scandeval

Dan Saattrup Nielsen

Apr 19, 2022

CONTENTS:

CHAPTER

ONE

SCANDEVAL PACKAGE

1.1 Subpackages

1.1.1 scandeval.benchmarks package

Subpackages

scandeval.benchmarks.abstract package

Submodules

scandeval.benchmarks.abstract.base module

Abstract base class for evaluating models

class scandeval.benchmarks.abstract.base.**BaseBenchmark**(*name: str*, *task: str*, *metric_names: Dict[str, str]*, *id2label: Optional[List[str]] = None*, *label_synonyms: Optional[List[List[str]]] = None*, *evaluate_train: bool = False*, *cache_dir: str = '.benchmark_models'*, *two_labels: bool = False*, *split_point: Optional[int] = None*, *verbose: bool = False*)

Bases: abc.ABC

Abstract base class for finetuning and evaluating models.

Parameters

- **name** (*str*) – The name of the dataset.
- **task** (*str*) – The type of task to be benchmarked.
- **metric_names** (*dict*) – A dictionary with the variable names of the metrics used in the dataset as keys, and a more human readable name of them as values.
- **id2label** (*list or None*) – A list of all the labels, which is used to convert indices to their labels. This will only be used if the pretrained model does not already have one. Defaults to None.
- **label_synonyms** (*list of lists of str*) – A list of synonyms for each label. Every entry in *label_synonyms* is a list of synonyms, where one of the synonyms is contained in *id2label*. If None then no synonyms will be used. Defaults to None.

- **evaluate_train** (*bool*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **cache_dir** (*str*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **two_labels** (*bool*) – Whether two labels should be predicted in the dataset. If this is True then *split_point* has to be set. Defaults to False.
- **split_point** (*int or None*) – When there are two labels to be predicted, this is the index such that *id2label[:split_point]* contains the labels for the first label, and *id2label[split_point]* contains the labels for the second label. Only relevant if *two_labels* is True. Defaults to None.
- **verbose** (*bool*) – Whether to print additional output during evaluation. Defaults to False.
- **name** – The name of the dataset.
- **task** – The type of task to be benchmarked.
- **metric_names** – The names of the metrics.
- **id2label** – A list converting indices to labels.
- **label2id** (*dict or None*) – A dictionary converting labels to indices.
- **num_labels** (*int or None*) – The number of labels in the dataset.
- **label_synonyms** – Synonyms of the dataset labels.
- **evaluate_train** – Whether the training set should be evaluated.
- **cache_dir** – Directory where models are cached.
- **two_labels** – Whether two labels should be predicted.
- **split_point** – Splitting point of *id2label* into labels.
- **verbose** – Whether to print additional output.

benchmark(*model_id: str*, *progress_bar: bool = True*) → Dict[str, dict]

Benchmark a model.

Parameters

- **model_id** (*str*) – The full HuggingFace Hub path to the pretrained transformer model. The specific model version to use can be added after the suffix '@': "model_id@v1.0.0". It can be a branch name, a tag name, or a commit id (currently only supported for HuggingFace models, and it defaults to 'main' for latest).
- **progress_bar** (*bool, optional*) – Whether to show a progress bar or not. Defaults to True.

Returns The keys in the dict are 'raw_metrics' and 'total', with all the raw metrics in the first dictionary and the aggregated metrics in the second.

Return type dict

Raises **RuntimeError** – If the extracted framework is not recognized.

scandeval.benchmarks.abstract.dep module

Abstract dependency parsing benchmark

class scandeval.benchmarks.abstract.dep.**DepBenchmark**(*name: str*, *cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: scandeval.benchmarks.abstract.token_classification.TokenClassificationBenchmark

Abstract dependency parsing benchmark.

Parameters

- **name** (*str*) – The name of the dataset.
- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.abstract.ner module

Abstract NER tagging benchmark

class scandeval.benchmarks.abstract.ner.**NerBenchmark**(*name: str*, *cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.token_classification.TokenClassificationBenchmark*

Abstract NER tagging benchmark.

