Ryumin [dl_03.03.1991@mail.ru]
        Alexey Karpov [karpov@iias.spb.su]
    Maintainers:
        Elena Ryumina [ryumina_ev@mail.ru]
        Dmitry Ryumin [dl_03.03.1991@mail.ru]
    Version: 1.0.0-a7
    License: GPLv3
```

Better not to do that – 1 –

In [2]:

```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core._metadata_info(last = 1, out = [])
```

[2]:

```
[2022-10-14 13:05:54] oceanai - personality traits:
    Authors:
        Elena Ryumina [ryumina_ev@mail.ru]
        Dmitry Ryumin [dl_03.03.1991@mail.ru]
        Alexey Karpov [karpov@iias.spb.su]
    Maintainers:
        Elena Ryumina [ryumina_ev@mail.ru]
        Dmitry Ryumin [dl_03.03.1991@mail.ru]
    Version: 1.0.0-a7
    License: GPLv3
```

˙notebook˙display˙markdown(*message: str*, *last: bool = False*, *out: bool = True*) → None

Message display

Note: protected method

Parameters

- message (*str*) – Message
- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns
None

Return type
None

Examples

True – 1 –

In [1]:

```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core._notebook_display_markdown('Message')
```

[1]:

```
Message
```

Error – 1 –

In [2]:

```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core._notebook_display_markdown(1)
```

[2]:
```
[2022-10-14 15:52:03] Invalid argument types or values in "Core._notebook_
↪display_markdown" ...
```

˙other˙error(*message: str*, *last: bool = False*, *out: bool = True*) → None

Other error message

Note: protected method

Parameters

- message (*str*) – Message
- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

try: raise Exception
except:
    core._other_error(
        message = 'Other error message 1',
        last = False, out = True
    )

core.color_simple_ = '#FFF'
core.color_err_ = 'FF0000'
core.bold_text_ = False

try: raise Exception
except:
    core._other_error(
        message = 'Other error message 2',
        last = True, out = True
    )
```

[1]:
```
[2022-10-14 16:25:11] Other error message 1

    File: /var/folders/gw/w3k5kxtx0s3_nqdqw94zr8yh0000gn/T/ipykernel_20011/
↪333478077.py
```

(continues on next page)

(continued from previous page)

```
    Line: 5
    Method: <cell line: 5>
    Error type: Exception

[2022-10-14 16:25:11] Other error message 2

    File: /var/folders/gw/w3k5kxtx0s3_nqdqw94zr8yh0000gn/T/ipykernel_20011/
↪333478077.py
    Line: 16
    Method: <cell line: 16>
    Error type: Exception
```

Error – 1 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

try: raise Exception
except:
    core._other_error(
        message = '',
        last = False, out = True
    )
```

[2]:
```
[2022-10-14 16:25:11] Invalid argument types or values in "Core._other_error" ...
```

priority calculation(*df_files: Optional[DataFrame] = None, correlation_coefficients: Optional[DataFrame] = None, col_name_ocean: str = 'Trait', threshold: float = 0.55, number_priority: int = 1, number_importance_traits: int = 1, out: bool = True*) → DataFrame

Ranking preferences

Note: protected method

Parameters

- df files (*pd.DataFrame*) – **DataFrame** with data
- correlation coefficients (*pd.DataFrame*) – **DataFrame** with correlation coefficients
- col name ocean (*str*) – Column with the names of personality traits
- threshold (*float*) – Threshold for scores of traits polarity (e.g., introvert < 0.55, extrovert > 0.55)
- number priority (*int*) – Number of priority preferences
- number importance traits (*int*) – Number of the most important personality traits
- out (*bool*) – Display

Returns

DataFrame with ranked priority

Return type
pd.DataFrame

˙priority˙skill˙calculation(*df_files: Optional[DataFrame] = None*, *correlation_coefficients: Optional[DataFrame] = None*, *threshold: float = 0.55*, *out: bool = True*) → DataFrame

Ranking candidates by professional skills

Note: protected method

Parameters

- df˙files (*pd.DataFrame*) – **DataFrame** with data
- correlation˙coefficients (*pd.DataFrame*) – **DataFrame** with correlation coefficients
- threshold (*float*) – Threshold for scores of traits polarity (e.g., introvert < 0.55, extrovert > 0.55)
- out (*bool*) – Display

Returns
DataFrame with ranked candidates

Return type
pd.DataFrame

˙professional˙match(*df_files: Optional[DataFrame] = None*, *correlation_coefficients: Optional[DataFrame] = None*, *personality_type: Optional[str] = None*, *col_name_ocean: str = 'Trait'*, *threshold: float = 0.55*, *out: bool = True*) → DataFrame

Ранжирование кандидатов по одному из шестнадцати персональных типов по версии MBTI

Note: protected method

Parameters

- df˙files (*pd.DataFrame*) – **DataFrame** with data
- correlation˙coefficients (*pd.DataFrame*) – **DataFrame** with correlation coefficients
- personality˙type (*str*) – Персональный тип по версии MBTI
- threshold (*float*) – Threshold for scores of traits polarity (e.g., introvert < 0.55, extrovert > 0.55)
- out (*bool*) – Display
- col˙name˙ocean (*str*) –

Returns
DataFrame with ranked candidates

Return type
pd.DataFrame

˙progressbar(*message: str*, *progress: str*, *clear_out: bool = True*, *last: bool = False*, *out: bool = True*) → None

Progressbar

Note: protected method

Parameters

- message (*str*) – Message
- progress (*str*) – Progressbar
- clear˙out (*bool*) – Clearing the output area
- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

for cnt in range(1, 4):
    core._progressbar(
        message = 'Action cycle',
        progress = 'Iteration ' + str(cnt),
        clear_out = False,
        last = False, out = True
    )
```

[1]:
```
[2022-10-14 16:52:20] Action cycle

    Iteration 1

[2022-10-14 16:52:20] Action cycle

    Iteration 2

[2022-10-14 16:52:20] Action cycle

    Iteration 3
```

– 2 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

for cnt in range(1, 4):
    core._progressbar(
        message = 'Action cycle',
        progress = 'Iteration ' + str(cnt),
        clear_out = True,
        last = True, out = True
    )
```

[2]:
```
[2022-10-14 16:52:20] Action cycle

    Iteration 3
```

Error – 1 –

In [3]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

for cnt in range(1, 4):
    core._progressbar(
        message = 1,
        progress = 2,
        clear_out = True,
        last = False, out = True
    )
```

[3]:
```
[2022-10-14 16:52:38] Invalid argument types or values in "Core._progressbar" ...
```

˙progressbar˙union˙predictions(*message: str*, *item: int*, *info: str*, *len_paths: int*, *clear_out: bool = True*, *last: bool = False*, *out: bool = True*) → None

Progressbar for getting scores by audio

Note: private method

Parameters

- message (*str*) – Message
- item (*int*) – Number video file
- info (*str*) – Local path
- len˙paths (*int*) – Number of video files
- clear˙out (*bool*) – Clearing the output area
- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

l = range(1, 4, 1)

for progress in l:
    core._progressbar_union_predictions(
        message = 'Action cycle',
        item = progress,
        info = 'The path to the file',
        len_paths = len(l),
        clear_out = False,
        last = False, out = True
    )
```

[1]:
```
[2022-10-20 16:51:49] Action cycle

    1 из 3 (33.33%) ... The path to the file ...

[2022-10-20 16:51:49] Action cycle

    2 из 3 (66.67%) ... The path to the file ...

[2022-10-20 16:51:49] Action cycle

    3 из 3 (100.0%) ... The path to the file ...
```

– 2 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

l = range(1, 4, 1)

for progress in l:
    core._progressbar_union_predictions(
        message = 'Action cycle',
        item = progress,
        info = 'The path to the file',
        len_paths = len(l),
```

(continues on next page)

(continued from previous page)

```
        clear_out = True,
        last = True, out = True
    )
```

[2]:

```
[2022-10-20 16:51:55] Action cycle

    3 из 3 (100.0%) ... The path to the file ...
```

Error – 1 –

In [3]:

```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

l = range(1, 4, 1)

for progress in l:
    core._progressbar_union_predictions(
        message = 1,
        item = progress,
        info = 'The path to the file',
        len_paths = len(l),
        clear_out = True,
        last = False, out = True
    )
```

[3]:

```
[2022-10-20 16:55:15] Invalid argument types or values in "Audio._progressbar_
↪union_predictions" ...
```

r̈end(*last: bool = False*, *out: bool = True*) → None

End of runtime countdown

Note: protected method

Hint: Works in conjunction with `_r_start()`

Parameters

- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns

None

Return type

None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._r_start()
for cnt in range(0, 10000000): res = cnt * 2
core._r_end()
```

[1]:
```
--- Runtime: 0.819 sec. ---
```

Error – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

for cnt in range(0, 10000000): res = cnt * 2
core._r_end()
```

[1]:
```
--- Runtime: 1665756222.704 сек. ---
```

˙r˙start() → None

Start time countdown

Note: protected method

Hint: Works in conjunction with `_r_end()`

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._r_start()
for cnt in range(0, 10000000): res = cnt * 2
core._r_end()
```

[1]:
```
--- Runtime: 0.819 sec. ---
```

Error – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

for cnt in range(0, 10000000): res = cnt * 2
core._r_end()
```

[1]:
```
--- Runtime: 1665756222.704 сек. ---
```

˙round˙math(*val: Union[int, float]*, *out: bool = True*) → Union[int, bool]

Rounding numbers according to mathematical law

Note: protected method

Parameters

- val (*Union[int, float]*) – Number to round
- out (*bool*) – Display

Returns

Rounded number if no errors found, **False** otherwise

Return type

Union[int, bool]

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()

core._round_math(4.5)
```

[1]:
```
5
```

– 2 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()

core._round_math(-2.5)
```

[1]:

```
-3
```

Error – 1 –

In [3]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._round_math('')
```

[3]:
```
[2022-11-03 15:52:30] Invalid argument types or values in "Core._round_math" ...

False
```

˙save˙logs(*df: DataFrame*, *name: str*, *out: bool = True*) → bool

Saving the LOG file

Note: protected method

Parameters

- df (*pd.DataFrame*) – DataFrame to be saved to LOG file
- name (*str*) – LOG filename
- out (*bool*) – Display

Returns

True if the LOG file is saved, otherwise **False**

Return type

bool

Example

True – 1 –

In [1]:
```
import pandas as pd
from oceanai.modules.core.core import Core

df = pd.DataFrame.from_dict(
    data = {'Test': [1, 2, 3]}
)

core = Core(lang = 'en')

core.path_to_logs_ = './logs'

core._save_logs(
    df = df, name = 'test', out = True
)
```

[1]:
```
True
```

search_file(*path_to_file: str*, *ext: str*, *create: bool = False*, *out: bool = True*) → bool

File Search

Note: protected method

Parameters

- path_to_file (*str*) – The path to the file
- ext (*str*) – File extension
- create (*bool*) – Creating a file in case of its absence
- out (*bool*) – Print the execution process

Returns

True if the file is found, otherwise **False**

Return type

bool

stat_acoustic_features(*last: bool = False*, *out: bool = True*, ***kwargs: Union[int, Tuple[int], TensorShape]*) → None

Message with statistics of extracted features from an acoustic signal

Note: protected method

Parameters

- last (*bool*) – Replacing the last message
- out (*bool*) – Display
- **kwargs (*Union[int, Tuple[int], tf.TensorShape]*) – Additional named arguments

Returns

None

Return type

None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(
    lang = 'en',
    color_simple = '#FFF',
    color_info = '#1776D2',
```

(continues on next page)

(continued from previous page)

```
    bold_text = True,
)

core._stat_acoustic_features(
    last = False, out = True,
    len_hc_features = 12,
    len_melspectrogram_features = 12,
    shape_hc_features = [196, 25],
    shape_melspectrogram_features = [224, 224, 3],
)
```

[1]:

```
[2022-10-14 17:59:20] Statistics of the features extracted from the acoustic␣
↪signal:
    Total number of segments with:
        1. expert features: 12
        2. mel-spectrogram log: 12
    Dimension of the matrix of expert features of one segment: 196 × 25
    Tensor dimension with log chalk spectrograms of one segment: 224 × 224 × 3
```

Error – 1 –

In [2]:

```
from oceanai.modules.core.core import Core

core = Core(
    lang = 'en',
    color_simple = '#FFF',
    color_info = '#1776D2',
    bold_text = True,
)

core._stat_acoustic_features(
    last = False, out = True
)
```

[2]:

```
[2022-10-14 17:59:21] Invalid argument types or values in "Core._stat_acoustic_
↪features" ...
```

˙stat˙text˙features(*last: bool = False*, *out: bool = True*, ***kwargs: Union[int, Tuple[int], TensorShape]*) → None

Message with statistics of extracted features from a text

Note: protected method

Parameters

- last (*bool*) – Replacing the last message
- out (*bool*) – Display
- **kwargs (*Union[int, Tuple[int], tf.TensorShape]*) – Additional named arguments

Returns
None

Return type
None

˙stat˙visual˙features(*last: bool = False*, *out: bool = True*, ***kwargs: Union[int, Tuple[int], TensorShape]*) → None

Message with statistics of extracted features from a visual signal

---

Note: protected method

Parameters

- last (*bool*) – Replacing the last message
- out (*bool*) – Display
- **kwargs (*Union[int, Tuple[int], tf.TensorShape]*) – Additional named arguments

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(
    lang = 'en',
    color_simple = '#FFF',
    color_info = '#1776D2',
    bold_text = True,
)

core._stat_visual_features(
    last = False, out = True,
    len_hc_features = 23,
    len_nn_features = 23,
    shape_hc_features = [10, 115],
    shape_nn_features = [10, 512],
    fps_before = 30,
    fps_after = 10
)
```

[1]:
```
[2022-11-03 16:18:40] Statistics of extracted features from visual signal:
    Total number of segments since:
        1. expert features: 23
```

(continues on next page)

(continued from previous page)

```
        2. eural network features: 23
    Dimension of the matrix of expert features of one segment: 10 × 115
    Dimension of a tensor with neural network features of one segment: 10 × 512
    FPS down: with 30 to 10
```

Error – 1 –

In [2]:

```
from oceanai.modules.core.core import Core

core = Core(
    lang = 'en',
    color_simple = '#FFF',
    color_info = '#1776D2',
    bold_text = True,
)

core._stat_visual_features(
    last = False, out = True
)
```

[2]:

```
[2022-11-03 16:19:35] Invalid argument types or values in "Core._stat_visual_
↪features" ...
```

static ˙traceback() → Dict

Exception trace

Note: protected method

Returns
Dictionary describing the exception

Return type
Dict

Example

True – 1 –

In [1]:

```
import pprint
from oceanai.modules.core.core import Core

core = Core()

try: raise Exception
except:
    pp = pprint.PrettyPrinter(compact = True)
    pp.pprint(core._traceback())
```

[1]:

```
{
    'filename': '/var/folders/gw/w3k5kxtx0s3_nqdqw94zr8yh0000gn/T/ipykernel_
↪22253/4179594971.py',
    'lineno': 6,
    'name': '<cell line: 6>',
    'type': 'Exception'
}
```

property df˙accuracy˙: DataFrame

Getting a DataFrame with precision calculation results

Returns
DataFrame with precision calculation results

Return type
pd.DataFrame

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()
len(core.df_accuracy_)
```

[1]:
```
0
```

property df˙files˙: DataFrame

Getting a DataFrame with data

Returns
DataFrame with data

Return type
pd.DataFrame

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()
len(core.df_files_)
```

[1]:
```
0
```

property df˙files˙MBTI˙colleague˙match˙: DataFrame

Получение DataFrame с ранжированными коллегами на основе MBTI

Returns
DataFrame with data

Return type
pd.DataFrame

property df_files_MBTI_disorders_: DataFrame

Получение DataFrame с ранжированными профессиональными расстройствами на основе MBTI

Returns
DataFrame with data

Return type
pd.DataFrame

property df_files_MBTI_job_match_: DataFrame

Получение DataFrame с ранжированными кандидатами на основе MBTI

Returns
DataFrame with data

Return type
pd.DataFrame

property df_files_colleague_: DataFrame

Getting a DataFrame with ranked colleagues based on data

Returns
DataFrame with data

Return type
pd.DataFrame

property df_files_priority_: DataFrame

Getting a DataFrame with ranked priority based on the data

Returns
DataFrame with data

Return type
pd.DataFrame

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
len(core.df_files_priority_)
```

[1]:
```
0
```

property df_files_priority_skill_: DataFrame

Getting a DataFrame with ranked colleagues based on data

Returns
DataFrame with data

Return type
pd.DataFrame

property df˙files˙ranking˙: DataFrame

Getting a DataFrame with ranked data

Returns
DataFrame with data

Return type
pd.DataFrame

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')

core._round_math('')
```

[1]:
```
0
```

property df˙pkgs˙: DataFrame

Getting a DataFrame with versions of installed libraries

Returns
DataFrame with versions of installed libraries

Return type
pd.DataFrame

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()
core.libs_vers(out = False, runtime = True, run = True)
core.df_pkgs_
```

[1]:
```
|----|--------------|---------|
|    | Package      | Version |
|----|--------------|---------|
| 1  | TensorFlow   | 2.11.0  |
| 2  | Keras        | 2.11.0  |
| 3  | OpenCV       | 4.6.0   |
| 4  | MediaPipe    | 0.9.0   |
| 5  | NumPy        | 1.23.5  |
| 6  | SciPy        | 1.9.3   |
| 7  | Pandas       | 1.5.2   |
| 8  | Scikit-learn | 1.1.3   |
| 9  | OpenSmile    | 2.4.1   |
```

(continues on next page)

(continued from previous page)

```
| 10 | Librosa      | 0.9.2   |
| 11 | AudioRead    | 3.0.0   |
| 12 | IPython      | 8.7.0   |
| 14 | Requests     | 2.28.1  |
| 15 | JupyterLab   | 3.5.0   |
|----|--------------|---------|
```

property dict˙of˙accuracy˙: Dict[str, List[Union[int, float]]]

