

[![Logo](../../_static/logo.png)](../..../index.html)

Getting Started

- * [Installation](../..../installation.html)
- * [Install with pip](../..../installation.html#install-with-pip)
- * [Install with Conda](../..../installation.html#install-with-conda)
- * [Install from Source](../..../installation.html#install-from-source)
- * [Editable Install](../..../installation.html#editable-install)
- * [Install PyTorch with CUDA support](../..../installation.html#install-pytorch-with-cuda-support)
- * [Quickstart](../..../quickstart.html)
- * [Sentence Transformer](../..../quickstart.html#sentence-transformer)
- * [Cross Encoder](../..../quickstart.html#cross-encoder)
- * [Next Steps](../..../quickstart.html#next-steps)

Sentence Transformer

- * [Usage](../..../sentence_transformer/usage/usage.html)
- * [Computing Embeddings](../..../examples/applications/computing-embeddings/README.html)
 - * [Initializing a Sentence Transformer Model](../..../examples/applications/computing-embeddings/README.html#initializing-a-sentence-transformer-model)
 - * [Calculating Embeddings](../..../examples/applications/computing-embeddings/README.html#calculating-embeddings)
 - * [Prompt Templates](../..../examples/applications/computing-embeddings/README.html#prompt-templates)

[* \[Input Sequence Length\]\(../../examples/applications/computing-embeddings/README.html#id1\)](#)

[* \[Multi-Process / Multi-GPU Encoding\]\(../../examples/applications/computing-embeddings/README.html#multi-process-multi-gpu-encoding\)](#)

[* \[Semantic Textual Similarity\]\(../../sentence_transformer/usage/semantic_textual_similarity.html\)](#)

[* \[Similarity Calculation\]\(../../sentence_transformer/usage/semantic_textual_similarity.html#similarity-calculation\)](#)

[* \[Semantic Search\]\(../../examples/applications/semantic-search/README.html\)](#)

[* \[Background\]\(../../examples/applications/semantic-search/README.html#background\)](#)

[* \[Symmetric vs. Asymmetric Semantic Search\]\(../../examples/applications/semantic-search/README.html#symmetric-vs-asymmetric-semantic-search\)](#)

[* \[Manual Implementation\]\(../../examples/applications/semantic-search/README.html#manual-implementation\)](#)

[* \[Optimized Implementation\]\(../../examples/applications/semantic-search/README.html#optimized-implementation\)](#)

[* \[Speed Optimization\]\(../../examples/applications/semantic-search/README.html#speed-optimization\)](#)

[* \[Elasticsearch\]\(../../examples/applications/semantic-search/README.html#elasticsearch\)](#)

[* \[Approximate Nearest Neighbor\]\(../../examples/applications/semantic-search/README.html#approximate-nearest-neighbor\)](#)

[* \[Retrieve & Re-Rank\]\(../../examples/applications/semantic-search/README.html#retrieve-re-rank\)](#)

* [Examples](../../examples/applications/semantic-search/README.html#examples)

* [Retrieve & Re-Rank](../../examples/applications/retrieve_rerank/README.html)

* [Retrieve & Re-Rank

Pipeline](../../examples/applications/retrieve_rerank/README.html#retrieve-re-rank-pipeline)

* [Retrieval:

Bi-Encoder](../../examples/applications/retrieve_rerank/README.html#retrieval-bi-encoder)

* [Re-Ranker:

Cross-Encoder](../../examples/applications/retrieve_rerank/README.html#re-ranker-cross-encoder)

* [Example Scripts](../../examples/applications/retrieve_rerank/README.html#example-scripts)

* [Pre-trained Bi-Encoders

(Retrieval)](../../examples/applications/retrieve_rerank/README.html#pre-trained-bi-encoders-retrieval)

* [Pre-trained Cross-Encoders

(Re-Ranker)](../../examples/applications/retrieve_rerank/README.html#pre-trained-cross-encoders-re-ranker)

* [Clustering](../../examples/applications/clustering/README.html)

* [k-Means](../../examples/applications/clustering/README.html#k-means)

* [Agglomerative

Clustering](../../examples/applications/clustering/README.html#agglomerative-clustering)

* [Fast Clustering](../../examples/applications/clustering/README.html#fast-clustering)

* [Topic Modeling](../../examples/applications/clustering/README.html#topic-modeling)

* [Paraphrase Mining](../../examples/applications/paraphrase-mining/README.html)