Parameters

- **name** (*str*) – The name of the dataset.
- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.abstract.pos module

Abstract POS tagging benchmark

class scandeval.benchmarks.abstract.pos.**PosBenchmark**(*name: str*, *cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.token_classification.TokenClassificationBenchmark*

Abstract NER tagging benchmark.

Parameters

- **name** (*str*) – The name of the dataset.
- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

`task`
The type of task to be benchmarked.

Type str

`metric_names`
The names of the metrics.

Type dict

`id2label`
A dictionary converting indices to labels.

Type dict or None

`label2id`
A dictionary converting labels to indices.

Type dict or None

`num_labels`
The number of labels in the dataset.

Type int or None

`label_synonyms`
Synonyms of the dataset labels.

Type list of lists of str

`evaluate_train`
Whether the training set should be evaluated.

Type bool

`cache_dir`
Directory where models are cached.

Type str

`two_labels`
Whether two labels should be predicted.

Type bool

`split_point`
Splitting point of *id2label* into labels.

Type int or None

`verbose`
Whether to print additional output.

Type bool

scandeval.benchmarks.abstract.text_classification module

Abstract text classification benchmark

class scandeval.benchmarks.abstract.text_classification.**TextClassificationBenchmark**(*name: str*, *id2label: list*, *label_synonyms: Optional[List[List[str]]] = None*, *evaluate_train: bool = False*, *cache_dir: str = '.benchmark_models'*, *two_labels: bool = False*, *split_point: Optional[int] = None*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.base.BaseBenchmark*, abc.ABC

Abstract text classification benchmark.

Parameters

- **name** (*str*) – The name of the dataset.
- **metric_names** (*dict*) – A dictionary with the variable names of the metrics used in the dataset as keys, and a more human readable name of them as values.
- **id2label** (*list or None, optional*) – A list of all the labels, which is used to convert indices to their labels. This will only be used if the pretrained model does not already have one. Defaults to None.
- **label_synonyms** (*list of lists of str or None, optional*) – A list of synonyms for each label. Every entry in *label_synonyms* is a list of synonyms, where one of the synonyms is contained in *id2label*. If None then no synonyms will be used. Defaults to None.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.

- **two_labels** (*bool, optional*) – Whether two labels should be predicted in the dataset. If this is True then *split_point* has to be set. Defaults to False.
- **split_point** (*int or None, optional*) – When there are two labels to be predicted, this is the index such that *id2label[:split_point]* contains the labels for the first label, and *id2label[split_point]* contains the labels for the second label. Only relevant if *two_labels* is True. Defaults to None.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

create_numerical_labels(*examples: dict*, *label2id: dict*) → dict

scandeval.benchmarks.abstract.token_classification module

Abstract token classification benchmark

class scandeval.benchmarks.abstract.token_classification.**TokenClassificationBenchmark**(*name: str*, *metric_names: Dict[str, str]*, *id2label: list*, *label_synonyms: Optional[List[List[str]]] = None*, *evaluate_train: bool = False*, *cache_dir: str = '.benchmark_models'*, *two_labels: bool = False*, *split_point: Optional[int] = None*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.base.BaseBenchmark*, abc.ABC

Abstract token classification benchmark.

Parameters

- **name** (*str*) – The name of the dataset.
- **metric_names** (*dict*) – A dictionary with the variable names of the metrics used in the dataset as keys, and a more human readable name of them as values.
- **id2label** (*list or None, optional*) – A list of all the labels, which is used to convert indices to their labels. This will only be used if the pretrained model does not already have one. Defaults to None.
- **label_synonyms** (*list of lists of str or None, optional*) – A list of synonyms for each label. Every entry in *label_synonyms* is a list of synonyms, where one of the synonyms is contained in *id2label*. If None then no synonyms will be used. Defaults to None.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **two_labels** (*bool, optional*) – Whether two labels should be predicted in the dataset. If this is True then *split_point* has to be set. Defaults to False.
- **split_point** (*int or None, optional*) – When there are two labels to be predicted, this is the index such that *id2label[:split_point]* contains the labels for the first label, and *id2label[split_point]* contains the labels for the second label. Only relevant if *two_labels* is True. Defaults to None.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