Getting a dictionary for a DataFrame with precision results

Hint: Based on this dictionary, a DataFrame is formed with the data `df_accuracy_`

Returns
Dictionary for DataFrame with precision results

Return type
Dict[str, List[Union[int, float]]]

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()
len(core.dict_of_accuracy_)
```

[1]:
```
0
```

property dict˙of˙files˙: Dict[str, List[Union[int, str, float]]]

Getting a dictionary for a DataFrame with data

Hint: Based on this dictionary, a DataFrame is formed with the data `df_files_`

Returns
Dictionary for DataFrame with data

Return type
Dict[str, List[Union[int, str, float]]]

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()
len(core.dict_of_files_)
```

[1]:
```
0
```

property is˙notebook˙: bool

Getting the result of a library run definition in Jupyter or similar

Returns
True if the library is run in Jupyter or similar, otherwise **False**

Return type
bool

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()
print(core.is_notebook_)
```

[1]:
```
True
```

libs˙vers(*out: bool = True*, *runtime: bool = True*, *run: bool = True*) → None

Getting and Displaying Versions of Installed Libraries

Parameters

- out (*bool*) – Display
- runtime (*bool*) – Run runtime
- run (*bool*) – Run blocking

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core.libs_vers(out = True, runtime = True, run = True)
```

[1]:
```
|----|---------------|---------|
|    | Package       | Version |
|----|---------------|---------|
| 1  | TensorFlow    | 2.11.0  |
| 2  | Keras         | 2.11.0  |
| 3  | OpenCV        | 4.6.0   |
| 4  | MediaPipe     | 0.9.0   |
| 5  | NumPy         | 1.23.5  |
| 6  | SciPy         | 1.9.3   |
| 7  | Pandas        | 1.5.2   |
| 8  | Scikit-learn  | 1.1.3   |
| 9  | OpenSmile     | 2.4.1   |
| 10 | Librosa       | 0.9.2   |
| 11 | AudioRead     | 3.0.0   |
| 12 | IPython       | 8.7.0   |
| 14 | Requests      | 2.28.1  |
| 15 | JupyterLab    | 3.5.0   |
| 16 | LIWC          | 0.5.0   |
| 17 | Transformers  | 4.24.0  |
| 18 | Sentencepiece | 0.1.99  |
| 19 | Torch         | 1.12.1  |
| 20 | Torchaudio    | 0.12.1  |
|----|---------------|---------|
--- Время выполнения: 0.005 сек. ---
```

– 2 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core.libs_vers(out = True, runtime = True, run = False)
```

[2]:
```
[2022-10-15 18:17:27] Run blocked by user ...
```

Error – 1 –

In [3]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core.libs_vers(out = True, runtime = True, run = 1)
```

[3]:
```
[2022-10-15 18:18:51] Invalid argument types or values in "Core.libs_vers" ...
```

property runtime˙

Getting runtime

Returns
Runtime

Return type
Union[int, float]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()

core._r_start()
for cnt in range(0, 10000000): res = cnt * 2
core._r_end(out = False)

print(core.runtime_)
```

[1]:
```
0.838
```

Error – 1 –

In [2]:
```
from oceanai.modules.core.core import Core

core = Core()

print(core.runtime_)
```

[2]:
```
-1
```

– 2 –

In [3]:
```
from oceanai.modules.core.core import Core

core = Core()

core._r_start()
for cnt in range(0, 10000000): res = cnt * 2

print(core.runtime_)
```

[3]:
```
-1
```

show˙notebook˙history˙output() → None

Display message output history in a Jupyter cell

Returns
None

Return type
None

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core._info(
    message = 'Announcement',
    last = False, out = False
)

core.show_notebook_history_output()
```

[1]:
```
[2022-10-15 18:27:46] Announcement
```

property true˙traits˙: Dict[str, str]

Getting paths to ground truth scores for calculating accuracy

Returns
Dictionary with paths to ground truth scores for calculating accuracy

Return type
Dict

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core()
core.true_traits_
```

[1]:
```
{
    'sberdisk': 'https://download.sberdisk.ru/download/file/410305241?
↪token=TFePK6w5CW6ADnq&filename=data_true_traits.csv'
}
```

property weights˙for˙big5˙: Dict[str, Dict]

Obtaining weights for neural network architectures

Returns
Dictionary with weights for neural network architectures

Return type
Dict

Example

True – 1 –

In [1]:
```
from oceanai.modules.core.core import Core

core = Core(lang = 'en')
core.weights_for_big5_
```

[1]:
```
{
    'audio': {
            'hc': {
                'sberdisk': 'https://download.sberdisk.ru/download/file/
↪400635799?token=MMRrak8fMsyzxLE&filename=weights_2022-05-05_11-27-55.h5',
            },
            'nn': {
                'sberdisk': 'https://download.sberdisk.ru/download/file/
↪400635678?token=W6LCtD33FQHnYEz&filename=weights_2022-05-03_07-46-14.h5',
            },
            'b5': {
                'openness': {
                        'sberdisk': 'https://download.sberdisk.ru/download/file/
↪405035301?token=443WRA9MFWqWBAE&filename=weights_2022-06-15_16-16-20.h5',
                },
                'conscientiousness': {
                        'sberdisk': 'https://download.sberdisk.ru/download/file/
↪405034601?token=eDG28m3H6c8bWoE&filename=weights_2022-06-15_16-21-57.h5',
                },
                'extraversion': {
                        'sberdisk': 'https://download.sberdisk.ru/download/file/
↪405034830?token=3daBSTYnmZaesee&filename=weights_2022-06-15_16-26-41.h5',
                },
                'agreeableness': {
                        'sberdisk': 'https://download.sberdisk.ru/download/file/
↪405034397?token=52ZPHMjb4CFmdYa&filename=weights_2022-06-15_16-32-51.h5',
                },
                'non_neuroticism': {
                        'sberdisk': 'https://download.sberdisk.ru/download/file/
↪405035156?token=q8CZJ99rZqcNxkM&filename=weights_2022-06-15_16-37-46.h5',
                },
            },
    },
    'video': {
            'hc': {
                'sberdisk': 'https://download.sberdisk.ru/download/file/
↪412059444?token=JXerCfAjJZg6crD&filename=weights_2022-08-27_18-53-35.h5',
            },
            'nn': {
                'sberdisk': 'https://download.sberdisk.ru/download/file/
↪412059478?token=85KeW6q4QKy6kP8&filename=weights_2022-03-22_16-31-48.h5',
            },
            'fe': {
```

(continues on next page)

(continued from previous page)

```
                'sberdisk': 'https://download.sberdisk.ru/download/file/
↪414207833?token=ygzxWEkndjSMnEL&filename=weights_2022-11-01_12-27-07.h5'
            },
            'b5': {
                'openness': {
                        'sberdisk': 'https://download.sberdisk.ru/download/file/
↪415127050?token=rfpy9TLdbeXtiN7&filename=weights_2022-06-15_16-46-30.h5',
                },
                'conscientiousness': {
                        'sberdisk': 'https://download.sberdisk.ru/download/file/
↪415126986?token=PnjzaHaR3wPg2RT&filename=weights_2022-06-15_16-48-50.h5',
                },
                'extraversion': {
                        'sberdisk': 'https://download.sberdisk.ru/download/file/
↪415127012?token=s5aTwbt8DBkt7G4&filename=weights_2022-06-15_16-54-06.h5',
                },
                'agreeableness': {
                        'sberdisk': 'https://download.sberdisk.ru/download/file/
↪415126845?token=joN7TMHk59Gffsf&filename=weights_2022-06-15_17-02-03.h5',
                },
                'non_neuroticism': {
                        'sberdisk': 'https://download.sberdisk.ru/download/file/
↪415127032?token=NEBSsE7mjyjen3o&filename=weights_2022-06-15_17-06-15.h5',
                }
            }
    }
}
```

Archive processing

class `oceanai.modules.lab.unzip.`UnzipMessages(*lang: str = 'ru'*, *color_simple: str = '#666'*, *color_info: str = '#1776D2'*, *color_err: str = '#FF0000'*, *color_true: str = '#008001'*, *bold_text: bool = True*, *text_runtime: str = ''*, *num_to_df_display: int = 30*)

Bases: *Core*

Class for messages

Parameters

- lang (***str***) – See *lang*
- color_simple (***str***) – See *color_simple*
- color_info (***str***) – See *color_info*
- color_err (***str***) – See *color_err*
- color_true (***str***) – See *color_true*
- bold_text (***bool***) – See *bold_text*
- num_to_df_display (***int***) – See *num_to_df_display*
- text_runtime (***str***) – See *text_runtime*

class oceanai.modules.lab.unzip.Unzip(*lang: str = 'ru'*, *color_simple: str = '#666'*, *color_info: str = '#1776D2'*, *color_err: str = '#FF0000'*, *color_true: str = '#008001'*, *bold_text: bool = True*, *text_runtime: str = ''*, *num_to_df_display: int = 30*)

Bases: *UnzipMessages*

Class for archive processing

Parameters

- lang (***str***) – See *lang*
- color˙simple (***str***) – See *color_simple*
- color˙info (***str***) – See *color_info*
- color˙err (***str***) – See *color_err*
- color˙true (***str***) – See *color_true*
- bold˙text (***bool***) – See *bold_text*
- num˙to˙df˙display (***int***) – See *num_to_df_display*
- text˙runtime (***str***) – See *text_runtime*

¨progressbar˙unzip(*path_to_zipfile: str*, *progress: float*, *clear_out: bool = True*, *last: bool = False*, *out: bool = True*) → None

Progressbar

Note: private method

Parameters

- path˙to˙zipfile (***str***) – Full path to the archive
- progress (***float***) – Percentage of progress (from **0.0** to **100.0**)
- clear˙out (***bool***) – Clearing the output
- last (***bool***) – Replacing the last message
- out (***bool***) – Display

Return type
None

˙unzip(*path_to_zipfile: str*, *new_name: Optional[str] = None*, *force_reload: bool = True*, *out: bool = True*) → bool

Unzipping the archive (without clearing the message output history in the Jupyter cell)

Note: protected method

Parameters

- path˙to˙zipfile (***str***) – Full path to the archive
- new˙name (***str***) – Directory name for unzipping
- force˙reload (***bool***) – Forced unzipping

- out (*bool*) – Display

Returns
True if unzipping was successful, otherwise **False**

Return type
bool

property path_to_unzip: str

Getting a directory for unzipping

Returns
Unzip directory

Return type
str

unzip(*path_to_zipfile: str*, *new_name: Optional[str] = None*, *force_reload: bool = True*, *out: bool = True*) → bool

Unzipping an archive

Parameters

- path_to_zipfile (*str*) – Full path to the archive
- new_name (*str*) – Directory name for unzipping
- force_reload (*bool*) – Forced unzipping
- out (*bool*) – Display

Returns
True if unzipping was successful, otherwise **False**

Return type
bool

Downloading files

class oceanai.modules.lab.download.DownloadMessages(*lang: str = 'ru'*, *color_simple: str = '#666'*, *color_info: str = '#1776D2'*, *color_err: str = '#FF0000'*, *color_true: str = '#008001'*, *bold_text: bool = True*, *text_runtime: str = ''*, *num_to_df_display: int = 30*)

Bases: *Unzip*

Class for messages

Parameters

- lang (*str*) – See *lang*
- color_simple (*str*) – See *color_simple*
- color_info (*str*) – See *color_info*
- color_err (*str*) – See *color_err*
- color_true (*str*) – See *color_true*
- bold_text (*bool*) – See *bold_text*
- num_to_df_display (*int*) – See *num_to_df_display*

- text˙runtime (*str*) – See *text_runtime*

class oceanai.modules.lab.download.Download(*lang: str = 'ru'*, *color_simple: str = '#666'*, *color_info: str = '#1776D2'*, *color_err: str = '#FF0000'*, *color_true: str = '#008001'*, *bold_text: bool = True*, *text_runtime: str = ''*, *num_to_df_display: int = 30*)

Bases: *DownloadMessages*

Class for downloading files

Parameters

- lang (*str*) – See *lang*
- color˙simple (*str*) – See *color_simple*
- color˙info (*str*) – See *color_info*
- color˙err (*str*) – See *color_err*
- color˙true (*str*) – See *color_true*
- bold˙text (*bool*) – See *bold_text*
- num˙to˙df˙display (*int*) – See *num_to_df_display*
- text˙runtime (*str*) – See *text_runtime*

¨progressbar˙download˙file˙from˙url(*url_filename: str*, *progress: float*, *clear_out: bool = True*, *last: bool = False*, *out: bool = True*) → None

File download progress bar from URL

Note: private method

Parameters

- url˙filename (*str*) – Path to file
- progress (*float*) – Percent complete (from **0.0** to **100.0**)
- clear˙out (*bool*) – Clearing the output area
- last (*bool*) – Replacing the last message
- out (*bool*) – Display

Returns
None

Return type
None

Examples

True – 1 –

In [1]:
```
import numpy as np
from oceanai.modules.lab.download import Download

download = Download(lang = 'en')

for progress in np.arange(0., 101, 25):
    download._Download__progressbar_download_file_from_url(
        url_filename = 'https://clck.ru/32Nwdk',
        progress = float(progress),
        clear_out = False,
        last = False, out = True
    )
```

[1]:
```
[2022-10-16 16:58:51] File download "https://clck.ru/32Nwdk" (0.0%) ...

[2022-10-16 16:58:51] File download "https://clck.ru/32Nwdk" (25.0%) ...

[2022-10-16 16:58:51] File download "https://clck.ru/32Nwdk" (50.0%) ...

[2022-10-16 16:58:51] File download "https://clck.ru/32Nwdk" (75.0%) ...

[2022-10-16 16:58:51] File download "https://clck.ru/32Nwdk" (100.0%) ...
```

– 2 –

In [2]:
```
import numpy as np
from oceanai.modules.lab.download import Download

download = Download(lang = 'en')

for progress in np.arange(0., 101, 25):
    download._Download__progressbar_download_file_from_url(
        url_filename = 'https://clck.ru/32Nwdk',
        progress = float(progress),
        clear_out = True,
        last = True, out = True
    )
```

[2]:
```
[2022-10-16 16:59:41] File download "https://clck.ru/32Nwdk" (100.0%) ...
```

Error – 1 –

In [3]:
```
import numpy as np
from oceanai.modules.lab.download import Download

download = Download(lang = 'en')

for progress in np.arange(0., 101, 25):
```

(continues on next page)

(continued from previous page)

```
    download._Download__progressbar_download_file_from_url(
        url_filename = 'https://clck.ru/32Nwdk',
        progress = 101,
        clear_out = True,
        last = False, out = True
    )
```

[3]:
```
[2022-10-16 17:00:11] Invalid argument types or values in "Download.__
↪progressbar_download_file_from_url" ...
```

˙download˙file˙from˙url(*url: str*, *force_reload: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → int

Downloading file from URL (without clearing message output history in Jupyter cell)

Note: protected method

Parameters

- url (*str*) – Full path to the file
- force˙reload (*bool*) – Force a file download from the network
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Execution blocking

Returns

Response status code:

- 200 - File downloaded
- 400 - Error validating arguments
- 403 - Run blocked by user
- 404 - Failed to download file

Return type

int

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.download import Download

download = Download(lang = 'en')

download.path_to_save_ = './models'
download.chunk_size_ = 2000000

res_download_file_from_url = download._download_file_from_url(
```

(continues on next page)

(continued from previous page)

```
    url = 'https://download.sberdisk.ru/download/file/400635799?
↪token=MMRrak8fMsyzxLE&filename=weights_2022-05-05_11-27-55.h5',
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[1]:

```
[2022-10-16 20:23:25] File download "weights_2022-05-05_11-27-55.h5" (100.0%) ..
↪.

--- Runtime: 0.373 сек. ---

200
```

– 2 –

In [2]:

```
from oceanai.modules.lab.download import Download

download = Download(lang = 'en')

download.path_to_save_ = './models'
download.chunk_size_ = 2000000

res_download_file_from_url = download._download_file_from_url(
    url = 'https://clck.ru/32Nwdk',
    force_reload = True,
    out = True,
    runtime = True,
    run = False
)
res_download_file_from_url
```

[2]:

```
[2022-10-16 19:33:05] Run blocked by user ...

403
```

Errors – 1 –

In [3]:

```
from oceanai.modules.lab.download import Download

download = Download(lang = 'en')

download.path_to_save_ = './models'
download.chunk_size_ = 2000000

res_download_file_from_url = download._download_file_from_url(
    url = 1,
    force_reload = True,
    out = True,
    runtime = True,
```

(continues on next page)

(continued from previous page)

```
    run = True
)
res_download_file_from_url
```

[3]:
```
[2022-10-16 19:33:01] Invalid argument types or values in "Download._download_
↪file_from_url" ...

400
```

– 2 –

In [4]:
```
from oceanai.modules.lab.download import Download

download = Download(lang = 'en')

download.path_to_save_ = './models'
download.chunk_size_ = 2000000

res_download_file_from_url = download._download_file_from_url(
    url = 'https://',
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
res_download_file_from_url
```

[4]:
```
[2022-10-16 19:33:10] Something went wrong ... the specified URL could not be␣
↪processed ...

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/download.py
    Line: 257
    Method: _download_file_from_url
    Error type: InvalidURL

--- Runtime: 0.061 сек. ---

404
```

– 3 –

In [5]:
```
from oceanai.modules.lab.download import Download

download = Download(lang = 'en')

download.path_to_save_ = './models'
download.chunk_size_ = 2000000

res_download_file_from_url = download._download_file_from_url(
    url = 'https://www.iconfinder.com/icons/4375050/download/svg/4096',
    force_reload = True,
```

(continues on next page)

(continued from previous page)

```
    out = True,
    runtime = True,
    run = True
)
res_download_file_from_url
```

[5]:

```
[2022-10-16 19:33:15] File download "4375050_logo_python_icon.svg"

[2022-10-16 19:33:15] Something went wrong ... Download file size not defined ..
↪.

