[![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](../sentence_transformer/loss_overview.html#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#multiplenegativesrankingloss)

* [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](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-retrie

val)

* [Pre-trained Cross-Encoders (Re-Ranker)](../../examples/applications/retrieve_rerank/README.html#pre-trained-cross-encoders-re-ranker)
  * [Pretrained Models](pretrained_models.html)
    * [MS MARCO](pretrained_models.html#ms-marco)
    * [SQuAD (QNLI)](pretrained_models.html#squad-qnli)
    * [STSbenchmark](pretrained_models.html#stsbenchmark)
    * [Quora Duplicate Questions](pretrained_models.html#quora-duplicate-questions)
    * [NLI](pretrained_models.html#nli)
    * [Community Models](pretrained_models.html#community-models)
  * Training Overview
  * [Training Examples](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](../package_reference/sentence_transformer/index.html)
    * [SentenceTransformer](../package_reference/sentence_transformer/SentenceTransformer.html)

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

*

[SentenceTransformerModelCardData](../package_reference/sentence_transformer/SentenceTransformer.html#sentencetransformermodelcarddata)
*
[SimilarityFunction](../package_reference/sentence_transformer/SentenceTransformer.html#similarityfunction)
  * [Trainer](../package_reference/sentence_transformer/trainer.html)
*
[SentenceTransformerTrainer](../package_reference/sentence_transformer/trainer.html#sentencetransformertrainer)
  * [Training Arguments](../package_reference/sentence_transformer/training_args.html)
*
[SentenceTransformerTrainingArguments](../package_reference/sentence_transformer/training_args.html#sentencetransformertrainingarguments)
  * [Losses](../package_reference/sentence_transformer/losses.html)
*
[BatchAllTripletLoss](../package_reference/sentence_transformer/losses.html#batchalltripletloss)
*
[BatchHardSoftMarginTripletLoss](../package_reference/sentence_transformer/losses.html#batchhardsoftmargintripletloss)
*
[BatchHardTripletLoss](../package_reference/sentence_transformer/losses.html#batchhardtripletloss)
*
[BatchSemiHardTripletLoss](../package_reference/sentence_transformer/losses.html#batchsemihardtripletloss)
    * [ContrastiveLoss](../package_reference/sentence_transformer/losses.html#contrastiveloss)
*

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

*

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

*

[ContrastiveTensionLossInBatchNegatives](../package_reference/sentence_transformer/losses.html#contrastivetensionlossinbatchnegatives)
    * [CoSENTLoss](../package_reference/sentence_transformer/losses.html#cosentloss)
    * [AnglELoss](../package_reference/sentence_transformer/losses.html#angleloss)

*

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

*

[DenoisingAutoEncoderLoss](../package_reference/sentence_transformer/losses.html#denoisingautoencoderloss)
    * [GISTEmbedLoss](../package_reference/sentence_transformer/losses.html#gistembedloss)

*

[CachedGISTEmbedLoss](../package_reference/sentence_transformer/losses.html#cachedgistembedloss)
    * [MSELoss](../package_reference/sentence_transformer/losses.html#mseloss)
    * [MarginMSELoss](../package_reference/sentence_transformer/losses.html#marginmseloss)
    * [MatryoshkaLoss](../package_reference/sentence_transformer/losses.html#matryoshkaloss)

*

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

*

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

*

[MegaBatchMarginLoss](../package_reference/sentence_transformer/losses.html#megabatchmarginloss)
*
[MultipleNegativesRankingLoss](../package_reference/sentence_transformer/losses.html#multiplenegativesrankingloss)
*
[CachedMultipleNegativesRankingLoss](../package_reference/sentence_transformer/losses.html#cachedmultiplenegativesrankingloss)
*
[MultipleNegativesSymmetricRankingLoss](../package_reference/sentence_transformer/losses.html#multiplenegativessymmetricrankingloss)
*
[CachedMultipleNegativesSymmetricRankingLoss](../package_reference/sentence_transformer/losses.html#cachedmultiplenegativessymmetricrankingloss)
    * [SoftmaxLoss](../package_reference/sentence_transformer/losses.html#softmaxloss)
    * [TripletLoss](../package_reference/sentence_transformer/losses.html#tripletloss)
  * [Samplers](../package_reference/sentence_transformer/sampler.html)
    * [BatchSamplers](../package_reference/sentence_transformer/sampler.html#batchsamplers)
*
[MultiDatasetBatchSamplers](../package_reference/sentence_transformer/sampler.html#multidatasetbatchsamplers)
  * [Evaluation](../package_reference/sentence_transformer/evaluation.html)
*
[BinaryClassificationEvaluator](../package_reference/sentence_transformer/evaluation.html#binaryclassificationevaluator)
*
[EmbeddingSimilarityEvaluator](../package_reference/sentence_transformer/evaluation.html#embed

dingsimilarityevaluator)

*

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

*

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

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

*

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

*

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

*

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

*

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

*

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

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

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

*

[ParallelSentencesDataset](../package_reference/sentence_transformer/datasets.html#parallelsente

ncesdataset)

*

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

*

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

*

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

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

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

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

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

*

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

*

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

*

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

* [Cross Encoder](../package_reference/cross_encoder/index.html)

* [CrossEncoder](../package_reference/cross_encoder/cross_encoder.html)

* [CrossEncoder](../package_reference/cross_encoder/cross_encoder.html#id1)