*

[`paraphrase_mining()`](../../examples/applications/paraphrase-mining/README.html#sentence_transformers.util.paraphrase_mining)

* [Translated Sentence

Mining](../../examples/applications/parallel-sentence-mining/README.html)

* [Margin Based

Mining](../../examples/applications/parallel-sentence-mining/README.html#margin-based-mining)

* [Examples](../../examples/applications/parallel-sentence-mining/README.html#examples)

* [Image Search](../../examples/applications/image-search/README.html)

* [Installation](../../examples/applications/image-search/README.html#installation)

* [Usage](../../examples/applications/image-search/README.html#usage)

* [Examples](../../examples/applications/image-search/README.html#examples)

* [Embedding Quantization](../../examples/applications/embedding-quantization/README.html)

* [Binary

Quantization](../../examples/applications/embedding-quantization/README.html#binary-quantization)

* [Scalar (int8)

Quantization](../../examples/applications/embedding-quantization/README.html#scalar-int8-quantization)

* [Additional

extensions](../../examples/applications/embedding-quantization/README.html#additional-extensions)

* [Demo](../../examples/applications/embedding-quantization/README.html#demo)

* [Try it

yourself](../../examples/applications/embedding-quantization/README.html#try-it-yourself)

* [Speeding up Inference](../../sentence_transformer/usage/efficiency.html)

* [PyTorch](../../sentence_transformer/usage/efficiency.html#pytorch)

* [ONNX](../../sentence_transformer/usage/efficiency.html#onnx)

* [OpenVINO](../../sentence_transformer/usage/efficiency.html#openvino)

* [Benchmarks](../../sentence_transformer/usage/efficiency.html#benchmarks)

* [Creating Custom Models](../../sentence_transformer/usage/custom_models.html)

* [Structure of Sentence Transformer

Models](../../sentence_transformer/usage/custom_models.html#structure-of-sentence-transformer-models)

* [Sentence Transformer Model from a Transformers

Model](../../sentence_transformer/usage/custom_models.html#sentence-transformer-model-from-a-transformers-model)

* [Pretrained Models](../../sentence_transformer/pretrained_models.html)

* [Original Models](../../sentence_transformer/pretrained_models.html#original-models)

* [Semantic Search

Models](../../sentence_transformer/pretrained_models.html#semantic-search-models)

* [Multi-QA Models](../../sentence_transformer/pretrained_models.html#multi-qa-models)

* [MSMARCO Passage

Models](../../sentence_transformer/pretrained_models.html#msmarco-passage-models)

* [Multilingual Models](../../sentence_transformer/pretrained_models.html#multilingual-models)

* [Semantic Similarity

Models](../../sentence_transformer/pretrained_models.html#semantic-similarity-models)

* [Bitext Mining](../../sentence_transformer/pretrained_models.html#bitext-mining)

* [Image & Text-Models](../../sentence_transformer/pretrained_models.html#image-text-models)

* [INSTRUCTOR models](../../sentence_transformer/pretrained_models.html#instructor-models)

* [Scientific Similarity

Models](../../sentence_transformer/pretrained_models.html#scientific-similarity-models)

* [Training Overview](../../sentence_transformer/training_overview.html)

* [Why Finetune?](../../sentence_transformer/training_overview.html#why-finetune)

* [Training Components](../../sentence_transformer/training_overview.html#training-components)

* [Dataset](../../sentence_transformer/training_overview.html#dataset)

* [Dataset Format](../../sentence_transformer/training_overview.html#dataset-format)

* [Loss Function](../../sentence_transformer/training_overview.html#loss-function)