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/download.py
    Line: 324
    Method: _download_file_from_url
    Error type: InvalidContentLength

--- Runtime: 0.386 сек. ---

404
```

download_file_from_url(*url: str*, *force_reload: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → int

Downloading a file from a URL

Parameters

- url (*str*) – Full path to the file
- force_reload (*bool*) – Force a file download from the network
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Execution blocking

Returns

Response status code:

- 200 - File downloaded
- 400 - Error validating arguments
- 403 - Run blocked by user
- 404 - Failed to download file

Return type

int

Example

Audio

class oceanai.modules.lab.audio.AudioMessages(*lang: str = 'ru'*, *color_simple: str = '#666'*, *color_info: str = '#1776D2'*, *color_err: str = '#FF0000'*, *color_true: str = '#008001'*, *bold_text: bool = True*, *text_runtime: str = ''*, *num_to_df_display: int = 30*)

Bases: *Download*

Class for messages

Parameters

- lang (*str*) – See *lang*
- color_simple (*str*) – See *color_simple*
- color_info (*str*) – See *color_info*
- color_err (*str*) – See *color_err*
- color_true (*str*) – See *color_true*
- bold_text (*bool*) – See *bold_text*
- num_to_df_display (*int*) – See *num_to_df_display*
- text_runtime (*str*) – See *text_runtime*

class oceanai.modules.lab.audio.Audio(*lang: str = 'ru'*, *color_simple: str = '#666'*, *color_info: str = '#1776D2'*, *color_err: str = '#FF0000'*, *color_true: str = '#008001'*, *bold_text: bool = True*, *text_runtime: str = ''*, *num_to_df_display: int = 30*)

Bases: *AudioMessages*

Audio processing class

Parameters

- lang (*str*) – See *lang*
- color_simple (*str*) – See *color_simple*
- color_info (*str*) – See *color_info*
- color_err (*str*) – See *color_err*
- color_true (*str*) – See *color_true*
- bold_text (*bool*) – See *bold_text*
- num_to_df_display (*int*) – See *num_to_df_display*
- text_runtime (*str*) – See *text_runtime*

__concat_pred(*pred_hc: ndarray*, *pred_melspectrogram: ndarray*, *out: bool = True*) → List[Optional[ndarray]]

Concatenation of scores by hand-crafted and deep features

Note: private method

Parameters

- pred˙hc (*np.ndarray*) – Scores based on had-crafted features
- pred˙melspectrogram (*np.ndarray*) – Scores based on deep features
- out (*bool*) – Display

Returns
Concatenated scores by hand-crafted and deep features

Return type
List[Optional[np.ndarray]]

Examples

True – 1 –

In [1]:
```
import numpy as np
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

arr_hc = np.array([
    [0.64113516, 0.6217892, 0.54451424, 0.6144415, 0.59334993],
    [0.6652424, 0.63606125, 0.572305, 0.63169795, 0.612515]
])

arr_melspectrogram = np.array([
    [0.56030345, 0.7488746, 0.44648764, 0.59893465, 0.5701077],
    [0.5900006, 0.7652722, 0.4795154, 0.6409055, 0.6088242]
])

audio._Audio__concat_pred(
    pred_hc = arr_hc,
    pred_melspectrogram = arr_melspectrogram,
    out = True
)
```

[1]:
```
[
    array([
            0.64113516, 0.6652424, 0.65318878, 0.65318878, 0.65318878,
            0.65318878, 0.65318878, 0.65318878, 0.65318878, 0.65318878,
            0.65318878, 0.65318878, 0.65318878, 0.65318878, 0.65318878,
            0.65318878, 0.56030345, 0.5900006, 0.57515202, 0.57515202,
            0.57515202, 0.57515202, 0.57515202, 0.57515202, 0.57515202,
            0.57515202, 0.57515202, 0.57515202, 0.57515202, 0.57515202,
            0.57515202, 0.57515202
    ]),
    array([
            0.6217892, 0.63606125, 0.62892523, 0.62892523, 0.62892523,
            0.62892523, 0.62892523, 0.62892523, 0.62892523, 0.62892523,
            0.62892523, 0.62892523, 0.62892523, 0.62892523, 0.62892523,
            0.62892523, 0.7488746, 0.7652722, 0.7570734, 0.7570734,
            0.7570734, 0.7570734, 0.7570734, 0.7570734, 0.7570734,
            0.7570734, 0.7570734, 0.7570734, 0.7570734, 0.7570734,
```

(continues on next page)

(continued from previous page)

```
            0.7570734, 0.7570734
    ]),
    array([
            0.54451424, 0.572305, 0.55840962, 0.55840962, 0.55840962,
            0.55840962, 0.55840962, 0.55840962, 0.55840962, 0.55840962,
            0.55840962, 0.55840962, 0.55840962, 0.55840962, 0.55840962,
            0.55840962, 0.44648764, 0.4795154, 0.46300152, 0.46300152,
            0.46300152, 0.46300152, 0.46300152, 0.46300152, 0.46300152,
            0.46300152, 0.46300152, 0.46300152, 0.46300152, 0.46300152,
            0.46300152, 0.46300152
    ]),
    array([
            0.6144415, 0.63169795, 0.62306972, 0.62306972, 0.62306972,
            0.62306972, 0.62306972, 0.62306972, 0.62306972, 0.62306972,
            0.62306972, 0.62306972, 0.62306972, 0.62306972, 0.62306972,
            0.62306972, 0.59893465, 0.6409055, 0.61992008, 0.61992008,
            0.61992008, 0.61992008, 0.61992008, 0.61992008, 0.61992008,
            0.61992008, 0.61992008, 0.61992008, 0.61992008, 0.61992008,
            0.61992008, 0.61992008
    ]),
    array([
            0.59334993, 0.612515, 0.60293247, 0.60293247, 0.60293247,
            0.60293247, 0.60293247, 0.60293247, 0.60293247, 0.60293247,
            0.60293247, 0.60293247, 0.60293247, 0.60293247, 0.60293247,
            0.60293247, 0.5701077, 0.6088242, 0.58946595, 0.58946595,
            0.58946595, 0.58946595, 0.58946595, 0.58946595, 0.58946595,
            0.58946595, 0.58946595, 0.58946595, 0.58946595, 0.58946595,
            0.58946595, 0.58946595
    ])
]
```

Error – 1 –

In [2]:

```
import numpy as np
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

arr_hc = np.array([
    [0.64113516, 0.6217892, 0.54451424, 0.6144415],
    [0.6652424, 0.63606125, 0.572305, 0.63169795, 0.612515]
])

arr_melspectrogram = np.array([
    [0.56030345, 0.7488746, 0.44648764, 0.59893465, 0.5701077],
    [0.5900006, 0.7652722, 0.4795154, 0.6409055, 0.6088242]
])

audio._Audio__concat_pred(
    pred_hc = arr_hc,
    pred_melspectrogram = arr_melspectrogram,
    out = True
```

(continues on next page)

(continued from previous page)

```
)
```

[3]:

```
[2022-10-20 22:33:31] Something went wrong ... concatenation of scores by hand-
↪crafted and deep features was not performed (audio modality) ...

[]
```

¨load˙audio˙model˙b5(*show_summary: bool = False*, *out: bool = True*) → Optional[Model]

Formation of the neural network architecture of the model to obtain the personality traits scores

Note: private method

Parameters

- show˙summary (*bool*) – Displaying the formed neural network architecture of the model
- out (*bool*) – Display

Returns

None if the types or values of the arguments are invalid, otherwise the neural network model **tf.keras.Model** to get the personality traits scores

Return type

Optional[tf.keras.Model]

Examples

True – 1 –

In [1]:

```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio._Audio__load_audio_model_b5(
    show_summary = True, out = True
)
```

[1]:

```
[2022-10-18 11:39:22] Formation of neural network architectures of models for␣
↪obtaining the personality traits scores (audio modality) ...

Model: "model_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32)]              0

 dense_1 (Dense)             (None, 1)                 33

 activ_1 (Activation)        (None, 1)                 0
```

(continues on next page)

(continued from previous page)

```
12
13 ==================================================================
14 Total params: 33
15 Trainable params: 33
16 Non-trainable params: 0
17 ------------------------------------------------------------------
18 --- Runtime: 0.163 sec. ---
19
20 True
```

Error – 1 –

In [2]:
```
1 from oceanai.modules.lab.audio import Audio
2
3 audio = Audio(lang = 'en')
4
5 audio._Audio__load_audio_model_b5(
6     show_summary = True, out = []
7 )
```

[3]:
```
1 [2022-10-17 10:53:03] Invalid argument types or values in "Audio.__load_audio_
  ↪model_b5" ...
```

¨load˙model˙weights(*url: str*, *force_reload: bool = True*, *info_text: str = ''*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Downloading the weights of the neural network model

Note: private method

Parameters

- url (*str*) – Full path to the file with weights of the neural network model
- force˙reload (*bool*) – Forced download of a file with weights of a neural network model from the network
- info˙text (*str*) – Text for informational message
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the weights of the neural network model are downloaded, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.path_to_save_ = './models'
audio.chunk_size_ = 2000000

audio._Audio__load_model_weights(
    url = 'https://download.sberdisk.ru/download/file/400635799?
↪token=MMRrak8fMsyzxLE&filename=weights_2022-05-05_11-27-55.h5',
    force_reload = True,
    info_text = 'Downloading the weights of the neural network model',
    out = True, runtime = True, run = True
)
```

[1]:
```
[2022-10-17 12:21:48] Downloading the weights of the neural network model

[2022-10-17 12:21:48] File download "weights_2022-05-05_11-27-55.h5" (100.0%) ..
↪.

--- Runtime: 0.439 sec. ---

True
```

– 2 –

In [2]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.path_to_save_ = './models'
audio.chunk_size_ = 2000000

audio._Audio__load_model_weights(
    url = './models/weights_2022-05-05_11-27-55.h5',
    force_reload = True,
    info_text = 'Downloading the weights of the neural network model',
    out = True, runtime = True, run = True
)
```

[2]:
```
[2022-10-17 12:21:50] Downloading the weights of the neural network model

--- Runtime: 0.002 sec. ---

True
```

Error – 1 –

In [3]:

```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.path_to_save_ = './models'
audio.chunk_size_ = 2000000

audio._Audio__load_model_weights(
    url = 'https://download.sberdisk.ru/download/file/400635799?
↪token=MMRrak8fMsyzxLE&filename=weights_2022-05-05_11-27-55.h5',
    force_reload = True, info_text = '',
    out = True, runtime = True, run = True
)
```

[3]:

```
[2022-10-17 12:21:57] Invalid argument types or values in "Audio.__load_model_
↪weights" ...

False
```

¨norm˙pred(*pred_data: ndarray*, *len_spec: int = 16*, *out: bool = True*) → ndarray

Normalization of scores by hand-crafted and deep features

Note: private method

Parameters

- pred˙data (*np.ndarray*) – Scores
- len˙spec (*int*) – The maximum size of the scores vector
- out (*bool*) – Display

Returns

Normalized scores by hand-crafted and deep features

Return type

np.ndarray

Examples

True – 1 –

In [1]:

```
import numpy as np
from oceanai.modules.lab.audio import Audio

audio = Audio()

arr = np.array([
    [0.64113516, 0.6217892, 0.54451424, 0.6144415, 0.59334993],
    [0.6652424, 0.63606125, 0.572305, 0.63169795, 0.612515]
])

```

(continues on next page)

(continued from previous page)

```
audio._Audio__norm_pred(
    pred_data = arr,
    len_spec = 4,
    out = True
)
```

[1]:

```
array([
    [0.64113516, 0.6217892 , 0.54451424, 0.6144415 , 0.59334993],
    [0.6652424 , 0.63606125, 0.572305  , 0.63169795, 0.612515],
    [0.65318878, 0.62892523, 0.55840962, 0.62306972, 0.60293247],
    [0.65318878, 0.62892523, 0.55840962, 0.62306972, 0.60293247]
])
```

Error – 1 –

In [2]:

```
import numpy as np
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

arr = np.array([])

audio._Audio__norm_pred(
    pred_data = arr,
    len_spec = 4,
    out = True
)
```

[3]:

```
[2022-10-20 22:03:17] Invalid argument types or values in "Audio.__norm_pred" ...

array([], dtype=float64)
```

¨smile() → Smile

Extracting OpenSmile features

Note: private method

Returns
Extracted OpenSmile features

Return type
opensmile.core.smile.Smile

Example

True – 1 –

In [1]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')
audio._Audio__smile()
```

[1]:
```
{
    '$opensmile.core.smile.Smile': {
        'feature_set': 'eGeMAPSv02',
        'feature_level': 'LowLevelDescriptors',
        'options': {},
        'sampling_rate': None,
        'channels': [0],
        'mixdown': False,
        'resample': False
    }
}
```

get_acoustic_features(*path: str*, *sr: int = 44100*, *window: Union[int, float] = 2.0*, *step: Union[int, float] = 1.0*, *last: bool = False*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → Tuple[List[Optional[ndarray]], List[Optional[ndarray]]]

Extracting features from an acoustic signal (without clearing the message output history in a Jupyter cell)

Note: protected method

Parameters

- path (*str*) – Path to the audio or video file
- sr (*int*) – Sampling frequency
- window (*Union[int, float]*) – Signal segment window size (in seconds)
- step (*Union[int, float]*) – Signal segment window shift step (in seconds)
- last (*bool*) – Replacing the last message
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

Tuple with two lists: 1. List with hand-crafted features 2. List with mel-spectrograms

Return type

Tuple[List[Optional[np.ndarray]], List[Optional[np.ndarray]]]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

sr = 44100
path = '/Users/dl/GitHub/oceanai/oceanai/dataset/test80_01/glgfB3vFewc.004.mp4'

hc_features, melspectrogram_features = audio._get_acoustic_features(
    path = path, sr = sr,
    window = 2, step = 1,
    last = False, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-10-19 14:58:19] Extraction of features (hand-crafted and mel-
↪spectrograms) from an acoustic signal ...

[2022-10-19 14:58:20] Statistics of the features extracted from the acoustic␣
↪signal:
    Total number of segments with:
    1. hand-crafted features: 12
    2. mel-spectrogram log: 12
    Dimension of the matrix of hand-crafted features of one segment: 196 × 25
    Dimension of the tensor with log mel-spectrograms of one segment: 224 × 224␣
↪× 3

--- Runtime: 1.273 sec. ---
```

Errors – 1 –

In [2]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

sr = 44100
path = '/Users/dl/GitHub/oceanai/oceanai/dataset/test80_01/glgfB3vFewc.004.mp4'

hc_features, melspectrogram_features = audio._get_acoustic_features(
    path = 1, sr = sr,
    window = 2, step = 1,
    last = False, out = True,
    runtime = True, run = True
)
```

[2]:
```
[2022-10-19 15:33:04] Invalid argument types or values in "Audio._get_acoustic_
↪features" ...
```

– 2 –

In [2]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

sr = 44100
path = '/Users/dl/GitHub/oceanai/oceanai/dataset/test80_01/glgfB3vFewc.004.mp4'

hc_features, melspectrogram_features = audio._get_acoustic_features(
    path = path, sr = sr,
    window = 0.04, step = 1,
    last = False, out = True,
    runtime = True, run = True
)
```

[2]:
```
[2022-10-19 15:34:38] Extraction of features (hand-crafted and mel-
↪spectrograms) from an acoustic signal ...

[2022-10-19 15:34:38] Something went wrong ... the size (0.04) of the signal␣
↪segment window is too small ...

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/audio.py
    Line: 863
    Method: _get_acoustic_features
    Error type: IsSmallWindowSizeError

--- Runtime: 0.049 sec. ---
```

property audio˙model˙hc˙: Optional[Model]

Obtaining a neural network model **tf.keras.Model** to obtain scores by hand-crafted features

Returns

Neural network model **tf.keras.Model** or None

Return type

Optional[tf.keras.Model]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.load_audio_model_hc(
    show_summary = False, out = True,
    runtime = True, run = True
)

audio.audio_model_hc_
```

[1]:

```
[2022-10-17 13:54:35] Formation of the neural network architecture of the model␣
↪for obtaining scores by hand-crafted features (audio modality) ...

--- Runtime: 0.509 sec. ---

<keras.engine.functional.Functional at 0x13dd600a0>
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.audio_model_hc_
```

[2]:
```

```

property audio˙model˙nn˙: Optional[Model]

Obtaining a neural network model **tf.keras.Model** to obtain scores for deep features

Returns
Neural network model **tf.keras.Model** or None

Return type
Optional[tf.keras.Model]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.load_audio_model_nn(
    show_summary = False, out = True,
    runtime = True, run = True
)

audio.audio_model_nn_
```

[1]:
```
[2022-10-17 13:58:29] Formation of a neural network architecture for obtaining␣
↪scores by deep features ...

--- Runtime: 0.444 sec. ---

<keras.engine.functional.Functional at 0x13db97760>
```

Error – 1 –

In [2]:

```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.audio_model_nn_
```

[2]:

```

```

property audio˙models˙b5˙: Dict[str, Optional[Model]]

Obtaining neural network models **tf.keras.Model** to obtain the personality traits scores

Returns
Dictionary with neural network models **tf.keras.Model**

Return type
Dict

Examples

True – 1 –

In [1]:

```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.load_audio_models_b5(
    show_summary = False, out = True,
    runtime = True, run = True
)

audio.audio_models_b5_
```

[1]:

```
[2022-10-19 15:45:35] Formation of neural network architectures of models for␣
↪obtaining the personality traits scores (audio modality) ...

