

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

Getting Started

- * [Installation](../../docs/installation.html)

- * [Install with pip](../../docs/installation.html#install-with-pip)

- * [Install with Conda](../../docs/installation.html#install-with-conda)

- * [Install from Source](../../docs/installation.html#install-from-source)

- * [Editable Install](../../docs/installation.html#editable-install)

- * [Install PyTorch with CUDA support](../../docs/installation.html#install-pytorch-with-cuda-support)

- * [Quickstart](../../docs/quickstart.html)

- * [Sentence Transformer](../../docs/quickstart.html#sentence-transformer)

- * [Cross Encoder](../../docs/quickstart.html#cross-encoder)

- * [Next Steps](../../docs/quickstart.html#next-steps)

Sentence Transformer

- * [Usage](../../docs/sentence_transformer/usage/usage.html)

- * [Computing Embeddings](../computing-embeddings/README.html)

- * [Initializing a Sentence Transformer Model](../computing-embeddings/README.html#initializing-a-sentence-transformer-model)

- * [Calculating Embeddings](../computing-embeddings/README.html#calculating-embeddings)

- * [Prompt Templates](../computing-embeddings/README.html#prompt-templates)

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

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

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

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

* [Semantic Search](../semantic-search/README.html)

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

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

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

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

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

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

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

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

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

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

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

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

* [Re-Ranker: Cross-Encoder](../retrieve_rerank/README.html#re-ranker-cross-encoder)

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

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

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

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

- * [k-Means](../clustering/README.html#k-means)
- * [Agglomerative Clustering](../clustering/README.html#agglomerative-clustering)
- * [Fast Clustering](../clustering/README.html#fast-clustering)
- * [Topic Modeling](../clustering/README.html#topic-modeling)
- * [Paraphrase Mining](../paraphrase-mining/README.html)

*

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

- * [Translated Sentence Mining](../parallel-sentence-mining/README.html)
- * [Margin Based Mining](../parallel-sentence-mining/README.html#margin-based-mining)
- * [Examples](../parallel-sentence-mining/README.html#examples)
- * [Image Search](../image-search/README.html)
- * [Installation](../image-search/README.html#installation)
- * [Usage](../image-search/README.html#usage)
- * [Examples](../image-search/README.html#examples)
- * [Embedding Quantization](../embedding-quantization/README.html)
- * [Binary Quantization](../embedding-quantization/README.html#binary-quantization)
- * [Scalar (int8) Quantization](../embedding-quantization/README.html#scalar-int8-quantization)
- * [Additional extensions](../embedding-quantization/README.html#additional-extensions)
- * [Demo](../embedding-quantization/README.html#demo)
- * [Try it yourself](../embedding-quantization/README.html#try-it-yourself)
- * [Speeding up Inference](../../docs/sentence_transformer/usage/efficiency.html)
- * [PyTorch](../../docs/sentence_transformer/usage/efficiency.html#pytorch)
- * [ONNX](../../docs/sentence_transformer/usage/efficiency.html#onnx)
- * [OpenVINO](../../docs/sentence_transformer/usage/efficiency.html#openvino)
- * [Benchmarks](../../docs/sentence_transformer/usage/efficiency.html#benchmarks)
- * [Creating Custom Models](../../docs/sentence_transformer/usage/custom_models.html)

* [Structure of Sentence Transformer

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

* [Sentence Transformer Model from a Transformers

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

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

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

* [Semantic Search

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

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

* [MSMARCO Passage

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

* [Multilingual

Models](../../../../docs/sentence_transformer/pretrained_models.html#multilingual-models)

* [Semantic Similarity

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

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

* [Image &

Text-Models](../../../../docs/sentence_transformer/pretrained_models.html#image-text-models)

* [INSTRUCTOR

models](../../../../docs/sentence_transformer/pretrained_models.html#instructor-models)

* [Scientific Similarity

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

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

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

* [Training

- * [Training Examples](../../docs/sentence_transformer/training/examples.html)
- * [Semantic Textual Similarity](../../training/sts/README.html)
- * [Training data](../../training/sts/README.html#training-data)
- * [Loss Function](../../training/sts/README.html#loss-function)
- * [Natural Language Inference](../../training/nli/README.html)
- * [Data](../../training/nli/README.html#data)
- * [SoftmaxLoss](../../training/nli/README.html#softmaxloss)
- * [MultipleNegativesRankingLoss](../../training/nli/README.html#multiplenegativesrankingloss)
- * [Paraphrase Data](../../training/paraphrases/README.html)
- * [Pre-Trained Models](../../training/paraphrases/README.html#pre-trained-models)
- * [Quora Duplicate Questions](../../training/quora_duplicate_questions/README.html)
- * [Training](../../training/quora_duplicate_questions/README.html#training)