- * [Training Arguments](../../sentence_transformer/training_overview.html#training-arguments)
- * [Evaluator](../../sentence_transformer/training_overview.html#evaluator)
- * [Trainer](../../sentence_transformer/training_overview.html#trainer)
- * [Callbacks](../../sentence_transformer/training_overview.html#callbacks)
- * [Multi-Dataset Training](../../sentence_transformer/training_overview.html#multi-dataset-training)
- * [Deprecated Training](../../sentence_transformer/training_overview.html#deprecated-training)
- * [Best Base Embedding Models](../../sentence_transformer/training_overview.html#best-base-embedding-models)
- * [Dataset Overview](../../sentence_transformer/dataset_overview.html)
 - * [Datasets on the Hugging Face Hub](../../sentence_transformer/dataset_overview.html#datasets-on-the-hugging-face-hub)
 - * [Pre-existing Datasets](../../sentence_transformer/dataset_overview.html#pre-existing-datasets)
- * [Loss Overview](../../sentence_transformer/loss_overview.html)
 - * [Loss modifiers](../../sentence_transformer/loss_overview.html#loss-modifiers)
 - * [Distillation]
 - * [Commonly used Loss Functions](../../sentence_transformer/loss_overview.html#commonly-used-loss-functions)
 - * [Custom Loss Functions](../../sentence_transformer/loss_overview.html#custom-loss-functions)
- * [Training Examples](../../sentence_transformer/training/examples.html)
 - * [Semantic Textual Similarity](../../examples/training/sts/README.html)
 - * [Training data](../../examples/training/sts/README.html#training-data)
 - * [Loss Function](../../examples/training/sts/README.html#loss-function)
 - * [Natural Language Inference](../../examples/training/nli/README.html)
 - * [Data](../../examples/training/nli/README.html#data)
 - * [SoftmaxLoss](../../examples/training/nli/README.html#softmaxloss)
 - * [MultipleNegativesRankingLoss](../../examples/training/nli/README.html#multiplenegativesrankin

gloss)

- * [Paraphrase Data](../../examples/training/paraphrases/README.html)
- * [Pre-Trained Models](../../examples/training/paraphrases/README.html#pre-trained-models)
- * [Quora Duplicate Questions](../../examples/training/quora_duplicate_questions/README.html)
- * [Training](../../examples/training/quora_duplicate_questions/README.html#training)

*

[MultipleNegativesRankingLoss](../../examples/training/quora_duplicate_questions/README.html#multiplenegativesrankingloss)

*

[Pretrained

Models](../../examples/training/quora_duplicate_questions/README.html#pretrained-models)

- * [MS MARCO](../../examples/training/ms_marco/README.html)
- * [Bi-Encoder](../../examples/training/ms_marco/README.html#bi-encoder)
- * [Matryoshka Embeddings](../../examples/training/matryoshka/README.html)
- * [Use Cases](../../examples/training/matryoshka/README.html#use-cases)
- * [Results](../../examples/training/matryoshka/README.html#results)
- * [Training](../../examples/training/matryoshka/README.html#training)
- * [Inference](../../examples/training/matryoshka/README.html#inference)
- * [Code Examples](../../examples/training/matryoshka/README.html#code-examples)
- * [Adaptive Layers](../../examples/training/adaptive_layer/README.html)
- * [Use Cases](../../examples/training/adaptive_layer/README.html#use-cases)
- * [Results](../../examples/training/adaptive_layer/README.html#results)
- * [Training](../../examples/training/adaptive_layer/README.html#training)
- * [Inference](../../examples/training/adaptive_layer/README.html#inference)
- * [Code Examples](../../examples/training/adaptive_layer/README.html#code-examples)
- * [Multilingual Models](../../examples/training/multilingual/README.html)

*

[Extend your own

models](../../examples/training/multilingual/README.html#extend-your-own-models)

- * [Training](../../examples/training/multilingual/README.html#training)
- * [Datasets](../../examples/training/multilingual/README.html#datasets)
 - * [Sources for Training Data](../../examples/training/multilingual/README.html#sources-for-training-data)
 - * [Evaluation](../../examples/training/multilingual/README.html#evaluation)
 - * [Available Pre-trained Models](../../examples/training/multilingual/README.html#available-pre-trained-models)
 - * [Usage](../../examples/training/multilingual/README.html#usage)
 - * [Performance](../../examples/training/multilingual/README.html#performance)
 - * [Citation](../../examples/training/multilingual/README.html#citation)
 - * [Model Distillation](../../examples/training/distillation/README.html)
 - * [Knowledge Distillation](../../examples/training/distillation/README.html#knowledge-distillation)
 - * [Speed - Performance Trade-Off](../../examples/training/distillation/README.html#speed-performance-trade-off)
 - * [Dimensionality Reduction](../../examples/training/distillation/README.html#dimensionality-reduction)
 - * [Quantization](../../examples/training/distillation/README.html#quantization)
 - * [Augmented SBERT](../../examples/training/data_augmentation/README.html)
 - * [Motivation](../../examples/training/data_augmentation/README.html#motivation)
 - * [Extend to your own datasets](../../examples/training/data_augmentation/README.html#extend-to-your-own-datasets)
 - * [Methodology](../../examples/training/data_augmentation/README.html#methodology)
 - * [Scenario 1: Limited or small annotated datasets (few labeled sentence-pairs)](../../examples/training/data_augmentation/README.html#scenario-1-limited-or-small-annotated-datasets-few-labeled-sentence-pairs)
 - * [Scenario 2: No annotated datasets (Only unlabeled