--- Runtime: 0.07 sec. ---

{
    'openness': <keras.engine.functional.Functional at 0x1481e03a0>,
    'conscientiousness': <keras.engine.functional.Functional at 0x147d13520>,
    'extraversion': <keras.engine.functional.Functional at 0x1481edfa0>,
    'agreeableness': <keras.engine.functional.Functional at 0x1481cfc40>,
    'non_neuroticism': <keras.engine.functional.Functional at 0x1481cffd0>
}
```

Error – 1 –

In [2]:

```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.audio_models_b5_
```

[2]:
```
{
    'openness': None,
    'conscientiousness': None,
    'extraversion': None,
    'agreeableness': None,
    'non_neuroticism': None
}
```

get_acoustic_features(*path: str*, *sr: int = 44100*, *window: Union[int, float] = 2.0*, *step: Union[int, float] = 1.0*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → Tuple[List[Optional[ndarray]], List[Optional[ndarray]]]

Extracting features from an acoustic signal

Parameters

- path (*str*) – Path to the audio or video file
- sr (*int*) – Sampling frequency
- window (*Union[int, float]*) – Signal segment window size (in seconds)
- step (*Union[int, float]*) – Signal segment window shift step (in seconds)
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

Tuple with two lists: 1. List with hand-crafted features 2. List with mel-spectrograms

Return type

Tuple[List[Optional[np.ndarray]], List[Optional[np.ndarray]]]

Example

get_audio_union_predictions(*depth: int = 1*, *recursive: bool = False*, *sr: int = 44100*, *window: Union[int, float] = 2.0*, *step: Union[int, float] = 1.0*, *accuracy=True*, *url_accuracy: str = ''*, *logs: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Get audio scores

Parameters

- depth (*int*) – Hierarchy depth for getting data
- recursive (*bool*) – Recursive data search
- sr (*int*) – Sampling frequency
- window (*Union[int, float]*) – Signal segment window size (in seconds)
- step (*Union[int, float]*) – Signal segment window shift step (in seconds)
- accuracy (*bool*) – Accuracy calculation
- url_accuracy (*str*) – Full path to the file with ground truth scores for calculating accuracy
- logs (*bool*) – If necessary, generate a LOG file
- out (*bool*) – Display
- runtime (*bool*) – Runtime count

- run (*bool*) – Run blocking

Returns
True if scores are successfully received, otherwise **False**

Return type
bool

Example

load_audio_model_hc(*show_summary: bool = False*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Formation of the neural network architecture of the model for obtaining scores by hand-crafted features

Parameters

- show_summary (*bool*) – Displaying the formed neural network architecture of the model
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns
True if the neural network architecture of the model is formed, otherwise **False**

Return type
bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')
audio.load_audio_model_hc(
    show_summary = False, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-10-17 13:16:23] Formation of the neural network architecture of the model␣
↪for obtaining scores by hand-crafted features (audio modality) ...

--- Runtime: 0.364 sec. ---

True
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')
audio.load_audio_model_hc(
```

(continues on next page)

(continued from previous page)

```
    show_summary = 1, out = True,
    runtime = True, run = True
)
```

[2]:

```
[2022-10-17 13:20:04] Invalid argument types or values in "Audio.load_audio_
↪model_hc" ...

False
```

load˙audio˙model˙nn(*show_summary: bool = False*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Formation of a neural network architecture for obtaining scores by deep features

Parameters

- show˙summary (*bool*) – Displaying the formed neural network architecture of the model
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the neural network architecture of the model is formed, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:

```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')
audio.load_audio_model_nn(
    show_summary = True, out = True,
    runtime = True, run = True
)
```

[1]:

```
[2022-10-17 13:25:34] Formation of a neural network architecture for obtaining␣
↪scores by deep features (audio modality) ...

Model: "model"

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

```

(continues on next page)

(continued from previous page)

```
block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0

block4_conv1 (Conv2D)       (None, 28, 28, 512)       1180160

block4_conv2 (Conv2D)       (None, 28, 28, 512)       2359808

block4_conv3 (Conv2D)       (None, 28, 28, 512)       2359808

block4_pool (MaxPooling2D)  (None, 14, 14, 512)       0

block5_conv1 (Conv2D)       (None, 14, 14, 512)       2359808

block5_conv2 (Conv2D)       (None, 14, 14, 512)       2359808

block5_conv3 (Conv2D)       (None, 14, 14, 512)       2359808

block5_pool (MaxPooling2D)  (None, 7, 7, 512)         0

flatten (Flatten)           (None, 25088)             0

dense (Dense)               (None, 512)               12845568

dropout (Dropout)           (None, 512)               0

dense_1 (Dense)             (None, 256)               131328

dense_2 (Dense)             (None, 5)                 1285

=================================================================
Total params: 27,692,869
Trainable params: 27,692,869
Non-trainable params: 0
_________________________________________________________________
--- Runtime: 0.407 sec. ---

True
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')
audio.load_audio_model_nn(
    show_summary = 1, out = True,
    runtime = True, run = True
)
```

[2]:
```
[2022-10-17 13:25:40] Invalid argument types or values in "Audio.load_audio_
↪model_nn" ...

False
```

load_audio_model_weights_hc(*url: str*, *force_reload: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Downloading the weights of the neural network model to obtain scores by hand-crafted features

Parameters

- url (*str*) – Full path to the file with weights of the neural network model
- force_reload (*bool*) – Forced download of a file with weights of a neural network model from the network
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the weights of the neural network model are downloaded, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.load_audio_model_hc(
    show_summary = False, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-10-17 14:24:28] Formation of the neural network architecture of the model␣
↪for obtaining scores by hand-crafted features (audio modality) ...

--- Runtime: 0.398 sec. ---
```

(continues on next page)

(continued from previous page)

```

True
```

In [2]:
```
audio.path_to_save_ = './models'
audio.chunk_size_ = 2000000

url = audio.weights_for_big5_['audio']['hc']['sberdisk']

audio.load_audio_model_weights_hc(
    url = url,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[2]:
```
[2022-10-17 14:24:30] Downloading the weights of the neural network model to␣
↪obtain scores by hand-crafted features (audio modality) ...

[2022-10-17 14:24:30] File download "weights_2022-05-05_11-27-55.h5" (100.0%) ..
↪.

--- Runtime: 0.414 sec. ---

True
```

Error – 1 –

In [3]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.path_to_save_ = './models'
audio.chunk_size_ = 2000000

url = audio.weights_for_big5_['audio']['hc']['sberdisk']

audio.load_audio_model_weights_hc(
    url = url,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[3]:
```
[2022-10-17 15:21:13] Downloading the weights of the neural network model to␣
↪obtain scores by hand-crafted features (audio modality) ...

[2022-10-17 15:21:14] File download "weights_2022-05-05_11-27-55.h5" (100.0%) ..
↪.
```

(continues on next page)

(continued from previous page)

```
[2022-10-17 15:21:14] Something went wrong ... the neural network architecture␣
↪of the model for obtaining scores by hand-crafted features has not been formed␣
↪(audio modality) ...

--- Runtime: 0.364 sec. ---

False
```

load_audio_model_weights_nn(*url: str*, *force_reload: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Downloading the weights of the neural network model to obtain scores for deep features

Parameters

- url (*str*) – Full path to the file with weights of the neural network model
- force_reload (*bool*) – Forced download of a file with weights of a neural network model from the network
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the weights of the neural network model are downloaded, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:

```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.load_audio_model_nn(
    show_summary = False, out = True,
    runtime = True, run = True
)
```

[1]:

```
[2022-10-17 15:47:20] Formation of a neural network architecture for obtaining␣
↪scores by deep features (audio modality) ...

--- Runtime: 0.419 sec. ---

True
```

In [2]:

```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.path_to_save_ = './models'
audio.chunk_size_ = 2000000

url = audio.weights_for_big5_['audio']['nn']['sberdisk']

audio.load_audio_model_weights_nn(
    url = url,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[2]:

```
[2022-10-17 15:47:22] Downloading the weights of the neural network model to␣
↪obtain scores for deep features (audio modality) ...

[2022-10-17 15:47:26] File download "weights_2022-05-03_07-46-14.h5" (100.0%) ..
↪.

--- Runtime: 3.884 sec. ---

True
```

Error – 1 –

In [3]:

```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.path_to_save_ = './models'
audio.chunk_size_ = 2000000

url = audio.weights_for_big5_['audio']['nn']['sberdisk']

audio.load_audio_model_weights_nn(
    url = url,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[3]:

```
[2022-10-17 15:49:57] Downloading the weights of the neural network model to␣
↪obtain scores for deep features (audio modality) ...

[2022-10-17 15:50:04] File download "weights_2022-05-03_07-46-14.h5" (100.0%) ..
↪.
```

(continues on next page)

(continued from previous page)

```
[2022-10-17 15:50:04] Something went wrong ... the neural network architecture␣
↪of the model for obtaining scores by deep features has not been formed (audio␣
↪modality) ...

--- Runtime: 6.786 sec. ---

False
```

load˙audio˙models˙b5(*show_summary: bool = False*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Formation of neural network architectures of models for obtaining the personality traits scores

Parameters

- show˙summary (*bool*) – Displaying the last generated neural network architecture of models
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the neural network architectures of the model are formed, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')
audio.load_audio_models_b5(
    show_summary = True, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-10-18 11:39:22] Formation of neural network architectures of models for␣
↪obtaining the personality traits scores (audio modality) ...

Model: "model_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32)]              0

 dense_1 (Dense)             (None, 1)                 33

 activ_1 (Activation)        (None, 1)                 0
```

(continues on next page)

(continued from previous page)

```
12
13 ================================================================
14 Total params: 33
15 Trainable params: 33
16 Non-trainable params: 0
17 ----------------------------------------------------------------
18 --- Runtime: 0.163 sec. ---
19
20 True
```

Error – 1 –

In [2]:

```
1 from oceanai.modules.lab.audio import Audio
2
3 audio = Audio(lang = 'en')
4 audio.load_audio_models_b5(
5     show_summary = 1, out = True,
6     runtime = True, run = True
7 )
```

[2]:

```
1 [2022-10-18 13:47:36] Invalid argument types or values in "Audio.load_audio_
  ↪models_b5" ...
2
3 False
```

load_audio_models_weights_b5(*url_openness: str*, *url_conscientiousness: str*, *url_extraversion: str*, *url_agreeableness: str*, *url_non_neuroticism: str*, *force_reload: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Downloading the weights of neural network models to obtain the personality traits scores

Parameters

- url_openness (*str*) – Full path to the file with the weights of the neural network model (openness)
- url_conscientiousness (*str*) – Full path to the file with the weights of the neural network model (conscientiousness)
- url_extraversion (*str*) – Full path to the file with the weights of the neural network model (extraversion)
- url_agreeableness (*str*) – Full path to the file with the weights of the neural network model (agreeableness)
- url_non_neuroticism (*str*) – Full path to the file with the weights of the neural network model (non-neuroticism)
- force_reload (*bool*) – Forced download of files with weights of neural network models from the network
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns
True if the weights of the neural network models are downloaded, otherwise **False**

Return type
bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.load_audio_models_b5(
    show_summary = False, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-10-18 22:40:05] Formation of neural network architectures of models for␣
↪obtaining the personality traits scores (audio modality) ...

--- Runtime: 0.163 sec. ---

True
```

In [2]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.path_to_save_ = './models'
audio.chunk_size_ = 2000000

url_openness = audio.weights_for_big5_['audio']['b5']['openness']['sberdisk']
url_conscientiousness = audio.weights_for_big5_['audio']['b5']['
↪'conscientiousness']['sberdisk']
url_extraversion = audio.weights_for_big5_['audio']['b5']['extraversion']['
↪'sberdisk']
url_agreeableness = audio.weights_for_big5_['audio']['b5']['agreeableness']['
↪'sberdisk']
url_non_neuroticism = audio.weights_for_big5_['audio']['b5']['non_neuroticism']['
↪'sberdisk']

audio.load_audio_models_weights_b5(
    url_openness = url_openness,
    url_conscientiousness = url_conscientiousness,
    url_extraversion = url_extraversion,
    url_agreeableness = url_agreeableness,
    url_non_neuroticism = url_non_neuroticism,
    force_reload = True,
    out = True,
    runtime = True,
```

(continues on next page)

(continued from previous page)

```
    run = True
)
```

[2]:

```
[2022-10-18 23:08:37] Downloading the weights of neural network models to␣
↪obtain the personality traits scores (audio modality) ...

[2022-10-18 23:08:37] File download "weights_2022-06-15_16-16-20.h5" (100.0%) ..
↪. Openness

[2022-10-18 23:08:38] File download "weights_2022-06-15_16-21-57.h5" (100.0%) ..
↪. Conscientiousness

[2022-10-18 23:08:38] File download "weights_2022-06-15_16-26-41.h5" (100.0%) ..
↪. Extraversion

[2022-10-18 23:08:38] File download "weights_2022-06-15_16-32-51.h5" (100.0%) ..
↪. Agreeableness

[2022-10-18 23:08:39] File download "weights_2022-06-15_16-37-46.h5" (100.0%) ..
↪. Non-Neuroticism

--- Runtime: 1.611 sec. ---

True
```

Error – 1 –

In [3]:

```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')

audio.path_to_save_ = './models'
audio.chunk_size_ = 2000000

url_openness = audio.weights_for_big5_['audio']['b5']['openness']['sberdisk']
url_conscientiousness = audio.weights_for_big5_['audio']['b5']['
↪'conscientiousness']['sberdisk']
url_extraversion = audio.weights_for_big5_['audio']['b5']['extraversion']['
↪'sberdisk']
url_agreeableness = audio.weights_for_big5_['audio']['b5']['agreeableness']['
↪'sberdisk']
url_non_neuroticism = audio.weights_for_big5_['audio']['b5']['non_neuroticism']['
↪'sberdisk']

audio.load_audio_models_weights_b5(
    url_openness = url_openness,
    url_conscientiousness = url_conscientiousness,
    url_extraversion = url_extraversion,
    url_agreeableness = url_agreeableness,
    url_non_neuroticism = url_non_neuroticism,
    force_reload = True,
```

(continues on next page)

(continued from previous page)

```
    out = True,
    runtime = True,
    run = True
)
```

[3]:

```
[2022-10-18 23:09:40] Downloading the weights of neural network models to␣
↪obtain the personality traits scores (audio modality) ...

[2022-10-18 23:09:41] File download "weights_2022-06-15_16-16-20.h5" (100.0%) ..
↪.

[2022-10-18 23:09:41] Something went wrong ... ailed to load neural network␣
↪model weights ... Openness

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/audio.py
    Line: 1764
    Method: load_models_weights_b5
    Error type: AttributeError

[2022-10-18 23:09:41] File download "weights_2022-06-15_16-21-57.h5" (100.0%) ..
↪.

[2022-10-18 23:09:41] Something went wrong ... ailed to load neural network␣
↪model weights ... Conscientiousness

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/audio.py
    Line: 1764
    Method: load_models_weights_b5
    Error type: AttributeError

[2022-10-18 23:09:41] File download "weights_2022-06-15_16-26-41.h5" (100.0%) ..
↪.

[2022-10-18 23:09:41] Something went wrong ... ailed to load neural network␣
↪model weights ... Extraversion

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/audio.py
    Line: 1764
    Method: load_models_weights_b5
    Error type: AttributeError

[2022-10-18 23:09:42] File download "weights_2022-06-15_16-32-51.h5" (100.0%) ..
↪.

[2022-10-18 23:09:42] Something went wrong ... ailed to load neural network␣
↪model weights ... Agreeableness

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/audio.py
    Line: 1764
    Method: load_models_weights_b5
    Error type: AttributeError
```

(continues on next page)

(continued from previous page)

```
[2022-10-18 23:09:42] File download "weights_2022-06-15_16-37-46.h5" (100.0%) ...