*

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

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

- * [Training](../../training/adaptive_layer/README.html#training)
- * [Inference](../../training/adaptive_layer/README.html#inference)
- * [Code Examples](../../training/adaptive_layer/README.html#code-examples)
- * [Multilingual Models](../../training/multilingual/README.html)
 - * [Extend your own models](../../training/multilingual/README.html#extend-your-own-models)
 - * [Training](../../training/multilingual/README.html#training)
 - * [Datasets](../../training/multilingual/README.html#datasets)
 - * [Sources for Training Data](../../training/multilingual/README.html#sources-for-training-data)
 - * [Evaluation](../../training/multilingual/README.html#evaluation)
 - * [Available Pre-trained Models](../../training/multilingual/README.html#available-pre-trained-models)
 - * [Usage](../../training/multilingual/README.html#usage)
 - * [Performance](../../training/multilingual/README.html#performance)
 - * [Citation](../../training/multilingual/README.html#citation)
- * [Model Distillation](../../training/distillation/README.html)
 - * [Knowledge Distillation](../../training/distillation/README.html#knowledge-distillation)
 - * [Speed - Performance Trade-Off](../../training/distillation/README.html#speed-performance-trade-off)
 - * [Dimensionality Reduction](../../training/distillation/README.html#dimensionality-reduction)
 - * [Quantization](../../training/distillation/README.html#quantization)
- * [Augmented SBERT](../../training/data_augmentation/README.html)
 - * [Motivation](../../training/data_augmentation/README.html#motivation)
 - * [Extend to your own datasets](../../training/data_augmentation/README.html#extend-to-your-own-datasets)
 - * [Methodology](../../training/data_augmentation/README.html#methodology)
 - * [Scenario 1: Limited or small annotated datasets (few labeled sentence-pairs)](../../training/data_augmentation/README.html#scenario-1-limited-or-small-annotat

ed-datasets-few-labeled-sentence-pairs)

* [Scenario 2: No annotated datasets (Only unlabeled sentence-pairs)](../../training/data_augmentation/README.html#scenario-2-no-annotated-datasets-only-unlabeled-sentence-pairs)

* [Training](../../training/data_augmentation/README.html#training)

* [Citation](../../training/data_augmentation/README.html#citation)

* [Training with Prompts](../../training/prompts/README.html)

* [What are Prompts?](../../training/prompts/README.html#what-are-prompts)

* [Why would we train with Prompts?](../../training/prompts/README.html#why-would-we-train-with-prompts)

* [How do we train with Prompts?](../../training/prompts/README.html#how-do-we-train-with-prompts)

* [Training with PEFT Adapters](../../training/peft/README.html)

* [Compatibility Methods](../../training/peft/README.html#compatibility-methods)

* [Adding a New Adapter](../../training/peft/README.html#adding-a-new-adapter)

* [Loading a Pretrained Adapter](../../training/peft/README.html#loading-a-pretrained-adapter)

* [Training Script](../../training/peft/README.html#training-script)

* [Unsupervised Learning](../../unsupervised_learning/README.html)

* [TSDAE](../../unsupervised_learning/README.html#tsdae)

* [SimCSE](../../unsupervised_learning/README.html#simcse)

* [CT](../../unsupervised_learning/README.html#ct)

* [CT (In-Batch Negative Sampling)](../../unsupervised_learning/README.html#ct-in-batch-negative-sampling)

* [Masked Language Model (MLM)](../../unsupervised_learning/README.html#masked-language-model-mlm)

* [GenQ](../../unsupervised_learning/README.html#genq)

* [GPL](../../unsupervised_learning/README.html#gpl)

* [Performance Comparison](../../unsupervised_learning/README.html#performance-comparison)

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

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

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

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

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

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

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

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

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

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

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

Cross Encoder

* [Usage](../../docs/cross_encoder/usage/usage.html)

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

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

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

* [Re-Ranker: Cross-Encoder](../retrieve_rerank/README.html#re-ranker-cross-encoder)

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

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

* [Pre-trained Cross-Encoders

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

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

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

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

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

* [Quora Duplicate

Questions](../docs/cross_encoder/pretrained_models.html#quora-duplicate-questions)