[sentence-pairs\)\]\(../../examples/training/data_augmentation/README.html#scenario-2-no-annotated-datasets-only-unlabeled-sentence-pairs\)](#)

- * [\[Training\]\(../../examples/training/data_augmentation/README.html#training\)](#)
- * [\[Citation\]\(../../examples/training/data_augmentation/README.html#citation\)](#)
- * [\[Training with Prompts\]\(../../examples/training/prompts/README.html\)](#)
- * [\[What are Prompts?\]\(../../examples/training/prompts/README.html#what-are-prompts\)](#)
 - * [\[Why would we train with Prompts?\]\(../../examples/training/prompts/README.html#why-would-we-train-with-prompts\)](#)
 - * [\[How do we train with Prompts?\]\(../../examples/training/prompts/README.html#how-do-we-train-with-prompts\)](#)
- * [\[Training with PEFT Adapters\]\(../../examples/training/peft/README.html\)](#)
- * [\[Compatibility Methods\]\(../../examples/training/peft/README.html#compatibility-methods\)](#)
- * [\[Adding a New Adapter\]\(../../examples/training/peft/README.html#adding-a-new-adapter\)](#)
 - * [\[Loading a Pretrained Adapter\]\(../../examples/training/peft/README.html#loading-a-pretrained-adapter\)](#)
- * [\[Training Script\]\(../../examples/training/peft/README.html#training-script\)](#)
- * [\[Unsupervised Learning\]\(../../examples/unsupervised_learning/README.html\)](#)
- * [\[TSDAE\]\(../../examples/unsupervised_learning/README.html#tsdae\)](#)
- * [\[SimCSE\]\(../../examples/unsupervised_learning/README.html#simcse\)](#)
- * [\[CT\]\(../../examples/unsupervised_learning/README.html#ct\)](#)
 - * [\[CT \(In-Batch Negative Sampling\)\]\(../../examples/unsupervised_learning/README.html#ct-in-batch-negative-sampling\)](#)
 - * [\[Masked Language Model \(MLM\)\]\(../../examples/unsupervised_learning/README.html#masked-language-model-mlm\)](#)
- * [\[GenQ\]\(../../examples/unsupervised_learning/README.html#genq\)](#)
- * [\[GPL\]\(../../examples/unsupervised_learning/README.html#gpl\)](#)
- * [\[Performance](#)

[Comparison\]\(../../examples/unsupervised_learning/README.html#performance-comparison\)](#)

* [\[Domain Adaptation\]\(../../examples/domain_adaptation/README.html\)](#)

* [\[Domain Adaptation vs. Unsupervised Learning\]\(../../examples/domain_adaptation/README.html#domain-adaptation-vs-unsupervised-learning\)](#)

* [\[Adaptive Pre-Training\]\(../../examples/domain_adaptation/README.html#adaptive-pre-training\)](#)

* [\[GPL: Generative Pseudo-Labeling\]\(../../examples/domain_adaptation/README.html#gpl-generative-pseudo-labeling\)](#)

* [\[Hyperparameter Optimization\]\(../../examples/training/hpo/README.html\)](#)

* [\[HPO Components\]\(../../examples/training/hpo/README.html#hpo-components\)](#)

* [\[Putting It All Together\]\(../../examples/training/hpo/README.html#putting-it-all-together\)](#)

* [\[Example Scripts\]\(../../examples/training/hpo/README.html#example-scripts\)](#)

* [\[Distributed Training\]\(../../sentence_transformer/training/distributed.html\)](#)

* [\[Comparison\]\(../../sentence_transformer/training/distributed.html#comparison\)](#)

* [\[FSDP\]\(../../sentence_transformer/training/distributed.html#fsdp\)](#)

Cross Encoder

* [\[Usage\]\(../../cross_encoder/usage/usage.html\)](#)

* [\[Retrieve & Re-Rank\]\(../../examples/applications/retrieve_rerank/README.html\)](#)

* [\[Retrieve & Re-Rank Pipeline\]\(../../examples/applications/retrieve_rerank/README.html#retrieve-re-rank-pipeline\)](#)

* [\[Retrieval: Bi-Encoder\]\(../../examples/applications/retrieve_rerank/README.html#retrieval-bi-encoder\)](#)