[2022-10-18 23:09:42] Something went wrong ... ailed to load neural network model weights ... Non-Neuroticism

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/audio.py
    Line: 1764
    Method: load_models_weights_b5
    Error type: AttributeError

--- Runtime: 1.573 sec. ---

False
```

property smile: Smile

Getting OpenSmile functions

Returns
Extracted OpenSmile features

Return type
opensmile.core.smile.Smile

Example

True – 1 –

In [1]:
```
from oceanai.modules.lab.audio import Audio

audio = Audio(lang = 'en')
audio.smile_
```

[1]:
```
{
    '$opensmile.core.smile.Smile': {
        'feature_set': 'eGeMAPSv02',
        'feature_level': 'LowLevelDescriptors',
        'options': {},
        'sampling_rate': None,
        'channels': [0],
        'mixdown': False,
        'resample': False
    }
}
```

Video

class oceanai.modules.lab.video.VideoMessages(*lang: str = 'ru'*, *color_simple: str = '#666'*, *color_info: str = '#1776D2'*, *color_err: str = '#FF0000'*, *color_true: str = '#008001'*, *bold_text: bool = True*, *text_runtime: str = ''*, *num_to_df_display: int = 30*)

Bases: *Download*

Class for messages

Parameters

- lang (*str*) – See *lang*
- color_simple (*str*) – See *color_simple*
- color_info (*str*) – See *color_info*
- color_err (*str*) – See *color_err*
- color_true (*str*) – See *color_true*
- bold_text (*bool*) – See *bold_text*
- num_to_df_display (*int*) – See *num_to_df_display*
- text_runtime (*str*) – See *text_runtime*

class oceanai.modules.lab.video.Video(*lang: str = 'ru'*, *color_simple: str = '#666'*, *color_info: str = '#1776D2'*, *color_err: str = '#FF0000'*, *color_true: str = '#008001'*, *bold_text: bool = True*, *text_runtime: str = ''*, *num_to_df_display: int = 30*)

Bases: *VideoMessages*

Video processing class

Parameters

- lang (*str*) – See *lang*
- color_simple (*str*) – See *color_simple*
- color_info (*str*) – See *color_info*
- color_err (*str*) – See *color_err*
- color_true (*str*) – See *color_true*
- bold_text (*bool*) – See *bold_text*
- num_to_df_display (*int*) – See *num_to_df_display*
- text_runtime (*str*) – See *text_runtime*

__calc_reshape_img_coef(*shape: Union[Tuple[int], List[int]]*, *new_shape: Union[int, Tuple[int], List[int]]*, *out: bool = True*) → float

Calculating the image resizing factor

Note: private method

Parameters

- shape (*Union[Tuple[int], List[int]]*) – Current image size (width, height)
- new_shape (*Union[int, Tuple[int], List[int]]*) – Desired image size
- out (*bool*) – Display

Returns
Image resizing factor

Return type
float

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video()

video._Video__calc_reshape_img_coef(
    shape = (1280, 720),
    new_shape = 224,
    out = True
)
```

[1]:
```
0.175
```

True – 2 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video()

video._Video__calc_reshape_img_coef(
    shape = (1280, 720),
    new_shape = (1920, 1080),
    out = True
)
```

[1]:
```
1.5
```

Error – 1 –

In [3]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video._Video__calc_reshape_img_coef(
    shape = (1280, 720),
    new_shape = '',
    out = True
)
```

[4]:
```
[2022-10-29 13:24:27] Invalid argument types or values in "Video.__calc_reshape_
↪img_coef" ...

-1.0
```

¨concat˙pred(*pred_hc: ndarray*, *pred_nn: ndarray*, *out: bool = True*) → List[Optional[ndarray]]

Concatenation of scores by hand-crafted and deep features

Note: private method

Parameters

- pred˙hc (*np.ndarray*) – Scores on hand-crafted features
- pred˙nn (*np.ndarray*) – Scores on deep features
- out (*bool*) – Display

Returns

Concatenated scores by hand-crafted and deep features

Return type

List[Optional[np.ndarray]]

Examples

True – 1 –

In [1]:
```
import numpy as np
from oceanai.modules.lab.video import Video

video = Video()

arr_hc = np.array([
    [0.64113516, 0.6217892, 0.54451424, 0.6144415, 0.59334993],
    [0.6652424, 0.63606125, 0.572305, 0.63169795, 0.612515]
])

arr_nn = np.array([
    [0.56030345, 0.7488746, 0.44648764, 0.59893465, 0.5701077],
    [0.5900006, 0.7652722, 0.4795154, 0.6409055, 0.6088242]
])

video._Video__concat_pred(
    pred_hc = arr_hc,
    pred_nn = arr_nn,
    out = True
)
```

[1]:
```
[
    array([
            0.64113516, 0.6652424, 0.65318878, 0.65318878, 0.65318878,
```

(continues on next page)

(continued from previous page)

```
            0.65318878, 0.65318878, 0.65318878, 0.65318878, 0.65318878,
            0.65318878, 0.65318878, 0.65318878, 0.65318878, 0.65318878,
            0.65318878, 0.56030345, 0.5900006, 0.57515202, 0.57515202,
            0.57515202, 0.57515202, 0.57515202, 0.57515202, 0.57515202,
            0.57515202, 0.57515202, 0.57515202, 0.57515202, 0.57515202,
            0.57515202, 0.57515202
    ]),
    array([
            0.6217892, 0.63606125, 0.62892523, 0.62892523, 0.62892523,
            0.62892523, 0.62892523, 0.62892523, 0.62892523, 0.62892523,
            0.62892523, 0.62892523, 0.62892523, 0.62892523, 0.62892523,
            0.62892523, 0.7488746, 0.7652722, 0.7570734, 0.7570734,
            0.7570734, 0.7570734, 0.7570734, 0.7570734, 0.7570734,
            0.7570734, 0.7570734, 0.7570734, 0.7570734, 0.7570734,
            0.7570734, 0.7570734
    ]),
    array([
            0.54451424, 0.572305, 0.55840962, 0.55840962, 0.55840962,
            0.55840962, 0.55840962, 0.55840962, 0.55840962, 0.55840962,
            0.55840962, 0.55840962, 0.55840962, 0.55840962, 0.55840962,
            0.55840962, 0.44648764, 0.4795154, 0.46300152, 0.46300152,
            0.46300152, 0.46300152, 0.46300152, 0.46300152, 0.46300152,
            0.46300152, 0.46300152, 0.46300152, 0.46300152, 0.46300152,
            0.46300152, 0.46300152
    ]),
    array([
            0.6144415, 0.63169795, 0.62306972, 0.62306972, 0.62306972,
            0.62306972, 0.62306972, 0.62306972, 0.62306972, 0.62306972,
            0.62306972, 0.62306972, 0.62306972, 0.62306972, 0.62306972,
            0.62306972, 0.59893465, 0.6409055, 0.61992008, 0.61992008,
            0.61992008, 0.61992008, 0.61992008, 0.61992008, 0.61992008,
            0.61992008, 0.61992008, 0.61992008, 0.61992008, 0.61992008,
            0.61992008, 0.61992008
    ]),
    array([
            0.59334993, 0.612515, 0.60293247, 0.60293247, 0.60293247,
            0.60293247, 0.60293247, 0.60293247, 0.60293247, 0.60293247,
            0.60293247, 0.60293247, 0.60293247, 0.60293247, 0.60293247,
            0.60293247, 0.5701077, 0.6088242, 0.58946595, 0.58946595,
            0.58946595, 0.58946595, 0.58946595, 0.58946595, 0.58946595,
            0.58946595, 0.58946595, 0.58946595, 0.58946595, 0.58946595,
            0.58946595, 0.58946595
    ])
]
```

Error – 1 –

In [2]:

```
import numpy as np
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

```

(continues on next page)

(continued from previous page)

```
arr_hc = np.array([
    [0.64113516, 0.6217892, 0.54451424, 0.6144415],
    [0.6652424, 0.63606125, 0.572305, 0.63169795, 0.612515]
])

arr_nn = np.array([
    [0.56030345, 0.7488746, 0.44648764, 0.59893465, 0.5701077],
    [0.5900006, 0.7652722, 0.4795154, 0.6409055, 0.6088242]
])

video._Video__concat_pred(
    pred_hc = arr_hc,
    pred_nn = arr_nn,
    out = True
)
```

[3]:

```
[2022-10-20 22:33:31] Ouch! Something went wrong ... concatenation of the␣
↪scores by hand-crafted and deep features was not performed (video modality) ..
↪.

[]
```

¨load˙model˙weights(*url: str*, *force_reload: bool = True*, *info_text: str = ''*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Downloading the weights of the neural network model

Note: private method

Parameters

- url (*str*) – Full path to the file with weights of the neural network model
- force˙reload (*bool*) – Forced download of a file with weights of a neural network model from the network
- info˙text (*str*) – Text for informational message
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the weights of the neural network model are downloaded, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

video._Video__load_model_weights(
    url = 'https://download.sberdisk.ru/download/file/412059444?
↪token=JXerCfAjJZg6crD&filename=weights_2022-08-27_18-53-35.h5',
    force_reload = True,
    info_text = 'Downloading the weights of the neural network model',
    out = True, runtime = True, run = True
)
```

[1]:
```
[2022-10-27 12:46:55] Downloading the weights of the neural network model

[2022-10-27 12:46:55] File download "weights_2022-08-27_18-53-35.h5" (100.0%) ..
↪.

--- Runtime: 0.626 sec. ---

True
```

– 2 –

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

video._Video__load_model_weights(
    url = './models/weights_2022-08-27_18-53-35.h5',
    force_reload = True,
    info_text = 'Downloading the weights of the neural network model',
    out = True, runtime = True, run = True
)
```

[2]:
```
[2022-10-27 12:47:52] Downloading the weights of the neural network model

--- Runtime: 0.002 sec. ---

True
```

Error – 1 –

In [3]:

```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

video._Video__load_model_weights(
    url = 'https://download.sberdisk.ru/download/file/412059444?
↪token=JXerCfAjJZg6crD&filename=weights_2022-08-27_18-53-35.h5',
    force_reload = True, info_text = '',
    out = True, runtime = True, run = True
)
```

[3]:

```
[2022-10-27 12:48:24] Invalid argument types or values in "Video.__load_model_
↪weights" ...

False
```

¨load˙video˙model˙b5(*show_summary: bool = False*, *out: bool = True*) → Optional[Model]

Formation of the neural network architecture of the model to obtain the personality traits scores

Note: private method

Parameters

- show˙summary (*bool*) – Displaying the formed neural network architecture of the model
- out (*bool*) – Display

Returns

None if the types or values of the arguments are invalid, otherwise the neural network model **tf.keras.Model** to get the personality traits scores

Return type

Optional[tf.keras.Model]

Examples

True – 1 –

In [1]:

```
from oceanai.modules.lab.video import Video

video = Video()

video._Video__load_video_model_b5(
    show_summary = True, out = True
)
```

[1]:

```
[2022-11-04 15:29:26] Formation of neural network architectures of models for␣
↪obtaining the personality traits scores (video modality) ...

Model: "model_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32)]              0

 dense_1 (Dense)             (None, 1)                 33

 activ_1 (Activation)        (None, 1)                 0

=================================================================
Total params: 33
Trainable params: 33
Non-trainable params: 0
_________________________________________________________________
--- Runtime: 0.116 sec. ---

True
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video._Video__load_video_model_b5(
    show_summary = True, out = []
)
```

[3]:
```
[2022-10-17 10:53:03] Invalid argument types or values in "Video.__load_video_
↪model_b5" ...
```

¨norm˙pred(*pred_data: ndarray*, *len_nn: int = 16*, *out: bool = True*) → ndarray

Normalization of scores by hand-crafted and deep features

Note: private method

Parameters

- pred˙data (*np.ndarray*) – Scores
- len˙nn (*int*) – The maximum size of the scores vector
- out (*bool*) – Display

Returns

Normalized scores by hand-crafted and deep features

Return type

np.ndarray

Examples

True – 1 –

In [1]:
```
import numpy as np
from oceanai.modules.lab.video import Video

video = Video()

arr = np.array([
    [0.64113516, 0.6217892, 0.54451424, 0.6144415, 0.59334993],
    [0.6652424, 0.63606125, 0.572305, 0.63169795, 0.612515]
])

video._Video__norm_pred(
    pred_data = arr,
    len_nn = 4,
    out = True
)
```

[1]:
```
array([
    [0.64113516, 0.6217892 , 0.54451424, 0.6144415 , 0.59334993],
    [0.6652424 , 0.63606125, 0.572305  , 0.63169795, 0.612515],
    [0.65318878, 0.62892523, 0.55840962, 0.62306972, 0.60293247],
    [0.65318878, 0.62892523, 0.55840962, 0.62306972, 0.60293247]
])
```

Error – 1 –

In [2]:
```
import numpy as np
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

arr = np.array([])

video._Video__norm_pred(
    pred_data = arr,
    len_nn = 4,
    out = True
)
```

[3]:
```
[2022-10-20 22:03:17] Invalid argument types or values in "Video.__norm_pred" ...

array([], dtype=float64)
```

˙get˙visual˙features(*path: str*, *reduction_fps: int = 5*, *window: int = 10*, *step: int = 5*, *lang: str = 'ru'*, *last: bool = False*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → Tuple[ndarray, ndarray]

Extracting features from a visual signal (without clearing the message output history in a Jupyter cell)

Note: protected method

Parameters

- path (*str*) – Path to video file
- reduction˙fps (*int*) – Frame rate reduction
- window (*int*) – Signal segment window size (in frames)
- step (*int*) – Signal segment window shift step (frames)
- lang (*str*) – Language
- last (*bool*) – Replacing the last message
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

Tuple with two np.ndarray: 1. np.ndarray with hand-crafted features 2. np.ndarray with deep features

Return type

Tuple[np.ndarray, np.ndarray]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

res_load_model_deep_fe = video.load_video_model_deep_fe(
    show_summary = False,
    out = True,
    runtime = True,
    run = True
)
```

[1]:
```
[2022-11-03 16:37:12] Formation of neural network architecture for obtaining␣
↪deep features (video modality) ...

--- Runtime: 1.564 sec. ---
```

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000
```

(continues on next page)

(continued from previous page)

```

url = video.weights_for_big5_['video']['fe']['sberdisk']

res_load_video_model_weights_deep_fe = video.load_video_model_weights_deep_fe(
    url = url,
    force_reload = True, out = True,
    runtime = True, run = True
)
```

[2]:

```
[2022-11-03 16:39:10] Downloading weights of a neural network model to obtain deep features (video modality) ...

[2022-11-03 16:39:14] File download "weights_2022-11-01_12-27-07.h5" (100.0%) ...

--- Runtime: 4.874 sec. ---
```

In [3]:

```
path = '/Users/dl/GitHub/oceanai/oceanai/dataset/test80_01/glgfB3vFewc.004.mp4'

hc_features, nn_features = video.get_visual_features(
    path = path, reduction_fps = 5,
    window = 10, step = 5,
    out = True, runtime = True, run = True
)
```

[3]:

```
[2022-11-03 16:56:52] Extraction of features (hand-crafted and deep) from a visual signal ...

[2022-11-03 16:56:58] Statistics of extracted features from visual signal:
    Total number of segments since:
    1. hand-crafted features: 12
    2. deep features: 12
    Dimension of the matrix with hand-crafted features of one segment: 10 × 115
    Dimension of the tensor with deep features of one segment: 10 × 512
    FPS down: with 30 to 5

--- Runtime: 6.109 sec. ---
```

Error – 1 –

In [4]:

```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

path = '/Users/dl/GitHub/oceanai/oceanai/dataset/test80_01/glgfB3vFewc.004.mp4'

hc_features, nn_features = video.get_visual_features(
    path = path, reduction_fps = 5,
    window = 10, step = 5,
    out = True, runtime = True, run = True
```

(continues on next page)

(continued from previous page)

```
)
```

[4]:

```
[2022-11-03 16:59:45] Extraction of features (hand-crafted and deep) from a visual signal ...

[2022-11-03 16:59:46] Ouch! Something went wrong ... the neural network architecture of the model for obtaining deep features is not formed (video modality) ...

--- Runtime: 1.358 sec. ---
```

get_video_union_predictions(*depth: int = 1*, *recursive: bool = False*, *reduction_fps: int = 5*, *window: int = 10*, *step: int = 5*, *lang: str = 'ru'*, *accuracy=True*, *url_accuracy: str = ''*, *logs: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Get video scores

Parameters

- depth (*int*) – Hierarchy depth for getting data
- recursive (*bool*) – Recursive data search
- reduction_fps (*int*) – Frame rate reduction
- window (*int*) – Signal segment window size (in frames)
- step (*int*) – Signal segment window shift step (frames)
- lang (*str*) – Language
- accuracy (*bool*) – Accuracy calculation
- url_accuracy (*str*) – Full path to the file with ground truth scores for calculating accuracy
- logs (*bool*) – If necessary, generate a LOG file
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if scores are successfully received, otherwise **False**

Return type

bool

Example

get_visual_features(*path: str*, *reduction_fps: int = 5*, *window: int = 10*, *step: int = 5*, *lang: str = 'ru'*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → Tuple[ndarray, ndarray]

Extracting features from a visual signal

Parameters

- path (*str*) – Path to video file

- reduction_fps (*int*) – Frame rate reduction
- window (*int*) – Signal segment window size (in frames)
- step (*int*) – Signal segment window shift step (frames)
- lang (*str*) – Language
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns
Tuple with two np.ndarray: 1. np.ndarray with hand-crafted features 2. np.ndarray with deep features

Return type
Tuple[np.ndarray, np.ndarray]

Example

load_video_model_deep_fe(*show_summary: bool = False*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Formation of neural network architecture for obtaining neural network features

Parameters

- show_summary (*bool*) – Displaying the formed neural network architecture of the model
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns
True if the neural network architecture of the model is formed, otherwise **False**

Return type
bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')
video.load_video_model_deep_fe(
    show_summary = True, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-11-01 12:18:14] Formation of neural network architecture for obtaining␣
↪deep features (video modality) ...