Module contents

Submodules

scandeval.benchmarks.absabank_imm module

Immigration sentiment classification on the ABSAbank-Imm dataset

class scandeval.benchmarks.absabank_imm.**AbsabankImmBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the ABSAbank-Imm dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.angry_tweets module

Sentiment evaluation of a language model on the AngryTweets dataset

class scandeval.benchmarks.angry_tweets.**AngryTweetsBenchmark**(*cache_dir: str = '.benchmark_models', evaluate_train: bool = False, verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the AngryTweets dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.dalaj module

Correct spelling classification of a language model on the DaLaJ dataset

class scandeval.benchmarks.dalaj.**DalajBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the DaLaJ dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.dane module

NER evaluation of a language model on the DaNE dataset

class scandeval.benchmarks.dane.**DaneBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.ner.NerBenchmark*

Benchmark of language models on the DaNE dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.ddt_dep module

Dependency parsing evaluation of a language model on the DDT dataset

class scandeval.benchmarks.ddt_dep.**DdtDepBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.dep.DepBenchmark*

Benchmark of language models on the dependency parsing part of the DDT.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.ddt_pos module

POS evaluation of a language model on the DDT dataset

class scandeval.benchmarks.ddt_pos.**DdtPosBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.pos.PosBenchmark*

Benchmark of language models on the POS part of the DDT dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.

- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.dkhate module

Hate speech classification of a language model on the DKHate dataset

class scandeval.benchmarks.dkhate.**DkHateBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the DKHate dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.europarl module

Sentiment evaluation of a language model on the Europarl dataset

class scandeval.benchmarks.europarl.**EuroparlBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the Europarl dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.fdt_dep module

Dependency parsing evaluation of a language model on the FDT dataset

class scandeval.benchmarks.fdt_dep.**FdtDepBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.dep.DepBenchmark*

Benchmark of language models on the dependency parsing part of the FDT.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.fdt_pos module

POS evaluation of a language model on the FDT dataset

class scandeval.benchmarks.fdt_pos.**FdtPosBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.pos.PosBenchmark*

Benchmark of language models on the POS part of the FDT dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.

- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.idt_dep module

Dependency parsing evaluation of a language model on the IDT dataset

class scandeval.benchmarks.idt_dep.**IdtDepBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.dep.DepBenchmark*

Benchmark of language models on the dependency parsing part of the IDT.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.idt_pos module

POS evaluation of a language model on the IDT dataset

class scandeval.benchmarks.idt_pos.**IdtPosBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.pos.PosBenchmark*

Benchmark of language models on the POS part of the IDT dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.lcc module

Sentiment evaluation of a language model on the LCC dataset

class scandeval.benchmarks.lcc.**LccBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the LCC dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.mim_gold_ner module

NER evaluation of a language model on the MIM-GOLD-NER dataset

class scandeval.benchmarks.mim_gold_ner.**MimGoldNerBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.ner.NerBenchmark*

Benchmark of language models on the MIM-GOLD-NER dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.

- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.ndt_nb_dep module

Dependency parsing evaluation of a language model on the NDT-NB dataset

class scandeval.benchmarks.ndt_nb_dep.**NdtNBDepBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.dep.DepBenchmark*

Benchmark of language models on the dependency parsing part of DDT-NB.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.ndt_nb_pos module

POS evaluation of a language model on the Bokmål part of the NDT dataset

class scandeval.benchmarks.ndt_nb_pos.**NdtNBPosBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.pos.PosBenchmark*

Benchmark of language models on the Bokmål POS part of the NDT dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.ndt_nn_dep module

Dependency parsing evaluation of a language model on the NDT-NN dataset

class scandeval.benchmarks.ndt_nn_dep.**NdtNNDepBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.dep.DepBenchmark*

Benchmark of language models on the dependency parsing part of DDT-NN.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.ndt_nn_pos module

POS evaluation of a language model on the Nynorsk part of the NDT dataset

class scandeval.benchmarks.ndt_nn_pos.**NdtNNPosBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.pos.PosBenchmark*

Benchmark of language models on the Nynorsk POS part of the NDT dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.

- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.nordial module

Dialect classification of a language model on the NorDial dataset

class scandeval.benchmarks.nordial.**NorDialBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the NorDial dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.norec module

Sentiment evaluation of a language model on the NoReC dataset

class scandeval.benchmarks.norec.**NorecBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the NoReC dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.norec_fo module

Sentiment evaluation of a language model on the NoReC-FO dataset

class scandeval.benchmarks.norec_fo.**NorecFOBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the NoReC-FO dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.norec_is module

Sentiment evaluation of a language model on the NoReC-IS dataset

class scandeval.benchmarks.norec_is.**NorecISBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the NoReC-IS dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.

- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.norne_nb module

NER evaluation of a language model on the Bokmål part of NorNE

class scandeval.benchmarks.norne_nb.**NorneNBBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.ner.NerBenchmark*

Benchmark of language models on the Bokmål part of the NorNE dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.norne_nn module

NER evaluation of a language model on the Nynorsk part of NorNE

class scandeval.benchmarks.norne_nn.**NorneNNBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.ner.NerBenchmark*

Benchmark of language models on the Nynorsk part of the NorNE dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.sdt_dep module

Dependency parsing evaluation of a language model on the SDT dataset

class scandeval.benchmarks.sdt_dep.**SdtDepBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.dep.DepBenchmark*

Benchmark of language models on the dependency parsing part of the SDT.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.sdt_pos module

POS evaluation of a language model on the SDT dataset

class scandeval.benchmarks.sdt_pos.**SdtPosBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.pos.PosBenchmark*

Benchmark of language models on the POS part of the SDT dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.

- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.suc3 module

NER evaluation of a language model on the SUC 3.0 dataset

class scandeval.benchmarks.suc3.**Suc3Benchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.ner.NerBenchmark*

Benchmark of language models on the NER part of the SUC 3.0 dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.twitter_sent module

Sentiment evaluation of a language model on the TwitterSent dataset

class scandeval.benchmarks.twitter_sent.**TwitterSentBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.text_classification.TextClassificationBenchmark*

Benchmark of language models on the TwitterSent dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

verbose
Whether to print additional output.

Type bool

scandeval.benchmarks.wikiann_fo module

NER evaluation of a language model on the Faroese WikiANN dataset

class scandeval.benchmarks.wikiann_fo.**WikiannFoBenchmark**(*cache_dir: str = '.benchmark_models'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: *scandeval.benchmarks.abstract.ner.NerBenchmark*

Benchmark of language models on the Faroese WikiANN dataset.

Parameters

- **cache_dir** (*str, optional*) – Where the downloaded models will be stored. Defaults to '.benchmark_models'.
- **evaluate_train** (*bool, optional*) – Whether the models should be evaluated on the training scores. Defaults to False. Whether to prefer Jax for the evaluation. Defaults to False.
- **verbose** (*bool, optional*) – Whether to print additional output during evaluation. Defaults to False.

name
The name of the dataset.

Type str

task
The type of task to be benchmarked.

Type str

metric_names
The names of the metrics.

Type dict

id2label
A dictionary converting indices to labels.

Type dict or None

label2id
A dictionary converting labels to indices.

Type dict or None

num_labels
The number of labels in the dataset.

Type int or None

label_synonyms
Synonyms of the dataset labels.

Type list of lists of str

evaluate_train
Whether the training set should be evaluated.

Type bool

cache_dir
Directory where models are cached.

Type str

two_labels
Whether two labels should be predicted.