* [\[Re-Ranker:](#)

Cross-Encoder](../../examples/applications/retrieve_rerank/README.html#re-ranker-cross-encoder)

* [Example Scripts](../../examples/applications/retrieve_rerank/README.html#example-scripts)

* [Pre-trained Bi-Encoders (Retrieval)](../../examples/applications/retrieve_rerank/README.html#pre-trained-bi-encoders-retrieval)

* [Pre-trained Cross-Encoders (Re-Ranker)](../../examples/applications/retrieve_rerank/README.html#pre-trained-cross-encoders-re-ranker)

* [Pretrained Models](../../cross_encoder/pretrained_models.html)

* [MS MARCO](../../cross_encoder/pretrained_models.html#ms-marco)

* [SQuAD (QNLI)](../../cross_encoder/pretrained_models.html#squad-qnli)

* [STSbenchmark](../../cross_encoder/pretrained_models.html#stsbenchmark)

* [Quora Duplicate Questions](../../cross_encoder/pretrained_models.html#quora-duplicate-questions)

* [NLI](../../cross_encoder/pretrained_models.html#nli)

* [Community Models](../../cross_encoder/pretrained_models.html#community-models)

* [Training Overview](../../cross_encoder/training_overview.html)

* [Training Examples](../../cross_encoder/training/examples.html)

* [MS MARCO](../../examples/training/ms_marco/cross_encoder_README.html)

*

[Cross-Encoder](../../examples/training/ms_marco/cross_encoder_README.html#cross-encoder)

* [Cross-Encoder Knowledge Distillation](../../examples/training/ms_marco/cross_encoder_README.html#cross-encoder-knowledge-distillation)

Package Reference

* [Sentence Transformer](../sentence_transformer/index.html)

* [SentenceTransformer](../sentence_transformer/SentenceTransformer.html)

* [SentenceTransformer](../sentence_transformer/SentenceTransformer.html#id1)

*

[SentenceTransformerModelCardData](../sentence_transformer/SentenceTransformer.html#sentence-transformer-model-card-data)

* [SimilarityFunction](../sentence_transformer/SentenceTransformer.html#similarity-function)

* [Trainer](../sentence_transformer/trainer.html)

*

[SentenceTransformerTrainer](../sentence_transformer/trainer.html#sentence-transformer-trainer)

* [Training Arguments](../sentence_transformer/training_args.html)

*

[SentenceTransformerTrainingArguments](../sentence_transformer/training_args.html#sentence-transformer-training-arguments)

* [Losses](../sentence_transformer/losses.html)

* [BatchAllTripletLoss](../sentence_transformer/losses.html#batch-all-triplet-loss)

*

[BatchHardSoftMarginTripletLoss](../sentence_transformer/losses.html#batch-hard-soft-margin-triplet-loss)

* [BatchHardTripletLoss](../sentence_transformer/losses.html#batch-hard-triplet-loss)

* [BatchSemiHardTripletLoss](../sentence_transformer/losses.html#batch-semi-hard-triplet-loss)

* [ContrastiveLoss](../sentence_transformer/losses.html#contrastive-loss)

* [OnlineContrastiveLoss](../sentence_transformer/losses.html#online-contrastive-loss)

* [ContrastiveTensionLoss](../sentence_transformer/losses.html#contrastive-tension-loss)

*

[ContrastiveTensionLossInBatchNegatives](../sentence_transformer/losses.html#contrastive-tension-loss-in-batch-negatives)

ossinbatchnegatives)

- * [CoSENTLoss](../sentence_transformer/losses.html#cosentloss)
- * [AngleLoss](../sentence_transformer/losses.html#angleloss)
- * [CosineSimilarityLoss](../sentence_transformer/losses.html#cosinesimilarityloss)
- * [DenoisingAutoEncoderLoss](../sentence_transformer/losses.html#denoisingautoencoderloss)
- * [GISTEmbedLoss](../sentence_transformer/losses.html#gistembedloss)
- * [CachedGISTEmbedLoss](../sentence_transformer/losses.html#cachedgistembedloss)
- * [MSELoss](../sentence_transformer/losses.html#mseloss)
- * [MarginMSELoss](../sentence_transformer/losses.html#marginmseloss)
- * [MatryoshkaLoss](../sentence_transformer/losses.html#matryoshkaloss)
- * [Matryoshka2dLoss](../sentence_transformer/losses.html#matryoshka2dloss)
- * [AdaptiveLayerLoss](../sentence_transformer/losses.html#adaptivelayerloss)
- * [MegaBatchMarginLoss](../sentence_transformer/losses.html#megabatchmarginloss)