Model: "model_1"
```

(continues on next page)

(continued from previous page)

```
__________________________________________________________________________________
↪__________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================
input_2 (InputLayer)            [(None, 224, 224, 3  0           []
                )]

conv1/7x7_s2 (Conv2D)           (None, 112, 112, 64  9408        ['input_2[0][0]
↪']
                )

conv1/7x7_s2/bn (BatchNormaliz  (None, 112, 112, 64  256         ['conv1/7x7_
↪s2[0][0]']
ation)                          )

activation_49 (Activation)      (None, 112, 112, 64  0           ['conv1/7x7_s2/
↪bn[0][0]']
                )

max_pooling2d_1 (MaxPooling2D)  (None, 55, 55, 64)  0           ['activation_
↪49[0][0]']

conv2_1_1x1_reduce (Conv2D)     (None, 55, 55, 64)   4096        ['max_
↪pooling2d_1[0][0]']

conv2_1_1x1_reduce/bn (BatchNo  (None, 55, 55, 64)  256         ['conv2_1_1x1_
↪reduce[0][0]']
rmalization)

activation_50 (Activation)      (None, 55, 55, 64)   0           ['conv2_1_1x1_
↪reduce/bn[0][0]']

conv2_1_3x3 (Conv2D)            (None, 55, 55, 64)   36864       ['activation_
↪50[0][0]']

conv2_1_3x3/bn (BatchNormaliza  (None, 55, 55, 64)  256         ['conv2_1_
↪3x3[0][0]']
tion)

activation_51 (Activation)      (None, 55, 55, 64)   0           ['conv2_1_3x3/
↪bn[0][0]']

conv2_1_1x1_increase (Conv2D)   (None, 55, 55, 256)  16384       ['activation_
↪51[0][0]']

conv2_1_1x1_proj (Conv2D)       (None, 55, 55, 256)  16384       ['max_
↪pooling2d_1[0][0]']

conv2_1_1x1_increase/bn (Batch  (None, 55, 55, 256)  1024        ['conv2_1_1x1_
↪increase[0][0]']
Normalization)

```

(continues on next page)

(continued from previous page)

```
conv2_1_1x1_proj/bn (BatchNorm  (None, 55, 55, 256)  1024        ['conv2_1_1x1_
↪proj[0][0]']
alization)

add_16 (Add)                    (None, 55, 55, 256)  0           ['conv2_1_1x1_
↪increase/bn[0][0]',
                                 'conv2_1_1x1_proj/bn[0][0]']

activation_52 (Activation)      (None, 55, 55, 256)  0           ['add_16[0][0]
↪']

conv2_2_1x1_reduce (Conv2D)     (None, 55, 55, 64)   16384       ['activation_
↪52[0][0]']

conv2_2_1x1_reduce/bn (BatchNo  (None, 55, 55, 64)  256         ['conv2_2_1x1_
↪reduce[0][0]']
rmalization)

activation_53 (Activation)      (None, 55, 55, 64)   0           ['conv2_2_1x1_
↪reduce/bn[0][0]']

conv2_2_3x3 (Conv2D)            (None, 55, 55, 64)   36864       ['activation_
↪53[0][0]']

conv2_2_3x3/bn (BatchNormaliza  (None, 55, 55, 64)  256         ['conv2_2_
↪3x3[0][0]']
tion)

activation_54 (Activation)      (None, 55, 55, 64)   0           ['conv2_2_3x3/
↪bn[0][0]']

conv2_2_1x1_increase (Conv2D)   (None, 55, 55, 256)  16384       ['activation_
↪54[0][0]']

conv2_2_1x1_increase/bn (Batch  (None, 55, 55, 256)  1024        ['conv2_2_1x1_
↪increase[0][0]']
Normalization)

add_17 (Add)                    (None, 55, 55, 256)  0           ['conv2_2_1x1_
↪increase/bn[0][0]',
                                 'activation_52[0][0]']

activation_55 (Activation)      (None, 55, 55, 256)  0           ['add_17[0][0]
↪']

conv2_3_1x1_reduce (Conv2D)     (None, 55, 55, 64)   16384       ['activation_
↪55[0][0]']

conv2_3_1x1_reduce/bn (BatchNo  (None, 55, 55, 64)  256         ['conv2_3_1x1_
↪reduce[0][0]']
rmalization)

```

(continues on next page)

(continued from previous page)

```
activation_56 (Activation)      (None, 55, 55, 64)   0           ['conv2_3_1x1_
↪reduce/bn[0][0]']

conv2_3_3x3 (Conv2D)            (None, 55, 55, 64)   36864       ['activation_
↪56[0][0]']

conv2_3_3x3/bn (BatchNormaliza  (None, 55, 55, 64)   256         ['conv2_3_
↪3x3[0][0]']
tion)

activation_57 (Activation)      (None, 55, 55, 64)   0           ['conv2_3_3x3/
↪bn[0][0]']

conv2_3_1x1_increase (Conv2D)   (None, 55, 55, 256)  16384       ['activation_
↪57[0][0]']

conv2_3_1x1_increase/bn (Batch  (None, 55, 55, 256)  1024        ['conv2_3_1x1_
↪increase[0][0]']
Normalization)

add_18 (Add)                    (None, 55, 55, 256)  0           ['conv2_3_1x1_
↪increase/bn[0][0]',
                                 'activation_55[0][0]']

activation_58 (Activation)      (None, 55, 55, 256)  0           ['add_18[0][0]
↪']

conv3_1_1x1_reduce (Conv2D)     (None, 28, 28, 128)  32768       ['activation_
↪58[0][0]']

conv3_1_1x1_reduce/bn (BatchNo  (None, 28, 28, 128)  512         ['conv3_1_1x1_
↪reduce[0][0]']
rmalization)

activation_59 (Activation)      (None, 28, 28, 128)  0           ['conv3_1_1x1_
↪reduce/bn[0][0]']

conv3_1_3x3 (Conv2D)            (None, 28, 28, 128)  147456      ['activation_
↪59[0][0]']

conv3_1_3x3/bn (BatchNormaliza  (None, 28, 28, 128)  512         ['conv3_1_
↪3x3[0][0]']
tion)

activation_60 (Activation)      (None, 28, 28, 128)  0           ['conv3_1_3x3/
↪bn[0][0]']

conv3_1_1x1_increase (Conv2D)   (None, 28, 28, 512)  65536       ['activation_
↪60[0][0]']

conv3_1_1x1_proj (Conv2D)       (None, 28, 28, 512)  131072      ['activation_
↪58[0][0]']
```

(continues on next page)

(continued from previous page)

```
115
116 conv3_1_1x1_increase/bn (Batch  (None, 28, 28, 512)  2048       ['conv3_1_1x1_
    ↪increase[0][0]']
117 Normalization)
118
119 conv3_1_1x1_proj/bn (BatchNorm  (None, 28, 28, 512)  2048       ['conv3_1_1x1_
    ↪proj[0][0]']
120 alization)
121
122 add_19 (Add)                  (None, 28, 28, 512)  0           ['conv3_1_1x1_
    ↪increase/bn[0][0]',
123                                 'conv3_1_1x1_proj/bn[0][0]']
124
125 activation_61 (Activation)     (None, 28, 28, 512)  0           ['add_19[0][0]
    ↪']
126
127 conv3_2_1x1_reduce (Conv2D)    (None, 28, 28, 128)  65536       ['activation_
    ↪61[0][0]']
128
129 conv3_2_1x1_reduce/bn (BatchNo  (None, 28, 28, 128)  512        ['conv3_2_1x1_
    ↪reduce[0][0]']
130 rmalization)
131
132 activation_62 (Activation)     (None, 28, 28, 128)  0           ['conv3_2_1x1_
    ↪reduce/bn[0][0]']
133
134 conv3_2_3x3 (Conv2D)          (None, 28, 28, 128)  147456      ['activation_
    ↪62[0][0]']
135
136 conv3_2_3x3/bn (BatchNormaliza  (None, 28, 28, 128)  512        ['conv3_2_
    ↪3x3[0][0]']
137 tion)
138
139 activation_63 (Activation)     (None, 28, 28, 128)  0           ['conv3_2_3x3/
    ↪bn[0][0]']
140
141 conv3_2_1x1_increase (Conv2D)  (None, 28, 28, 512)  65536       ['activation_
    ↪63[0][0]']
142
143 conv3_2_1x1_increase/bn (Batch  (None, 28, 28, 512)  2048       ['conv3_2_1x1_
    ↪increase[0][0]']
144 Normalization)
145
146 add_20 (Add)                  (None, 28, 28, 512)  0           ['conv3_2_1x1_
    ↪increase/bn[0][0]',
147                                 'activation_61[0][0]']
148
149 activation_64 (Activation)     (None, 28, 28, 512)  0           ['add_20[0][0]
    ↪']
150
151 conv3_3_1x1_reduce (Conv2D)    (None, 28, 28, 128)  65536       ['activation_
    ↪64[0][0]']
```

(continues on next page)

(continued from previous page)

```
conv3_3_1x1_reduce/bn (BatchNo  (None, 28, 28, 128)  512         ['conv3_3_1x1_
↪reduce[0][0]']
rmalization)

activation_65 (Activation)     (None, 28, 28, 128)  0           ['conv3_3_1x1_
↪reduce/bn[0][0]']

conv3_3_3x3 (Conv2D)           (None, 28, 28, 128)  147456      ['activation_
↪65[0][0]']

conv3_3_3x3/bn (BatchNormaliza  (None, 28, 28, 128)  512         ['conv3_3_
↪3x3[0][0]']
tion)

activation_66 (Activation)     (None, 28, 28, 128)  0           ['conv3_3_3x3/
↪bn[0][0]']

conv3_3_1x1_increase (Conv2D)  (None, 28, 28, 512)  65536       ['activation_
↪66[0][0]']

conv3_3_1x1_increase/bn (Batch  (None, 28, 28, 512)  2048        ['conv3_3_1x1_
↪increase[0][0]']
Normalization)

add_21 (Add)                   (None, 28, 28, 512)  0           ['conv3_3_1x1_
↪increase/bn[0][0]',
                                 'activation_64[0][0]']

activation_67 (Activation)     (None, 28, 28, 512)  0           ['add_21[0][0]
↪']

conv3_4_1x1_reduce (Conv2D)    (None, 28, 28, 128)  65536       ['activation_
↪67[0][0]']

conv3_4_1x1_reduce/bn (BatchNo  (None, 28, 28, 128)  512         ['conv3_4_1x1_
↪reduce[0][0]']
rmalization)

activation_68 (Activation)     (None, 28, 28, 128)  0           ['conv3_4_1x1_
↪reduce/bn[0][0]']

conv3_4_3x3 (Conv2D)           (None, 28, 28, 128)  147456      ['activation_
↪68[0][0]']

conv3_4_3x3/bn (BatchNormaliza  (None, 28, 28, 128)  512         ['conv3_4_
↪3x3[0][0]']
tion)

activation_69 (Activation)     (None, 28, 28, 128)  0           ['conv3_4_3x3/
↪bn[0][0]']

```

(continues on next page)

(continued from previous page)

```
conv3_4_1x1_increase (Conv2D)  (None, 28, 28, 512)  65536       ['activation_
↪69[0][0]']

conv3_4_1x1_increase/bn (Batch  (None, 28, 28, 512)  2048       ['conv3_4_1x1_
↪increase[0][0]']
Normalization)

add_22 (Add)                   (None, 28, 28, 512)  0           ['conv3_4_1x1_
↪increase/bn[0][0]',
                                 'activation_67[0][0]']

activation_70 (Activation)     (None, 28, 28, 512)  0           ['add_22[0][0]
↪']

conv4_1_1x1_reduce (Conv2D)    (None, 14, 14, 256)  131072      ['activation_
↪70[0][0]']

conv4_1_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024       ['conv4_1_1x1_
↪reduce[0][0]']
rmalization)

activation_71 (Activation)     (None, 14, 14, 256)  0           ['conv4_1_1x1_
↪reduce/bn[0][0]']

conv4_1_3x3 (Conv2D)           (None, 14, 14, 256)  589824      ['activation_
↪71[0][0]']

conv4_1_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_1_
↪3x3[0][0]']
tion)

activation_72 (Activation)     (None, 14, 14, 256)  0           ['conv4_1_3x3/
↪bn[0][0]']

conv4_1_1x1_increase (Conv2D)  (None, 14, 14, 1024  262144      ['activation_
↪72[0][0]']
                )

conv4_1_1x1_proj (Conv2D)      (None, 14, 14, 1024  524288      ['activation_
↪70[0][0]']
                )

conv4_1_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_1_1x1_
↪increase[0][0]']
Normalization)                 )

conv4_1_1x1_proj/bn (BatchNorm  (None, 14, 14, 1024  4096       ['conv4_1_1x1_
↪proj[0][0]']
alization)                     )

add_23 (Add)                   (None, 14, 14, 1024  0           ['conv4_1_1x1_
↪increase/bn[0][0]',
```

(continues on next page)

(continued from previous page)

```
                )                                    'conv4_1_1x1_proj/bn[0][0]']

 activation_73 (Activation)     (None, 14, 14, 1024  0           ['add_23[0][0]
↪']
                )

 conv4_2_1x1_reduce (Conv2D)    (None, 14, 14, 256)  262144      ['activation_
↪73[0][0]']

 conv4_2_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024       ['conv4_2_1x1_
↪reduce[0][0]']
 rmalization)

 activation_74 (Activation)     (None, 14, 14, 256)  0           ['conv4_2_1x1_
↪reduce/bn[0][0]']

 conv4_2_3x3 (Conv2D)           (None, 14, 14, 256)  589824      ['activation_
↪74[0][0]']

 conv4_2_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_2_
↪3x3[0][0]']
 tion)

 activation_75 (Activation)     (None, 14, 14, 256)  0           ['conv4_2_3x3/
↪bn[0][0]']

 conv4_2_1x1_increase (Conv2D)  (None, 14, 14, 1024  262144      ['activation_
↪75[0][0]']
                )

 conv4_2_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_2_1x1_
↪increase[0][0]']
 Normalization)                 )

 add_24 (Add)                   (None, 14, 14, 1024  0           ['conv4_2_1x1_
↪increase/bn[0][0]',
                )                                  'activation_73[0][0]']

 activation_76 (Activation)     (None, 14, 14, 1024  0           ['add_24[0][0]
↪']
                )

 conv4_3_1x1_reduce (Conv2D)    (None, 14, 14, 256)  262144      ['activation_
↪76[0][0]']

 conv4_3_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024       ['conv4_3_1x1_
↪reduce[0][0]']
 rmalization)

 activation_77 (Activation)     (None, 14, 14, 256)  0           ['conv4_3_1x1_
↪reduce/bn[0][0]']

```

(continues on next page)

(continued from previous page)

```
conv4_3_3x3 (Conv2D)           (None, 14, 14, 256)  589824      ['activation_
↪77[0][0]']

conv4_3_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_3_
↪3x3[0][0]']
tion)

activation_78 (Activation)     (None, 14, 14, 256)  0           ['conv4_3_3x3/
↪bn[0][0]']

conv4_3_1x1_increase (Conv2D)  (None, 14, 14, 1024  262144      ['activation_
↪78[0][0]']
                               )

conv4_3_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_3_1x1_
↪increase[0][0]']
Normalization)                 )

add_25 (Add)                   (None, 14, 14, 1024  0           ['conv4_3_1x1_
↪increase/bn[0][0]',
                               )                                 'activation_76[0][0]']

activation_79 (Activation)     (None, 14, 14, 1024  0           ['add_25[0][0]
↪']
                               )

conv4_4_1x1_reduce (Conv2D)    (None, 14, 14, 256)  262144      ['activation_
↪79[0][0]']

conv4_4_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024       ['conv4_4_1x1_
↪reduce[0][0]']
rmalization)

activation_80 (Activation)     (None, 14, 14, 256)  0           ['conv4_4_1x1_
↪reduce/bn[0][0]']

conv4_4_3x3 (Conv2D)           (None, 14, 14, 256)  589824      ['activation_
↪80[0][0]']

conv4_4_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_4_
↪3x3[0][0]']
tion)

activation_81 (Activation)     (None, 14, 14, 256)  0           ['conv4_4_3x3/
↪bn[0][0]']

conv4_4_1x1_increase (Conv2D)  (None, 14, 14, 1024  262144      ['activation_
↪81[0][0]']
                               )

conv4_4_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_4_1x1_
↪increase[0][0]']
```

(continues on next page)

(continued from previous page)

```
Normalization)                  )

add_26 (Add)                    (None, 14, 14, 1024  0           ['conv4_4_1x1_
↪increase/bn[0][0]',
                )                                    'activation_79[0][0]']

activation_82 (Activation)      (None, 14, 14, 1024  0           ['add_26[0][0]
↪']
                )

conv4_5_1x1_reduce (Conv2D)     (None, 14, 14, 256)  262144      ['activation_
↪82[0][0]']

conv4_5_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024        ['conv4_5_1x1_
↪reduce[0][0]']
rmalization)

activation_83 (Activation)      (None, 14, 14, 256)  0           ['conv4_5_1x1_
↪reduce/bn[0][0]']

conv4_5_3x3 (Conv2D)            (None, 14, 14, 256)  589824      ['activation_
↪83[0][0]']

conv4_5_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024        ['conv4_5_
↪3x3[0][0]']
tion)

activation_84 (Activation)      (None, 14, 14, 256)  0           ['conv4_5_3x3/
↪bn[0][0]']

conv4_5_1x1_increase (Conv2D)   (None, 14, 14, 1024  262144      ['activation_
↪84[0][0]']
                )

conv4_5_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096        ['conv4_5_1x1_
↪increase[0][0]']
Normalization)                  )

add_27 (Add)                    (None, 14, 14, 1024  0           ['conv4_5_1x1_
↪increase/bn[0][0]',
                )                                    'activation_82[0][0]']

activation_85 (Activation)      (None, 14, 14, 1024  0           ['add_27[0][0]
↪']
                )

conv4_6_1x1_reduce (Conv2D)     (None, 14, 14, 256)  262144      ['activation_
↪85[0][0]']

conv4_6_1x1_reduce/bn (BatchNo  (None, 14, 14, 256)  1024        ['conv4_6_1x1_
↪reduce[0][0]']
rmalization)
```

(continues on next page)

(continued from previous page)

```
activation_86 (Activation)     (None, 14, 14, 256)  0           ['conv4_6_1x1_
↪reduce/bn[0][0]']

conv4_6_3x3 (Conv2D)           (None, 14, 14, 256)  589824      ['activation_
↪86[0][0]']

conv4_6_3x3/bn (BatchNormaliza  (None, 14, 14, 256)  1024       ['conv4_6_
↪3x3[0][0]']
tion)

activation_87 (Activation)     (None, 14, 14, 256)  0           ['conv4_6_3x3/
↪bn[0][0]']

conv4_6_1x1_increase (Conv2D)  (None, 14, 14, 1024  262144      ['activation_
↪87[0][0]']
                               )

conv4_6_1x1_increase/bn (Batch  (None, 14, 14, 1024  4096       ['conv4_6_1x1_
↪increase[0][0]']
Normalization)                 )

add_28 (Add)                   (None, 14, 14, 1024  0           ['conv4_6_1x1_
↪increase/bn[0][0]',
                               )                                 'activation_85[0][0]']

activation_88 (Activation)     (None, 14, 14, 1024  0           ['add_28[0][0]
↪']
                               )

conv5_1_1x1_reduce (Conv2D)    (None, 7, 7, 512)    524288      ['activation_
↪88[0][0]']

conv5_1_1x1_reduce/bn (BatchNo  (None, 7, 7, 512)   2048        ['conv5_1_1x1_
↪reduce[0][0]']
rmalization)

activation_89 (Activation)     (None, 7, 7, 512)    0           ['conv5_1_1x1_
↪reduce/bn[0][0]']

conv5_1_3x3 (Conv2D)           (None, 7, 7, 512)    2359296     ['activation_
↪89[0][0]']

conv5_1_3x3/bn (BatchNormaliza  (None, 7, 7, 512)   2048        ['conv5_1_
↪3x3[0][0]']
tion)

activation_90 (Activation)     (None, 7, 7, 512)    0           ['conv5_1_3x3/
↪bn[0][0]']

conv5_1_1x1_increase (Conv2D)  (None, 7, 7, 2048)   1048576     ['activation_
↪90[0][0]']
```

(continues on next page)

(continued from previous page)