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

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

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

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

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

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

* [Cross-Encoder Knowledge

Distillation](../training/ms_marco/cross_encoder_README.html#cross-encoder-knowledge-distillation)

Package Reference

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

*

[SentenceTransformer](../docs/package_reference/sentence_transformer/SentenceTransformer.html)

*

[SentenceTransformer](../docs/package_reference/sentence_transformer/SentenceTransformer.html#id1)

*

[SentenceTransformerModelCardData](../../docs/package_reference/sentence_transformer/SentenceTransformer.html#sentencetransformermodelcarddata)

*

[SimilarityFunction](../../docs/package_reference/sentence_transformer/SentenceTransformer.html#similarityfunction)

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

*

[SentenceTransformerTrainer](../../docs/package_reference/sentence_transformer/trainer.html#sentencetransformertrainer)

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

*

[SentenceTransformerTrainingArguments](../../docs/package_reference/sentence_transformer/training_args.html#sentencetransformertrainingarguments)

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

*

[BatchAllTripletLoss](../../docs/package_reference/sentence_transformer/losses.html#batchalltripletloss)

*

[BatchHardSoftMarginTripletLoss](../../docs/package_reference/sentence_transformer/losses.html#batchhardsoftmargintripletloss)

*

[BatchHardTripletLoss](../../docs/package_reference/sentence_transformer/losses.html#batchhardtripletloss)

*

[BatchSemiHardTripletLoss](../../docs/package_reference/sentence_transformer/losses.html#batchsemihardtripletloss)

*

[ContrastiveLoss](../../docs/package_reference/sentence_transformer/losses.html#contrastiveloss)

*

[OnlineContrastiveLoss](../../docs/package_reference/sentence_transformer/losses.html#onlinecontrastiveloss)

*

[ContrastiveTensionLoss](../../docs/package_reference/sentence_transformer/losses.html#contrastivetensionloss)

*

[ContrastiveTensionLossInBatchNegatives](../../docs/package_reference/sentence_transformer/losses.html#contrastivetensionlossinbatchnegatives)

* [CoSENTLoss](../../docs/package_reference/sentence_transformer/losses.html#cosentloss)

* [AngleLoss](../../docs/package_reference/sentence_transformer/losses.html#angleloss)

*

[CosineSimilarityLoss](../../docs/package_reference/sentence_transformer/losses.html#cosinesimilarityloss)

*

[DenoisingAutoEncoderLoss](../../docs/package_reference/sentence_transformer/losses.html#denoisingautoencoderloss)

*

[GISTEmbedLoss](../../docs/package_reference/sentence_transformer/losses.html#gistembedloss)

*

[CachedGISTEmbedLoss](../../docs/package_reference/sentence_transformer/losses.html#cachedgistembedloss)

* [MSELoss](../../docs/package_reference/sentence_transformer/losses.html#mseloss)

*

[MarginMSELoss](../../docs/package_reference/sentence_transformer/losses.html#marginmseloss)

)

*

[MatryoshkaLoss](../../docs/package_reference/sentence_transformer/losses.html#matryoshkaloss)

)

*

[Matryoshka2dLoss](../../docs/package_reference/sentence_transformer/losses.html#matryoshka2dloss)

*

[AdaptiveLayerLoss](../../docs/package_reference/sentence_transformer/losses.html#adaptivelayerloss)

*

[MegaBatchMarginLoss](../../docs/package_reference/sentence_transformer/losses.html#megabatchmarginloss)

*

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

*

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

*

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

*

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

* [SoftmaxLoss](../../docs/package_reference/sentence_transformer/losses.html#softmaxloss)

* [TripletLoss](../../docs/package_reference/sentence_transformer/losses.html#tripletloss)

* [Samplers](../../docs/package_reference/sentence_transformer/sampler.html)

*

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

*

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

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

*

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

*

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

*

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

*

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

*

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

*

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

*

[RerankingEvaluator](../../../../docs/package_reference/sentence_transformer/evaluation.html#reranking-evaluator)

*

[SentenceEvaluator](../../../../docs/package_reference/sentence_transformer/evaluation.html#sentence-evaluator)

*

[SequentialEvaluator](../../../../docs/package_reference/sentence_transformer/evaluation.html#sequential-evaluator)

*

[TranslationEvaluator](../../../../docs/package_reference/sentence_transformer/evaluation.html#translation-evaluator)

*

[TripletEvaluator](../../../../docs/package_reference/sentence_transformer/evaluation.html#triplet-evaluator)