*

[MultipleNegativesRankingLoss](../sentence_transformer/losses.html#multiplenegativesrankingloss)

*

[CachedMultipleNegativesRankingLoss](../sentence_transformer/losses.html#cachedmultiplenegativesrankingloss)

*

[MultipleNegativesSymmetricRankingLoss](../sentence_transformer/losses.html#multiplenegativessymmetricrankingloss)

*

[CachedMultipleNegativesSymmetricRankingLoss](../sentence_transformer/losses.html#cachedmultiplenegativessymmetricrankingloss)

- * [SoftmaxLoss](../sentence_transformer/losses.html#softmaxloss)
- * [TripletLoss](../sentence_transformer/losses.html#tripletloss)
- * [Samplers](../sentence_transformer/sampler.html)

* [BatchSamplers](../sentence_transformer/sampler.html#batchsamplers)

*

[MultiDatasetBatchSamplers](../sentence_transformer/sampler.html#multidatasetbatchsamplers)

* [Evaluation](../sentence_transformer/evaluation.html)

*

[BinaryClassificationEvaluator](../sentence_transformer/evaluation.html#binaryclassificationevaluator)

*

[EmbeddingSimilarityEvaluator](../sentence_transformer/evaluation.html#embeddingsimilarityevaluator)

*

[InformationRetrievalEvaluator](../sentence_transformer/evaluation.html#informationretrievalevaluator)

* [NanoBEIREvaluator](../sentence_transformer/evaluation.html#nanobeirevaluator)

* [MSEEvaluator](../sentence_transformer/evaluation.html#mseevaluator)

*

[ParaphraseMiningEvaluator](../sentence_transformer/evaluation.html#paraphraseminingevaluator)

* [RerankingEvaluator](../sentence_transformer/evaluation.html#rerankingevaluator)

* [SentenceEvaluator](../sentence_transformer/evaluation.html#sentenceevaluator)

* [SequentialEvaluator](../sentence_transformer/evaluation.html#sequentialevaluator)

* [TranslationEvaluator](../sentence_transformer/evaluation.html#translationevaluator)

* [TripletEvaluator](../sentence_transformer/evaluation.html#tripletevaluator)

* [Datasets](../sentence_transformer/datasets.html)

* [ParallelSentencesDataset](../sentence_transformer/datasets.html#parallelsentencesdataset)

* [SentenceLabelDataset](../sentence_transformer/datasets.html#sentencelabeldataset)

*

[DenoisingAutoEncoderDataset](../sentence_transformer/datasets.html#denoisingautoencoderdataset)

et)

- * [NoDuplicatesDataLoader](../sentence_transformer/datasets.html#noduplicatesdataloader)

- * [Models](../sentence_transformer/models.html)

- * [Main Classes](../sentence_transformer/models.html#main-classes)

- * [Further Classes](../sentence_transformer/models.html#further-classes)

- * [quantization](../sentence_transformer/quantization.html)

*

[`quantize_embeddings()`](../sentence_transformer/quantization.html#sentence_transformers.quantization.quantize_embeddings)

*

[`semantic_search_faiss()`](../sentence_transformer/quantization.html#sentence_transformers.quantization.semantic_search_faiss)

*

[`semantic_search_usearch()`](../sentence_transformer/quantization.html#sentence_transformers.quantization.semantic_search_usearch)

- * [Cross Encoder](index.html)

- * [CrossEncoder](cross_encoder.html)

- * [CrossEncoder](cross_encoder.html#id1)

- * [Training Inputs](cross_encoder.html#training-inputs)

- * Evaluation

- * CEBinaryAccuracyEvaluator

- * CEBinaryClassificationEvaluator

- * CECorrelationEvaluator

- * CEF1Evaluator

- * CESoftmaxAccuracyEvaluator

- * CERerankingEvaluator

- * [util](../util.html)