```
conv5_1_1x1_proj (Conv2D)       (None, 7, 7, 2048)   2097152     ['activation_
↪88[0][0]']

conv5_1_1x1_increase/bn (Batch  (None, 7, 7, 2048)   8192        ['conv5_1_1x1_
↪increase[0][0]']
Normalization)

conv5_1_1x1_proj/bn (BatchNorm  (None, 7, 7, 2048)   8192        ['conv5_1_1x1_
↪proj[0][0]']
alization)

add_29 (Add)                    (None, 7, 7, 2048)   0           ['conv5_1_1x1_
↪increase/bn[0][0]',
                                 'conv5_1_1x1_proj/bn[0][0]']

activation_91 (Activation)      (None, 7, 7, 2048)   0           ['add_29[0][0]
↪']

conv5_2_1x1_reduce (Conv2D)     (None, 7, 7, 512)    1048576     ['activation_
↪91[0][0]']

conv5_2_1x1_reduce/bn (BatchNo  (None, 7, 7, 512)    2048        ['conv5_2_1x1_
↪reduce[0][0]']
rmalization)

activation_92 (Activation)      (None, 7, 7, 512)    0           ['conv5_2_1x1_
↪reduce/bn[0][0]']

conv5_2_3x3 (Conv2D)            (None, 7, 7, 512)    2359296     ['activation_
↪92[0][0]']

conv5_2_3x3/bn (BatchNormaliza  (None, 7, 7, 512)    2048        ['conv5_2_
↪3x3[0][0]']
tion)

activation_93 (Activation)      (None, 7, 7, 512)    0           ['conv5_2_3x3/
↪bn[0][0]']

conv5_2_1x1_increase (Conv2D)   (None, 7, 7, 2048)   1048576     ['activation_
↪93[0][0]']

conv5_2_1x1_increase/bn (Batch  (None, 7, 7, 2048)   8192        ['conv5_2_1x1_
↪increase[0][0]']
Normalization)

add_30 (Add)                    (None, 7, 7, 2048)   0           ['conv5_2_1x1_
↪increase/bn[0][0]',
                                 'activation_91[0][0]']

activation_94 (Activation)      (None, 7, 7, 2048)   0           ['add_30[0][0]
↪']
```

(continues on next page)

(continued from previous page)

```
413
414 conv5_3_1x1_reduce (Conv2D)    (None, 7, 7, 512)    1048576     ['activation_
    ↪94[0][0]']
415
416 conv5_3_1x1_reduce/bn (BatchNo  (None, 7, 7, 512)   2048        ['conv5_3_1x1_
    ↪reduce[0][0]']
417 rmalization)
418
419 activation_95 (Activation)     (None, 7, 7, 512)    0           ['conv5_3_1x1_
    ↪reduce/bn[0][0]']
420
421 conv5_3_3x3 (Conv2D)           (None, 7, 7, 512)    2359296     ['activation_
    ↪95[0][0]']
422
423 conv5_3_3x3/bn (BatchNormaliza  (None, 7, 7, 512)   2048        ['conv5_3_
    ↪3x3[0][0]']
424 tion)
425
426 activation_96 (Activation)     (None, 7, 7, 512)    0           ['conv5_3_3x3/
    ↪bn[0][0]']
427
428 conv5_3_1x1_increase (Conv2D)  (None, 7, 7, 2048)   1048576     ['activation_
    ↪96[0][0]']
429
430 conv5_3_1x1_increase/bn (Batch  (None, 7, 7, 2048)  8192        ['conv5_3_1x1_
    ↪increase[0][0]']
431 Normalization)
432
433 add_31 (Add)                   (None, 7, 7, 2048)   0           ['conv5_3_1x1_
    ↪increase/bn[0][0]',
434                                  'activation_94[0][0]']
435
436 activation_97 (Activation)     (None, 7, 7, 2048)   0           ['add_31[0][0]
    ↪']
437
438 avg_pool (AveragePooling2D)    (None, 1, 1, 2048)   0           ['activation_
    ↪97[0][0]']
439
440 global_average_pooling2d_1 (Gl  (None, 2048)        0           ['avg_
    ↪pool[0][0]']
441 obalAveragePooling2D)
442
443 gaussian_noise_1 (GaussianNois  (None, 2048)        0           ['global_
    ↪average_pooling2d_1[0][0
444 e)                                                           ]']
445
446 dense_x (Dense)                (None, 512)          1049088     ['gaussian_
    ↪noise_1[0][0]']
447
448 dropout_1 (Dropout)            (None, 512)          0           ['dense_x[0][0]
    ↪']
449
```

(continues on next page)

(continued from previous page)

```
 dense_1 (Dense)                 (None, 7)             3591        ['dropout_
↪1[0][0]']

==================================================================================================
Total params: 24,613,831
Trainable params: 24,560,711
Non-trainable params: 53,120
--------------------------------------------------------------------------------
↪------------------
--- Runtime: 2.222 sec. ---

True
```

Error – 1 –

In [2]:

```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')
video.load_video_model_deep_fe(
    show_summary = 1, out = True,
    runtime = True, run = True
)
```

[2]:

```
[2022-11-01 12:21:23] Invalid argument types or values in "Video.load_video_
↪model_deep_fe" ...

False
```

load_video_model_hc(*lang: str*, *show_summary: bool = False*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Formation of the neural network architecture of the model for obtaining scores by hand-crafted features

Parameters

- lang (*str*) – Language
- show_summary (*bool*) – Displaying the formed neural network architecture of the model
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the neural network architecture of the model is formed, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')
video.load_video_model_hc(
    show_summary = False, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-10-25 16:37:43] Formation of the neural network architecture of the model␣
↪for obtaining scores by hand-crafted features (video modality) ...

--- Runtime: 0.659 sec. ---

True
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')
video.load_video_model_hc(
    show_summary = 1, out = True,
    runtime = True, run = True
)
```

[2]:
```
[2022-10-26 12:27:41] Invalid argument types or values in "Video.load_video_
↪model_hc" ...

False
```

load_video_model_nn(*show_summary: bool = False*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Formation of a neural network architecture for obtaining scores by deep features

Parameters

- show_summary (*bool*) – Displaying the formed neural network architecture of the model
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the neural network architecture of the model is formed, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')
video.load_video_model_nn(
    show_summary = True, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-10-27 14:46:11] Formation of a neural network architecture for obtaining␣
↪scores by deep features (video modality) ...

Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 10, 512)]         0

 lstm (LSTM)                 (None, 1024)              6295552

 dropout (Dropout)           (None, 1024)              0

 dense (Dense)               (None, 5)                 5125

 activation (Activation)     (None, 5)                 0

=================================================================
Total params: 6,300,677
Trainable params: 6,300,677
Non-trainable params: 0
_________________________________________________________________
--- Runtime: 2.018 sec. ---

True
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')
video.load_video_model_nn(
    show_summary = 1, out = True,
    runtime = True, run = True
)
```

[2]:
```
[2022-10-27 14:47:22] Invalid argument types or values in "Video.load_video_
↪model_nn" ...

False
```

load_video_model_weights_deep_fe(*url: str*, *force_reload: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Downloading weights of a neural network model to obtain neural network features

Parameters

- url (*str*) – Full path to the file with weights of the neural network model
- force_reload (*bool*) – Forced download of a file with weights of a neural network model from the network
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the weights of the neural network model are downloaded, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.load_video_model_deep_fe(
    show_summary = False, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-11-01 12:41:59] Formation of neural network architecture for obtaining␣
↪deep features (video modality) ...

--- Runtime: 1.306 sec. ---

True
```

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

url = video.weights_for_big5_['video']['fe']['sberdisk']

video.load_video_model_weights_deep_fe(
    url = url,
    force_reload = True,
```

(continues on next page)

(continued from previous page)

```
    out = True,
    runtime = True,
    run = True
)
```

[2]:

```
[2022-11-01 12:42:51] Downloading weights of a neural network model to obtain deep features (video modality) ...

[2022-11-01 12:43:06] File download "weights_2022-11-01_12-27-07.h5" (100.0%) ...

--- Runtime: 14.781 sec. ---

True
```

Error – 1 –

In [3]:

```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

url = video.weights_for_big5_['video']['fe']['sberdisk']

video.load_video_model_weights_deep_fe(
    url = url,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[3]:

```
[2022-11-01 12:44:14] Downloading weights of a neural network model to obtain deep features (video modality) ...

[2022-11-01 12:44:28] File download "weights_2022-11-01_12-27-07.h5" (100.0%) ...

[2022-11-01 12:44:28] Ouch! Something went wrong ... the neural network architecture of the model for obtaining deep features is not formed (video modality) ...

--- Runtime: 13.926 sec. ---

False
```

load_video_model_weights_hc(*url: str*, *force_reload: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Downloading the weights of the neural network model to obtain scores by hand-crafted features

Parameters

- url (*str*) – Full path to the file with weights of the neural network model
- force˙reload (*bool*) – Forced download of a file with weights of a neural network model from the network
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the weights of the neural network model are downloaded, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.load_video_model_hc(
    show_summary = False, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-10-27 12:55:31] Formation of the neural network architecture of the model␣
↪for obtaining scores by hand-crafted features (video modality) ...

--- Runtime: 0.606 sec. ---

True
```

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

url = video.weights_for_big5_['video']['hc']['sberdisk']

video.load_video_model_weights_hc(
    url = url,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[2]:
```
[2022-10-27 13:08:04] Downloading the weights of the neural network model to␣
↪obtain scores by hand-crafted features (video modality) ...

[2022-10-27 13:08:05] File download "weights_2022-08-27_18-53-35.h5" (100.0%) ..
↪.

--- Runtime: 0.493 sec. ---

True
```

Error – 1 –

In [3]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

url = video.weights_for_big5_['video']['hc']['sberdisk']

video.load_video_model_weights_hc(
    url = url,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[3]:
```
[2022-10-27 13:09:54] Downloading the weights of the neural network model to␣
↪obtain scores by hand-crafted features (video modality) ...

[2022-10-27 13:09:54] File download "weights_2022-08-27_18-53-35.h5" (100.0%) ..
↪.

[2022-10-27 13:09:54] Ouch! Something went wrong ... the neural network␣
↪architecture of the model for obtaining scores by hand-crafted features has␣
↪not been formed (video modality) ...

--- Runtime: 0.424 sec. ---

False
```

load_video_model_weights_nn(*url: str*, *force_reload: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Downloading the weights of the neural network model to obtain scores for deep features

Parameters

- url (*str*) – Full path to the file with weights of the neural network model
- force_reload (*bool*) – Forced download of a file with weights of a neural network model from the network
- out (*bool*) – Display

- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns
True if the weights of the neural network model are downloaded, otherwise **False**

Return type
bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.load_video_model_nn(
    show_summary = False, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-10-27 15:17:13] Formation of a neural network architecture for obtaining␣
↪scores by deep features (video modality) ...

--- Runtime: 1.991 sec. ---

True
```

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

url = video.weights_for_big5_['video']['nn']['sberdisk']

video.load_video_model_weights_nn(
    url = url,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[2]:
```
[2022-10-27 15:19:08] Downloading the weights of the neural network model to␣
↪obtain scores for deep features (video modality) ...

[2022-10-27 15:19:11] File download "weights_2022-03-22_16-31-48.h5" (100.0%) ..
↪.
```

(continues on next page)

(continued from previous page)

```

--- Runtime: 3.423 sec. ---

True
```

Error – 1 –

In [3]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

url = video.weights_for_big5_['video']['nn']['sberdisk']

video.load_video_model_weights_nn(
    url = url,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[3]:
```
[2022-10-27 15:19:40] Downloading the weights of the neural network model to␣
↪obtain scores for deep features (video modality) ...

[2022-10-27 15:19:43] File download "weights_2022-03-22_16-31-48.h5" (100.0%) ..
↪.

[2022-10-27 15:19:43] Ouch! Something went wrong ... the neural network␣
↪architecture of the model for obtaining scores by deep features has not been␣
↪formed (video modality) ...

--- Runtime: 3.469 sec. ---

False
```

load_video_models_b5(*show_summary: bool = False*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Formation of neural network architectures of models for obtaining the personality traits scores

Parameters

- show_summary (*bool*) – Displaying the last generated neural network architecture of models
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the neural network architectures of the model are formed, otherwise **False**

Return type
bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')
video.load_video_models_b5(
    show_summary = True, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-11-04 15:29:26] Formation of neural network architectures of models for␣
↪obtaining the personality traits scores (video modality) ...

Model: "model_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32)]              0

 dense_1 (Dense)             (None, 1)                 33

 activ_1 (Activation)        (None, 1)                 0

=================================================================
Total params: 33
Trainable params: 33
Non-trainable params: 0
_________________________________________________________________
--- Runtime: 0.116 sec. ---

True
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')
video.load_video_models_b5(
    show_summary = 1, out = True,
    runtime = True, run = True
)
```

[2]:
```
[2022-11-04 15:30:15] Invalid argument types or values in "Video.load_video_
↪models_b5" ...

False
```

load_video_models_weights_b5(*url_openness: str*, *url_conscientiousness: str*, *url_extraversion: str*, *url_agreeableness: str*, *url_non_neuroticism: str*, *force_reload: bool = True*, *out: bool = True*, *runtime: bool = True*, *run: bool = True*) → bool

Downloading the weights of neural network models to obtain the personality traits scores

Parameters

- url_openness (*str*) – Full path to the file with the weights of the neural network model (openness)
- url_conscientiousness (*str*) – Full path to the file with the weights of the neural network model (conscientiousness)
- url_extraversion (*str*) – Full path to the file with the weights of the neural network model (extraversion)
- url_agreeableness (*str*) – Full path to the file with the weights of the neural network model (agreeableness)
- url_non_neuroticism (*str*) – Full path to the file with the weights of the neural network model (non-neuroticism)
- force_reload (*bool*) – Forced download of files with weights of neural network models from the network
- out (*bool*) – Display
- runtime (*bool*) – Runtime count
- run (*bool*) – Run blocking

Returns

True if the weights of the neural network models are downloaded, otherwise **False**

Return type

bool

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.load_video_models_b5(
    show_summary = False, out = True,
    runtime = True, run = True
)
```

[1]:
```
[2022-11-04 18:56:41] Formation of neural network architectures of models for␣
↪obtaining the personality traits scores (video modality) ...

--- Runtime: 0.117 sec. ---

True
```

In [2]:

```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

url_openness = video.weights_for_big5_['video']['b5']['openness']['sberdisk']
url_conscientiousness = video.weights_for_big5_['video']['b5']['conscientiousness']['sberdisk']
url_extraversion = video.weights_for_big5_['video']['b5']['extraversion']['sberdisk']
url_agreeableness = video.weights_for_big5_['video']['b5']['agreeableness']['sberdisk']
url_non_neuroticism = video.weights_for_big5_['video']['b5']['non_neuroticism']['sberdisk']

video.load_video_models_weights_b5(
    url_openness = url_openness,
    url_conscientiousness = url_conscientiousness,
    url_extraversion = url_extraversion,
    url_agreeableness = url_agreeableness,
    url_non_neuroticism = url_non_neuroticism,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[2]:

```
[2022-11-04 18:58:59] Downloading the weights of neural network models to obtain the personality traits scores (video modality) ...

[2022-11-04 18:59:00] File download "weights_2022-06-15_16-46-30.h5" (100.0%) .. Openness

[2022-11-04 18:59:00] File download "weights_2022-06-15_16-48-50.h5" (100.0%) .. Conscientiousness

[2022-11-04 18:59:00] File download "weights_2022-06-15_16-54-06.h5" (100.0%) .. Extraversion

[2022-11-04 18:59:01] File download "weights_2022-06-15_17-02-03.h5" (100.0%) .. Agreeableness

[2022-11-04 18:59:01] File download "weights_2022-06-15_17-06-15.h5" (100.0%) .. Non-Neuroticism

--- Runtime: 1.827 sec. ---

True
```

Error – 1 –

In [3]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.path_to_save_ = './models'
video.chunk_size_ = 2000000

url_openness = video.weights_for_big5_['video']['b5']['openness']['sberdisk']
url_conscientiousness = video.weights_for_big5_['video']['b5']['conscientiousness']['sberdisk']
url_extraversion = video.weights_for_big5_['video']['b5']['extraversion']['sberdisk']
url_agreeableness = video.weights_for_big5_['video']['b5']['agreeableness']['sberdisk']
url_non_neuroticism = video.weights_for_big5_['video']['b5']['non_neuroticism']['sberdisk']

video.load_video_models_weights_b5(
    url_openness = url_openness,
    url_conscientiousness = url_conscientiousness,
    url_extraversion = url_extraversion,
    url_agreeableness = url_agreeableness,
    url_non_neuroticism = url_non_neuroticism,
    force_reload = True,
    out = True,
    runtime = True,
    run = True
)
```

[3]:
```
[2022-11-04 19:02:32] Downloading the weights of neural network models to obtain the personality traits scores (video modality) ...

[2022-11-04 19:02:32] File download "weights_2022-06-15_16-46-30.h5" (100.0%) ..

[2022-11-04 19:02:32] Something went wrong ... ailed to load neural network model weights ... Openness

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/video.py
    Line: 2833
    Method: load_models_weights_b5
    Error type: AttributeError

[2022-11-04 19:02:32] File download "weights_2022-06-15_16-48-50.h5" (100.0%) ..

[2022-11-04 19:02:32] Something went wrong ... ailed to load neural network model weights ... Conscientiousness

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/video.py
    Line: 2833
    Method: load_models_weights_b5
```

(continues on next page)

(continued from previous page)

```
    Error type: AttributeError

[2022-11-04 19:02:33] File download "weights_2022-06-15_16-54-06.h5" (100.0%) ...

[2022-11-04 19:02:33] Something went wrong ... ailed to load neural network model weights ... Extraversion

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/video.py
    Line: 2833
    Method: load_models_weights_b5
    Error type: AttributeError

[2022-11-04 19:02:33] File download "weights_2022-06-15_17-02-03.h5" (100.0%) ...

[2022-11-04 19:02:33] Something went wrong ... ailed to load neural network model weights ... Agreeableness

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/video.py
    Line: 2833
    Method: load_models_weights_b5
    Error type: AttributeError

[2022-11-04 19:02:34] File download "weights_2022-06-15_17-06-15.h5" (100.0%) ...

[2022-11-04 19:02:34] Something went wrong ... ailed to load neural network model weights ... Non-Neuroticism

    File: /Users/dl/GitHub/oceanai/oceanai/modules/lab/video.py
    Line: 2833
    Method: load_models_weights_b5
    Error type: AttributeError

--- Runtime: 1.831 sec. ---

False
```

property video˙model˙deep˙fe˙: Optional[Model]

Obtaining a neural network model **tf.keras.Model** to obtain deep features

Returns

Neural network model **tf.keras.Model** or None

Return type

Optional[tf.keras.Model]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.load_video_model_deep_fe(
    show_summary = False, out = True,
    runtime = True, run = True
)

video.video_model_deep_fe_
```

[1]:
```
[2022-11-01 12:12:35] Formation of neural network architecture for obtaining␣
↪deep features (video modality) ...