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

*

[ParallelSentencesDataset](../../../../docs/package_reference/sentence_transformer/datasets.html#parallel-sentences-dataset)

*

[SentenceLabelDataset](../../../../docs/package_reference/sentence_transformer/datasets.html#sentence-label-dataset)

*

[DenoisingAutoEncoderDataset](../../../../docs/package_reference/sentence_transformer/datasets.html#denoising-auto-encoder-dataset)

*

[NoDuplicatesDataLoader](../../../../docs/package_reference/sentence_transformer/datasets.html#no-duplicates-data-loader)

- * [Models](../../docs/package_reference/sentence_transformer/models.html)
 - * [Main Classes](../../docs/package_reference/sentence_transformer/models.html#main-classes)
 - * [Further Classes](../../docs/package_reference/sentence_transformer/models.html#further-classes)
- * [quantization](../../docs/package_reference/sentence_transformer/quantization.html)
 - * [quantize_embeddings()](../../docs/package_reference/sentence_transformer/quantization.html#sentence_transformers.quantization.quantize_embeddings)
 - * [semantic_search_faiss()](../../docs/package_reference/sentence_transformer/quantization.html#sentence_transformers.quantization.semantic_search_faiss)
 - * [semantic_search_usearch()](../../docs/package_reference/sentence_transformer/quantization.html#sentence_transformers.quantization.semantic_search_usearch)
- * [Cross Encoder](../../docs/package_reference/cross_encoder/index.html)
 - * [CrossEncoder](../../docs/package_reference/cross_encoder/cross_encoder.html)
 - * [CrossEncoder](../../docs/package_reference/cross_encoder/cross_encoder.html#id1)
 - * [Training Inputs](../../docs/package_reference/cross_encoder/cross_encoder.html#training-inputs)
 - * [Evaluation](../../docs/package_reference/cross_encoder/evaluation.html)
 - * [CEBinaryAccuracyEvaluator](../../docs/package_reference/cross_encoder/evaluation.html#cebinaryaccuracyevaluator)
 - * [CEBinaryClassificationEvaluator](../../docs/package_reference/cross_encoder/evaluation.html#cebinaryclassificationevaluator)

*

[CECorrelationEvaluator](../../docs/package_reference/cross_encoder/evaluation.html#cecorrelationevaluator)

* [CEF1Evaluator](../../docs/package_reference/cross_encoder/evaluation.html#cef1evaluator)

*

[CESoftmaxAccuracyEvaluator](../../docs/package_reference/cross_encoder/evaluation.html#cesoftmaxaccuracyevaluator)

*

[CERerankingEvaluator](../../docs/package_reference/cross_encoder/evaluation.html#cererankingevaluator)

* [util](../../docs/package_reference/util.html)

* [Helper Functions](../../docs/package_reference/util.html#module-sentence_transformers.util)

*

[`community_detection()`](../../docs/package_reference/util.html#sentence_transformers.util.community_detection)

* [`http_get()`](../../docs/package_reference/util.html#sentence_transformers.util.http_get)

*

[`is_training_available()`](../../docs/package_reference/util.html#sentence_transformers.util.is_training_available)

*

[`mine_hard_negatives()`](../../docs/package_reference/util.html#sentence_transformers.util.mine_hard_negatives)

*

[`normalize_embeddings()`](../../docs/package_reference/util.html#sentence_transformers.util.normalize_embeddings)

*

[`paraphrase_mining()`](../../docs/package_reference/util.html#sentence_transformers.util.paraphrase_mining)

ase_mining)

*

[`semantic_search()](../../docs/package_reference/util.html#sentence_transformers.util.semantic_search)

*

[`truncate_embeddings()](../../docs/package_reference/util.html#sentence_transformers.util.truncate_embeddings)

*

[Model

Optimization](../../docs/package_reference/util.html#module-sentence_transformers.backend)

*

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

*

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

*

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

* [Similarity Metrics](../../docs/package_reference/util.html#module-sentence_transformers.util)

* [`cos_sim()](../../docs/package_reference/util.html#sentence_transformers.util.cos_sim)

* [`dot_score()](../../docs/package_reference/util.html#sentence_transformers.util.dot_score)

*

[`euclidean_sim()](../../docs/package_reference/util.html#sentence_transformers.util.euclidean_sim)

*

[`manhattan_sim()](../../docs/package_reference/util.html#sentence_transformers.util.manhattan_sim)