Type bool

split_point
Splitting point of *id2label* into labels.

Type int or None

Module contents

1.2 Submodules

1.3 scandeval.benchmark module

Fetches an updated list of all Scandinavian models on the HuggingFace Hub

class scandeval.benchmark.**Benchmark**(*progress_bar: bool = True*, *save_results: bool = False*, *language: Union[str, List[str]] = ['da', 'sv', 'no', 'nb', 'nn', 'is', 'fo']*, *task: Union[str, List[str]] = 'all'*, *evaluate_train: bool = False*, *verbose: bool = False*)

Bases: `object`

Benchmarking all the Scandinavian language models.

Parameters

- **progress_bar** (*bool, optional*) – Whether progress bars should be shown. Defaults to True.
- **save_results** (*bool, optional*) – Whether to save the benchmark results to 'scandeval_benchmark_results.json'. Defaults to False.
- **language** (*str or list of str, optional*) – The language codes of the languages to include in the list. Set this to 'all' if all languages (also non-Scandinavian) should be considered. Defaults to ['da', 'sv', 'no', 'nb', 'nn', 'is', 'fo'].
- **task** (*str or list of str, optional*) – The tasks to consider in the list. Set this to 'all' if all tasks should be considered. Defaults to 'all'.
- **evaluate_train** (*bool, optional*) – Whether to evaluate the training set as well. Defaults to False.
- **verbose** (*bool, optional*) – Whether to output additional output. Defaults to False.

progress_bar
Whether progress bars should be shown.

Type bool

save_results
Whether to save the benchmark results.

Type bool

language
The languages to include in the list.

Type str or list of str

task
The tasks to consider in the list.

Type str or list of str

evaluate_train
Whether to evaluate the training set as well.

Type bool

verbose
Whether to output additional output.

Type bool

benchmark_results
The benchmark results.

Type dict

benchmark(*model_id: Optional[Union[str, List[str]]] = None*, *dataset: Optional[Union[str, List[str]]] = None*, *progress_bar: Optional[bool] = None*, *save_results: Optional[bool] = None*, *language: Optional[Union[str, List[str]]] = None*, *task: Optional[Union[str, List[str]]] = None*, *evaluate_train: Optional[bool] = None*, *verbose: Optional[bool] = None*) → Dict[str, Dict[str, dict]]

Benchmarks models on datasets.

Parameters

- **model_id** (*str, list of str or None, optional*) – The model ID(s) of the models to benchmark. If None then all relevant model IDs will be benchmarked. Defaults to None.
- **dataset** (*str, list of str or None, optional*) – The datasets to benchmark on. If None then all datasets will be benchmarked. Defaults to None.
- **progress_bar** (*bool or None, optional*) – Whether progress bars should be shown. If None then the default value from the constructor will be used. Defaults to None.
- **save_results** (*bool or None, optional*) – Whether to save the benchmark results to 'scandeval_benchmark_results.json'. If None then the default value from the constructor will be used. Defaults to None.
- **language** (*str, list of str or None, optional*) – The language codes of the languages to include in the list. Set this to 'all' if all languages (also non-Scandinavian) should be considered. If None then the default value from the constructor will be used. Defaults to None.
- **task** (*str, list of str or None, optional*) – The tasks to consider in the list. Set this to 'all' if all tasks should be considered. If None then the default value from the constructor will be used. Defaults to None.
- **evaluate_train** (*bool or None, optional*) – Whether to evaluate the training set as well. If None then the default value from the constructor will be used. Defaults to None.
- **verbose** (*bool or None, optional*) – Whether to output additional output. If None then the default value from the constructor will be used. Defaults to None.

Returns A nested dictionary of the benchmark results. The keys are the names of the datasets, with values being new dictionaries having the model IDs as keys.

Return type dict

1.4 scandeval.cli module

Command-line interface for benchmarking

1.5 scandeval.datasets module

Functions that load datasets

scandeval.datasets.**load_absabank_imm**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the ABSAbank-Imm dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_angry_tweets**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the AngryTweets dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_dalaj**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the DaLaJ dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_dane**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the the DaNE dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_dataset**(*name: str*) → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load a benchmark dataset.