--- Runtime: 1.468 sec. ---

<keras.engine.functional.Functional at 0x14e138100>
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.video_model_deep_fe_
```

[2]:
```

```

property video˙model˙hc˙: Optional[Model]

Obtaining a neural network model **tf.keras.Model** to obtain scores by hand-crafted features

Returns
Neural network model **tf.keras.Model** or None

Return type
Optional[tf.keras.Model]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.load_video_model_hc(
    show_summary = False, out = True,
    runtime = True, run = True
```

(continues on next page)

(continued from previous page)

```
)

video.video_model_hc_
```

[1]:
```
[2022-10-26 12:37:42] Formation of the neural network architecture of the model␣
↪for obtaining scores by hand-crafted features (video modality) ...

--- Runtime: 1.112 sec. ---

<keras.engine.functional.Functional at 0x1434eb1f0>
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.video_model_hc_
```

[2]:
```

```

property video˙model˙nn˙: Optional[Model]

Obtaining a neural network model **tf.keras.Model** to obtain scores for deep features

Returns

Neural network model **tf.keras.Model** or None

Return type

Optional[tf.keras.Model]

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.load_video_model_nn(
    show_summary = False, out = True,
    runtime = True, run = True
)

video.video_model_nn_
```

[1]:
```
[2022-10-27 14:49:00] Formation of a neural network architecture for obtaining␣
↪scores by deep features (video modality) ...

--- Runtime: 1.986 sec. ---

<keras.engine.functional.Functional at 0x13d5295b0>
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')

video.video_model_nn_
```

[2]:
```

```

property video˙models˙b5˙: Dict[str, Optional[Model]]

Obtaining neural network models **tf.keras.Model** to obtain the personality traits scores

Returns
Dictionary with neural network models **tf.keras.Model**

Return type
Dict

Examples

True – 1 –

In [1]:
```
from oceanai.modules.lab.video import Video

video = Video()

video.load_video_models_b5(
    show_summary = False, out = True,
    runtime = True, run = True
)

video.video_models_b5_
```

[1]:
```
[2022-10-19 15:45:35] Formation of neural network architectures of models for␣
↪obtaining the personality traits scores ...

--- Runtime: 0.07 sec. ---

{
    'openness': <keras.engine.functional.Functional at 0x1481e03a0>,
    'conscientiousness': <keras.engine.functional.Functional at 0x147d13520>,
    'extraversion': <keras.engine.functional.Functional at 0x1481edfa0>,
    'agreeableness': <keras.engine.functional.Functional at 0x1481cfc40>,
    'non_neuroticism': <keras.engine.functional.Functional at 0x1481cffd0>
}
```

Error – 1 –

In [2]:
```
from oceanai.modules.lab.video import Video

video = Video(lang = 'en')
```

(continues on next page)

(continued from previous page)

```
4
5 video.video_models_b5_
```

[2]:

```
1 {
2     'openness': None,
3     'conscientiousness': None,
4     'extraversion': None,
5     'agreeableness': None,
6     'non_neuroticism': None
7 }
```

Text

Multimodal information fusion

Build

Custom exceptions
: Documentation for `modules/core/exceptions.py` file

Language detection
: Documentation for `modules/core/language.py` file

Messages
: Documentation for `modules/core/messages.py` file

Settings
: Documentation for `modules/core/settings.py` file

Core
: Documentation for `modules/core/core.py` file

Archive processing
: Documentation for `modules/lab/unzip.py` file

Downloading files
: Documentation for `modules/lab/download.py` file

Audio
: Documentation for `modules/lab/audio.py` file

Video
: Documentation for `modules/lab/video.py` file

Text
: Documentation for `modules/lab/text.py` file

Multimodal information fusion
: Documentation for `modules/lab/prediction.py` file

Build
: Documentation for `modules/lab/build.py` file

4.2.3 Class Diagram

Рис.1: Exceptions

Рис.2: Main Classes

4.2.4 Development Team

The `OCEAN-AI` library is developed and maintained by a research group from the Speech and Multimodal Interfaces laboratory (SMIL) St. Petersburg Federal Research Center of the Russian Academy of Sciences (St. Petersburg FRC RAS), which is part of the Research Center Strong Artificial Intelligence in industry (ITMO University) Namely:

- Ryumina Elena - PhD student of ITMO University, junior researcher at SMIL, SPC RAS.
- Ryumin Dmitry - PhD in Engineering, senior researcher at SMIL, SPC RAS.
- Karpov Aleksey - Doctor of Technical Sciences (2013), Professor, specialty 05.13.11 (2022), Chief Researcher (Head), SMIL SPC RAS.

4.2.5 FAQ

This is a list of FAQ about *OCEAN-AI* and the answers to them.

What is OCEAN-AI**?**

An open-source library consisting of a set of algorithms for intellectual analysis of human behavior based on multimodal data for automatic personal traits assessment. The library evaluates five traits: Openness to experience, Conscientiousness, Extraversion, Agreeableness, Non-Neuroticism.

Why it is named OCEAN-AI**?**

OCEAN is an abbreviation for five personality traits (Openness, Conscientiousness, Extraversion, Agreeableness, Non-Neuroticism), and AI is Artificial Intelligence abbreviation.

Can I install OCEAN-AI **using pip?**

Yes, follow the link.

FIVE

INDEXING

- genindex
- modindex
- search

PYTHON MODULE INDEX

O
oceanai.modules.core.exceptions, 167

INDEX

Symbols

__calc_reshape_img_coef() (*oceanai.modules.lab.video.Video method*), 278
__concat_pred() (*oceanai.modules.lab.audio.Audio method*), 251
__concat_pred() (*oceanai.modules.lab.video.Video method*), 280
__get_languages() (*oceanai.modules.core.language.Language method*), 168
__get_locales() (*oceanai.modules.core.language.Language method*), 169
__is_notebook() (*oceanai.modules.core.core.Core method*), 199
__load_audio_model_b5() (*oceanai.modules.lab.audio.Audio method*), 254
__load_model_weights() (*oceanai.modules.lab.audio.Audio method*), 255
__load_model_weights() (*oceanai.modules.lab.video.Video method*), 282
__load_video_model_b5() (*oceanai.modules.lab.video.Video method*), 284
__norm_pred() (*oceanai.modules.lab.audio.Audio method*), 257
__norm_pred() (*oceanai.modules.lab.video.Video method*), 285
__progressbar_download_file_from_url() (*oceanai.modules.lab.download.Download method*), 245
__progressbar_unzip() (*oceanai.modules.lab.unzip.Unzip method*), 243
__set_locale() (*oceanai.modules.core.language.Language method*), 169
__smile() (*oceanai.modules.lab.audio.Audio method*), 258
_add_last_el_notebook_history_output() (*oceanai.modules.core.core.Core method*), 199
_add_notebook_history_output() (*oceanai.modules.core.core.Core method*), 200
_append_to_list_of_accuracy() (*oceanai.modules.core.core.Core method*), 201
_append_to_list_of_files() (*oceanai.modules.core.core.Core method*), 203
_bold_wrapper() (*oceanai.modules.core.core.Core method*), 205
_candidate_ranking() (*oceanai.modules.core.core.Core method*), 206
_clear_notebook_history_output() (*oceanai.modules.core.core.Core method*), 206
_colleague_personality_desorders() (*oceanai.modules.core.core.Core method*), 207
_colleague_personality_type_match() (*oceanai.modules.core.core.Core method*), 207
_colleague_ranking() (*oceanai.modules.core.core.Core method*), 208
_compatibility_percentage() (*oceanai.modules.core.core.Core method*), 208
_create_folder_for_logs() (*oceanai.modules.core.core.Core method*), 209
_del_last_el_notebook_history_output() (*oceanai.modules.core.core.Core method*), 209
_download_file_from_url() (*oceanai.modules.lab.download.Download method*), 247
_error() (*oceanai.modules.core.core.Core method*), 210
_error_wrapper() (*oceanai.modules.core.core.Core*

method), 211
_get_acoustic_features() (*oceanai.modules.lab.audio.Audio method*), 259
_get_paths() (*oceanai.modules.core.core.Core method*), 212
_get_visual_features() (*oceanai.modules.lab.video.Video method*), 286
_info() (*oceanai.modules.core.core.Core method*), 213
_info_true() (*oceanai.modules.core.core.Core method*), 214
_info_wrapper() (*oceanai.modules.core.core.Core method*), 215
_inv_args() (*oceanai.modules.core.core.Core method*), 216
_metadata_info() (*oceanai.modules.core.core.Core method*), 217
_notebook_display_markdown() (*oceanai.modules.core.core.Core method*), 218
_other_error() (*oceanai.modules.core.core.Core method*), 219
_priority_calculation() (*oceanai.modules.core.core.Core method*), 220
_priority_skill_calculation() (*oceanai.modules.core.core.Core method*), 221
_professional_match() (*oceanai.modules.core.core.Core method*), 221
_progressbar() (*oceanai.modules.core.core.Core method*), 221
_progressbar_union_predictions() (*oceanai.modules.core.core.Core method*), 223
_r_end() (*oceanai.modules.core.core.Core method*), 225
_r_start() (*oceanai.modules.core.core.Core method*), 226
_round_math() (*oceanai.modules.core.core.Core method*), 227
_save_logs() (*oceanai.modules.core.core.Core method*), 228
_search_file() (*oceanai.modules.core.core.Core method*), 229
_stat_acoustic_features() (*oceanai.modules.core.core.Core method*), 229
_stat_text_features() (*oceanai.modules.core.core.Core method*), 230
_stat_visual_features() (*oceanai.modules.core.core.Core method*), 231
_traceback() (*oceanai.modules.core.core.Core static method*), 232
_unzip() (*oceanai.modules.lab.unzip.Unzip method*), 243

A

Audio (*class in oceanai.modules.lab.audio*), 251
audio_model_hc_ (*oceanai.modules.lab.audio.Audio property*), 261
audio_model_nn_ (*oceanai.modules.lab.audio.Audio property*), 262
audio_models_b5_ (*oceanai.modules.lab.audio.Audio property*), 263
AudioMessages (*class in oceanai.modules.lab.audio*), 251

B

bold_text (*oceanai.modules.core.settings.Settings attribute*), 174
bold_text_ (*oceanai.modules.core.settings.Settings property*), 175

C

chunk_size_ (*oceanai.modules.core.settings.Settings property*), 176
color_err (*oceanai.modules.core.settings.Settings attribute*), 177
color_err_ (*oceanai.modules.core.settings.Settings property*), 178
color_info (*oceanai.modules.core.settings.Settings attribute*), 179
color_info_ (*oceanai.modules.core.settings.Settings property*), 180
color_simple (*oceanai.modules.core.settings.Settings attribute*), 182
color_simple_ (*oceanai.modules.core.settings.Settings property*), 183
color_true (*oceanai.modules.core.settings.Settings attribute*), 184
color_true_ (*oceanai.modules.core.settings.Settings property*), 185
Core (*class in oceanai.modules.core.core*), 198
CoreMessages (*class in oceanai.modules.core.core*), 198
CustomException, 167

D

df_accuracy_ (*oceanai.modules.core.core.Core property*), 233
df_files_ (*oceanai.modules.core.core.Core property*), 233

df_files_colleague_ (*oceanai.modules.core.core.Core property*), 234
df_files_MBTI_colleague_match_ (*oceanai.modules.core.core.Core property*), 233
df_files_MBTI_disorders_ (*oceanai.modules.core.core.Core property*), 234
df_files_MBTI_job_match_ (*oceanai.modules.core.core.Core property*), 234
df_files_priority_ (*oceanai.modules.core.core.Core property*), 234
df_files_priority_skill_ (*oceanai.modules.core.core.Core property*), 234
df_files_ranking_ (*oceanai.modules.core.core.Core property*), 235
df_pkgs_ (*oceanai.modules.core.core.Core property*), 235
dict_of_accuracy_ (*oceanai.modules.core.core.Core property*), 236
dict_of_files_ (*oceanai.modules.core.core.Core property*), 236
Download (*class in oceanai.modules.lab.download*), 245
download_file_from_url() (*oceanai.modules.lab.download.Download method*), 250
DownloadMessages (*class in oceanai.modules.lab.download*), 244

E

ext_ (*oceanai.modules.core.settings.Settings property*), 186

G

get_acoustic_features() (*oceanai.modules.lab.audio.Audio method*), 264
get_audio_union_predictions() (*oceanai.modules.lab.audio.Audio method*), 264
get_video_union_predictions() (*oceanai.modules.lab.video.Video method*), 289
get_visual_features() (*oceanai.modules.lab.video.Video method*), 289

I

ignore_dirs_ (*oceanai.modules.core.settings.Settings property*), 187
InvalidContentLength, 167
is_notebook_ (*oceanai.modules.core.core.Core property*), 237
IsSmallWindowSizeError, 168

K

keys_dataset_ (*oceanai.modules.core.settings.Settings property*), 188

L

lang (*oceanai.modules.core.language.Language attribute*), 170
lang_ (*oceanai.modules.core.language.Language property*), 171
Language (*class in oceanai.modules.core.language*), 168
libs_vers() (*oceanai.modules.core.core.Core method*), 237
load_audio_model_hc() (*oceanai.modules.lab.audio.Audio method*), 265
load_audio_model_nn() (*oceanai.modules.lab.audio.Audio method*), 266
load_audio_model_weights_hc() (*oceanai.modules.lab.audio.Audio method*), 268
load_audio_model_weights_nn() (*oceanai.modules.lab.audio.Audio method*), 270
load_audio_models_b5() (*oceanai.modules.lab.audio.Audio method*), 272
load_audio_models_weights_b5() (*oceanai.modules.lab.audio.Audio method*), 273
load_video_model_deep_fe() (*oceanai.modules.lab.video.Video method*), 290
load_video_model_hc() (*oceanai.modules.lab.video.Video method*), 303
load_video_model_nn() (*oceanai.modules.lab.video.Video method*), 304
load_video_model_weights_deep_fe() (*oceanai.modules.lab.video.Video method*), 305
load_video_model_weights_hc() (*oceanai.modules.lab.video.Video method*), 307
load_video_model_weights_nn() (*oceanai.modules.lab.video.Video method*), 309

load_video_models_b5() (*oceanai.modules.lab.video.Video method*), 311
load_video_models_weights_b5() (*oceanai.modules.lab.video.Video method*), 312
locales_ (*oceanai.modules.core.language.Language property*), 172

M

Messages (*class in oceanai.modules.core.messages*), 173
module
    oceanai.modules.core.exceptions, 167

N

num_to_df_display (*oceanai.modules.core.settings.Settings attribute*), 190
num_to_df_display_ (*oceanai.modules.core.settings.Settings property*), 191

O

oceanai.modules.core.exceptions
    module, 167

P

path_to_dataset_ (*oceanai.modules.core.settings.Settings property*), 192
path_to_locales_ (*oceanai.modules.core.language.Language property*), 173
path_to_logs_ (*oceanai.modules.core.settings.Settings property*), 193
path_to_save_ (*oceanai.modules.core.settings.Settings property*), 195
path_to_unzip (*oceanai.modules.lab.unzip.Unzip property*), 244

R

runtime_ (*oceanai.modules.core.core.Core property*), 238

S

Settings (*class in oceanai.modules.core.settings*), 173
show_notebook_history_output() (*oceanai.modules.core.core.Core method*), 239
smile_ (*oceanai.modules.lab.audio.Audio property*), 277

T

text_runtime (*oceanai.modules.core.settings.Settings attribute*), 196
text_runtime_ (*oceanai.modules.core.settings.Settings property*), 197
true_traits_ (*oceanai.modules.core.core.Core property*), 240

U

Unzip (*class in oceanai.modules.lab.unzip*), 242
unzip() (*oceanai.modules.lab.unzip.Unzip method*), 244
UnzipMessages (*class in oceanai.modules.lab.unzip*), 242

V

Video (*class in oceanai.modules.lab.video*), 278
video_model_deep_fe_ (*oceanai.modules.lab.video.Video property*), 316
video_model_hc_ (*oceanai.modules.lab.video.Video property*), 317
video_model_nn_ (*oceanai.modules.lab.video.Video property*), 318
video_models_b5_ (*oceanai.modules.lab.video.Video property*), 319
VideoMessages (*class in oceanai.modules.lab.video*), 278

W

weights_for_big5_ (*oceanai.modules.core.core.Core property*), 240