*

[` pairwise_cos_sim() `](../../docs/package_reference/util.html#sentence_transformers.util.pairwise_cos_sim)

*

[` pairwise_dot_score() `](../../docs/package_reference/util.html#sentence_transformers.util.pairwise_dot_score)

*

[` pairwise_euclidean_sim() `](../../docs/package_reference/util.html#sentence_transformers.util.pairwise_euclidean_sim)

*

[` pairwise_manhattan_sim() `](../../docs/package_reference/util.html#sentence_transformers.util.pairwise_manhattan_sim)

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

* [(../../index.html)

* Cross-Encoders

*

[

Edit

on

GitHub](https://github.com/UKPLab/sentence-transformers/blob/master/examples/applications/cross-encoder/README.md)

* * *

Cross-Encoders¶

SentenceTransformers also supports to load Cross-Encoders for sentence pair scoring and sentence pair classification tasks.

Bi-Encoder vs. Cross-Encoder¶•

First, it is important to understand the difference between Bi- and Cross-Encoder.

Bi-Encoders produce for a given sentence a sentence embedding. We pass to a BERT independently the sentences A and B, which result in the sentence embeddings u and v . These sentence embedding can then be compared using cosine similarity:

![[BiEncoder]](https://raw.githubusercontent.com/UKPLab/sentence-transformers/master/docs/img/Bi_vs_Cross-Encoder.png)

In contrast, for a **Cross-Encoder**, we pass both sentences simultaneously to the Transformer network. It produces then an output value between 0 and 1 indicating the similarity of the input sentence pair:

A **Cross-Encoder** does not produce a sentence embedding. Also, we are not able to pass individual sentences to a Cross-Encoder.

As detailed in our [paper](<https://arxiv.org/abs/1908.10084>), Cross-Encoder achieve better performances than Bi-Encoders. However, for many application they are not practical as they do not produce embeddings we could e.g. index or efficiently compare using cosine similarity.

When to use Cross- / Bi-Encoders?¶•

Cross-Encoders can be used whenever you have a pre-defined set of sentence pairs you want to score. For example, you have 100 sentence pairs and you want to get similarity scores for these 100 pairs.

Bi-Encoders (see [Computing Sentence Embeddings](./computing-embeddings/README.html)) are used whenever you need a sentence embedding in a vector space for efficient comparison. Applications are for example Information Retrieval / Semantic Search or Clustering. Cross-Encoders would be the wrong choice for these application: Clustering 10,000 sentence with CrossEncoders would require computing similarity scores for about 50 Million sentence combinations, which takes about 65 hours. With a Bi-Encoder, you compute the embedding for each sentence, which takes only 5 seconds. You can then perform the clustering.

Cross-Encoders Usage

Using Cross-Encoders is quite easy:

```
from sentence_transformers.cross_encoder import CrossEncoder

model = CrossEncoder("model_name_or_path")

scores = model.predict([["My first", "sentence pair"], ["Second text", "pair"]])
```

You pass to `model.predict`` a list of sentence **pairs**. Note, Cross-Encoder do not work on individual sentence, you have to pass sentence pairs.

As model name, you can pass any model or path that is compatible with Hugging Face `[AutoModel]`(https://huggingface.co/transformers/model_doc/auto.html) class

For a full example, to score a query with all possible sentences in a corpus see `[cross-encoder_usage.py]`(https://github.com/UKPLab/sentence-transformers/tree/master/examples/applications/cross-encoder/cross-encoder_usage.py).

Combining Bi- and Cross-Encoders

Cross-Encoder achieve higher performance than Bi-Encoders, however, they do not scale well for large datasets. Here, it can make sense to combine Cross- and Bi-Encoders, for example in Information Retrieval / Semantic Search scenarios: First, you use an efficient Bi-Encoder to retrieve e.g. the top-100 most similar sentences for a query. Then, you use a Cross-Encoder to re-rank these 100 hits by computing the score for every (query, hit) combination.

For more details on combining Bi- and Cross-Encoders, see `[Application - Information Retrieval]`([../retrieve_rerank/README.html](https://github.com/UKPLab/sentence-transformers/tree/master/examples/applications/cross-encoder/cross-encoder_usage.py)).

Training Cross-Encoders

See `[Cross-Encoder Training]`([../training/cross-encoder/README.html](https://github.com/UKPLab/sentence-transformers/tree/master/examples/applications/cross-encoder/cross-encoder_usage.py)) how to

train your own Cross-Encoder models.

* * *

(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>).