- * [Helper Functions](../util.html#module-sentence_transformers.util)
- * [community_detection()](../util.html#sentence_transformers.util.community_detection)
- * [http_get()](../util.html#sentence_transformers.util.http_get)
- * [is_training_available()](../util.html#sentence_transformers.util.is_training_available)
- * [mine_hard_negatives()](../util.html#sentence_transformers.util.mine_hard_negatives)
- * [normalize_embeddings()](../util.html#sentence_transformers.util.normalize_embeddings)
- * [paraphrase_mining()](../util.html#sentence_transformers.util.paraphrase_mining)
- * [semantic_search()](../util.html#sentence_transformers.util.semantic_search)
- * [truncate_embeddings()](../util.html#sentence_transformers.util.truncate_embeddings)
- * [Model Optimization](../util.html#module-sentence_transformers.backend)

*

[export_dynamic_quantized_onnx_model()](../util.html#sentence_transformers.backend.export_dynamic_quantized_onnx_model)

*

[export_optimized_onnx_model()](../util.html#sentence_transformers.backend.export_optimized_onnx_model)

*

[export_static_quantized_openvino_model()](../util.html#sentence_transformers.backend.export_static_quantized_openvino_model)

- * [Similarity Metrics](../util.html#module-sentence_transformers.util)
- * [cos_sim()](../util.html#sentence_transformers.util.cos_sim)
- * [dot_score()](../util.html#sentence_transformers.util.dot_score)
- * [euclidean_sim()](../util.html#sentence_transformers.util.euclidean_sim)
- * [manhattan_sim()](../util.html#sentence_transformers.util.manhattan_sim)
- * [pairwise_cos_sim()](../util.html#sentence_transformers.util.pairwise_cos_sim)
- * [pairwise_dot_score()](../util.html#sentence_transformers.util.pairwise_dot_score)
- * [pairwise_euclidean_sim()](../util.html#sentence_transformers.util.pairwise_euclidean_sim)

* [pairwise_manhattan_sim()](../util.html#sentence_transformers.util.pairwise_manhattan_sim)

__[Sentence Transformers](../index.html)

* [(../index.html)]

* [Cross Encoder](index.html)

* Evaluation

* [Edit on

GitHub](https://github.com/UKPLab/sentence-transformers/blob/master/docs/package_reference/cross_encoder/evaluation.md)

* * *

Evaluation

CrossEncoder have their own evaluation classes, that are in

``sentence_transformers.cross_encoder.evaluation``.

CEBinaryAccuracyEvaluator

`_class`

`_sentence_transformers.cross_encoder.evaluation.CEBinaryAccuracyEvaluator(_sentence_pairs`

`: list[list[str]], _labels : list[int], _name : str = "_", _threshold :`

`float = 0.5, _write_csv : bool =`

`True_)`[\[\[source\]\]\(https://github.com/UKPLab/sentence-](https://github.com/UKPLab/sentence-transformers/blob/master/sentence_transformers\\cross_encoder\\evaluation\\CEBinaryAccuracyEvaluator.py#L14-L80)

[transformers/blob/master/sentence_transformers\\cross_encoder\\evaluation\\CEBinaryAccuracyEvaluator.py#L14-L80\)](https://github.com/UKPLab/sentence-transformers/blob/master/sentence_transformers\\cross_encoder\\evaluation\\CEBinaryAccuracyEvaluator.py#L14-L80)

This evaluator can be used with the CrossEncoder class.

It is designed for CrossEncoders with 1 outputs. It measure the accuracy of the predict class vs. the gold labels. It uses a fixed threshold to determine the label (0 vs 1).

See CEBinaryClassificationEvaluator for an evaluator that determines automatically the optimal threshold.

CEBinaryClassificationEvaluator¶

_class

_sentence_transformers.cross_encoder.evaluation.CEBinaryClassificationEvaluator(_sentence_pairs

: list[list[str]], _labels : list[int], _name : str = "_",

_show_progress_bar : bool = False, _write_csv : bool =

True_)[[source]]([https://github.com/UKPLab/sentence-](https://github.com/UKPLab/sentence-transformers/blob/master/sentence_transformers\\cross_encoder\\evaluation\\CEBinaryClassificationEvaluator.py#L16-L104)

[transformers/blob/master/sentence_transformers\\cross_encoder\\evaluation\\CEBinaryClassificationEvaluator.py#L16-L104](https://github.com/UKPLab/sentence-transformers/blob/master/sentence_transformers\\cross_encoder\\evaluation\\CEBinaryClassificationEvaluator.py#L16-L104))¶