Parameters **name** (*str*) – Name of the dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

Raises **RuntimeError** – If *name* is not a valid dataset name.

scandeval.datasets.**load_ddt_dep**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the dependency parsing part of the DDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_ddt_pos**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the POS part of the DDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_dkhate**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the DKHate dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_europarl**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the Europarl dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_fdt_dep**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the dependency parsing part of the FDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_fdt_pos**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the POS part of the FDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_idt_dep**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the dependency parsing part of the IDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_idt_pos**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the POS part of the IDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_lcc**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the LCC dataset.

This dataset is the concatenation of the LCC1 and LCC2 datasets.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_mim_gold_ner**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the the MIM-GOLD-NER dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_ndt_nb_dep**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the Bokmål POS part of the NDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_ndt_nb_pos**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the Bokmål POS part of the NDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_ndt_nn_dep**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the Nynorsk POS part of the NDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_ndt_nn_pos**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the Nynorsk POS part of the NDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_nordial**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the Bokmål/Nynorsk part of the NorDial dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_norec**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the NoReC dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_norec_fo**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the NoReC-FO dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_norec_is**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the NoReC-IS dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_norne_nb**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the the Bokmål part of the NorNE dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_norne_nn**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the the Nynorsk part of the NorNE dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_sdt_dep**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the dependency parsing part of the SDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_sdt_pos**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the POS part of the SDT dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_suc3**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the SUC 3.0 dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_twitter_sent**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the TwitterSent dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

scandeval.datasets.**load_wikiann_fo**() → Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]

Load the the Faroese WikiANN dataset.

Returns Four dataframes, *X_train*, *X_test*, *y_train* and *y_test*, where *X_train* and *X_test* corresponds to the feature matrices for the training and test split, respectively, and *y_train* and *y_test* contains the target vectors.

Return type tuple

1.6 scandeval.utils module

Utility functions to be used in other scripts

class scandeval.utils.**DocInherit**(*mthd: Callable*)
Bases: object

Docstring inheriting method descriptor.

The class itself is also used as a decorator.

get_no_inst(*cls*)

get_with_inst(*obj*, *cls*)

use_parent_doc(*func*, *source*)

exception scandeval.utils.**InvalidBenchmark**(*message: str = 'This model cannot be benchmarked on the given dataset.'*)
Bases: Exception

class scandeval.utils.**NeverLeaveProgressCallback**
Bases: transformers.trainer_callback.ProgressCallback

Progress callback which never leaves the progress bar

on_prediction_step(*args*, *state*, *control*, *eval_dataloader=None*, ***kwargs*)
Event called after a prediction step.

on_train_begin(*args*, *state*, *control*, ***kwargs*)
Event called at the beginning of training.

class scandeval.utils.**TwolabelTrainer**(*split_point: int*, ***kwargs*)
Bases: transformers.trainer.Trainer

Trainer class which deals with two labels.

compute_loss(*model*, *inputs*, *return_outputs=False*)
How the loss is computed by Trainer. By default, all models return the loss in the first element.

Subclass and override for custom behavior.

scandeval.utils.**block_terminal_output**()
Blocks libraries from writing output to the terminal

scandeval.utils.**doc_inherit**
alias of *scandeval.utils.DocInherit*

scandeval.utils.**get_all_datasets**() → list
Load a list of all datasets.

Returns First entry in each tuple is the short name of the dataset, second entry the long name, third entry the benchmark class and fourth entry the loading function.

Return type list of tuples

scandeval.utils.**is_module_installed**(*module: str*) → bool
Check if a module is installed.

Parameters **module** (*str*) – The name of the module.

Returns Whether the module is installed or not.

Return type bool

1.7 Module contents

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

S

INDEX

D

E

F

G

I

L

M

N

O

P

S

T