This evaluator can be used with the CrossEncoder class. Given sentence pairs and binary labels (0 and 1), it compute the average precision and the best

possible f1 score

```
## CECorrelationEvaluatorïf•
```

```
_class
```

```
_sentence_transformers.cross_encoder.evaluation.CECorrelationEvaluator(_sentence_pairs
```

```
: list[list[str]]_, _scores : list[float]_, _name : str = "_", _write_csv :
```

```
bool = True_) [[source]](https://github.com/UKPLab/sentence-
```

```
transformers/blob/master/sentence_transformers\\cross_encoder\\evaluation\\CECorrelationEvaluat
```

```
or.py#L14-L67)ïf•
```

This evaluator can be used with the CrossEncoder class. Given sentence pairs and continuous scores, it compute the pearson & spearman correlation between the predicted score for the sentence pair and the gold score.

```
## CEF1Evaluatorïf•
```

```
_class
```

```
_sentence_transformers.cross_encoder.evaluation.CEF1Evaluator(_sentence_pairs
```

```
: list[list[str]]_, _labels : list[int]_, _*_ , _batch_size : int = 32_,
```

```
_show_progress_bar : bool = False_, _name : str = "_", _write_csv : bool =
```

```
True_) [[source]](https://github.com/UKPLab/sentence-
```

```
transformers/blob/master/sentence_transformers\\cross_encoder\\evaluation\\CEF1Evaluator.py#L1
```

```
6-L140)ïf•
```

CrossEncoder F1 score based evaluator for binary and multiclass tasks.

The task type (binary or multiclass) is determined from the labels array. For binary tasks the returned metric is binary F1 score. For the multiclass tasks the returned metric is macro F1 score.

Parameters:

sentence_pairs (`List[List[str]]`) â€” A list of sentence pairs, where each pair is a list of two strings.

labels (`List[int]`) â€” A list of integer labels corresponding to each sentence pair.

batch_size (`int`, `optional`) â€” Batch size for prediction. Defaults to 32.

show_progress_bar (`bool`, `optional`) â€” Show tqdm progress bar.

name (`str`, `optional`) â€” An optional name for the CSV file with stored results. Defaults to an empty string.

write_csv (`bool`, `optional`) â€” Flag to determine if the data should be saved to a CSV file. Defaults to True.

CESoftmaxAccuracyEvaluator¶

_class

_sentence_transformers.cross_encoder.evaluation.CESoftmaxAccuracyEvaluator(_sentence_pairs : list[list[str]], _labels : list[int], _name : str = "_", _write_csv : bool = True_)[[source]](https://github.com/UKPLab/sentence-transformers/blob/master/sentence_transformers\\cross_encoder\\evaluation\\CESoftmaxAccuracyEvaluator.py#L14-L70)¶

This evaluator can be used with the CrossEncoder class.

It is designed for CrossEncoders with 2 or more outputs. It measure the accuracy of the predict class vs. the gold labels.

CERerankingEvaluator¶

_class _sentence_transformers.cross_encoder.evaluation.CERerankingEvaluator(_samples_ , _at_k : int = 10_, _name : str = "_", _write_csv : bool = True_, _mrr_at_k : int | None = None_)[[source]](https://github.com/UKPLab/sentence-transformers/blob/master/sentence_transformers\\cross_encoder\\evaluation\\CERerankingEvaluator.py#L13-L109)¶

This class evaluates a CrossEncoder model for the task of re-ranking.

Given a query and a list of documents, it computes the score [query, doc_i]

for all possible documents and sorts them in decreasing order. Then,

[MRR@10](/cdn-cgi/l/email-

protection#ffb2adadd9dcccc8c4d9dccacdc4d9dccbc7c4cecf) and [NDCG@10]/(cdn-

cgi//email-protection#703e343337565343474b565345424b565344484b4140) are

computed to measure the quality of the ranking.

Parameters:

****samples**** (List [Dict , str , Union [str , List [str , str]]]) ∈ “Must be a list and each element is of the form: { \tilde{query} }TM:

 $\hat{a} \in \hat{a}^{\text{TM}}, \hat{a} \in \text{positive} \hat{a}^{\text{TM}}: [], \hat{a} \in \text{negative} \hat{a}^{\text{TM}}: []$. Query is the search query,

positive is a list of positive (relevant) documents, negative is a list of

negative (irrelevant) documents.

[\[Previous\]\(cross_encoder.html "CrossEncoder"\)](#) [\[Next \]\(./util.html "util"\)](#)

* * *

(C) Copyright 2025.

Built with [Sphinx](https://www.sphinx-doc.org/) using a

[theme](https://github.com/readthedocs/sphinx_rtd_theme) provided by [Read the

Docs](<https://readthedocs.org>).