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

    * [Evaluation](../package_reference/cross_encoder/evaluation.html)
      * [CEBinaryAccuracyEvaluator](../package_reference/cross_encoder/evaluation.html#cebinaryaccuracyevaluator)
      * [CEBinaryClassificationEvaluator](../package_reference/cross_encoder/evaluation.html#cebinaryclassificationevaluator)
      * [CECorrelationEvaluator](../package_reference/cross_encoder/evaluation.html#cecorrelationevaluator)
        * [CEF1Evaluator](../package_reference/cross_encoder/evaluation.html#cef1evaluator)
      * [CESoftmaxAccuracyEvaluator](../package_reference/cross_encoder/evaluation.html#cesoftmaxaccuracyevaluator)
      * [CERerankingEvaluator](../package_reference/cross_encoder/evaluation.html#cererankingevaluator)
  * [util](../package_reference/util.html)
    * [Helper Functions](../package_reference/util.html#module-sentence_transformers.util)
      * [`community_detection()`](../package_reference/util.html#sentence_transformers.util.community_detection)
      * [`http_get()`](../package_reference/util.html#sentence_transformers.util.http_get)
      * [`is_training_available()`](../package_reference/util.html#sentence_transformers.util.is_training_available)
      *

[`mine_hard_negatives()`](../package_reference/util.html#sentence_transformers.util.mine_hard_negatives)
*
[`normalize_embeddings()`](../package_reference/util.html#sentence_transformers.util.normalize_embeddings)
*
[`paraphrase_mining()`](../package_reference/util.html#sentence_transformers.util.paraphrase_mining)
*
[`semantic_search()`](../package_reference/util.html#sentence_transformers.util.semantic_search)
*
[`truncate_embeddings()`](../package_reference/util.html#sentence_transformers.util.truncate_embeddings)
* [Model Optimization](../package_reference/util.html#module-sentence_transformers.backend)
*
[`export_dynamic_quantized_onnx_model()`](../package_reference/util.html#sentence_transformers.backend.export_dynamic_quantized_onnx_model)
*
[`export_optimized_onnx_model()`](../package_reference/util.html#sentence_transformers.backend.export_optimized_onnx_model)
*
[`export_static_quantized_openvino_model()`](../package_reference/util.html#sentence_transformers.backend.export_static_quantized_openvino_model)
* [Similarity Metrics](../package_reference/util.html#module-sentence_transformers.util)
* [`cos_sim()`](../package_reference/util.html#sentence_transformers.util.cos_sim)
* [`dot_score()`](../package_reference/util.html#sentence_transformers.util.dot_score)
* [`euclidean_sim()`](../package_reference/util.html#sentence_transformers.util.euclidean_sim)

* [`manhattan_sim()`](../package_reference/util.html#sentence_transformers.util.manhattan_sim)
* [`pairwise_cos_sim()`](../package_reference/util.html#sentence_transformers.util.pairwise_cos_sim)
* [`pairwise_dot_score()`](../package_reference/util.html#sentence_transformers.util.pairwise_dot_score)
* [`pairwise_euclidean_sim()`](../package_reference/util.html#sentence_transformers.util.pairwise_euclidean_sim)
* [`pairwise_manhattan_sim()`](../package_reference/util.html#sentence_transformers.util.pairwise_manhattan_sim)

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

* [](../../index.html)
* Training Overview
* [Edit on GitHub](https://github.com/UKPLab/sentence-transformers/blob/master/docs/cross_encoder/training_overview.md)

* * *

Training Overviewï

Note

The CrossEncoder training approach has not been updated in v3.0 when [training Sentence Transformer models](../sentence_transformer/training_overview.html) was improved. Improving training CrossEncoders is planned for a future major update.

The `CrossEncoder` class is a wrapper around Hugging Face `AutoModelForSequenceClassification`, but with some methods to make training and predicting scores a little bit easier. The saved models are 100% compatible with Hugging Face and can also be loaded with their classes.

First, you need some sentence pair data. You can either have a continuous score, like:

Documentation

*

[`InputExample`](../package_reference/cross_encoder/cross_encoder.html#sentence_transformers.readers.InputExample "sentence_transformers.readers.InputExample")

```
from sentence_transformers import InputExample

train_samples = [
    InputExample(texts=["sentence1", "sentence2"], label=0.3),
    InputExample(texts=["Another", "pair"], label=0.8),
]
```

Or you have distinct classes as in the [training_nli.py](https://github.com/UKPLab/sentence-transformers/tree/master/docs/cross_encoder/../../examples/training/cross-encoder/training_nli.py) example:

```
from sentence_transformers import InputExample

label2int = {"contradiction": 0, "entailment": 1, "neutral": 2}
train_samples = [
    InputExample(texts=["sentence1", "sentence2"], label=label2int["neutral"]),
    InputExample(texts=["Another", "pair"], label=label2int["entailment"]),
]
```

Then, you define the base model and the number of labels. You can take any [Hugging Face pre-trained model](https://huggingface.co/models) that is compatible with AutoModel:

```
model = CrossEncoder('distilroberta-base', num_labels=1)
```

For binary tasks and tasks with continuous scores (like STS), we set
num_labels=1. For classification tasks, we set it to the number of labels we
have.

We start the training by calling
[`CrossEncoder.fit`](../package_reference/cross_encoder/cross_encoder.html#sentence_transforme
rs.cross_encoder.CrossEncoder.fit
"sentence_transformers.cross_encoder.CrossEncoder.fit"):

Documentation

*
[`CrossEncoder`](../package_reference/cross_encoder/cross_encoder.html#sentence_transformers.
cross_encoder.CrossEncoder "sentence_transformers.cross_encoder.CrossEncoder")

*
[`CrossEncoder.fit`](../package_reference/cross_encoder/cross_encoder.html#sentence_transforme
rs.cross_encoder.CrossEncoder.fit "sentence_transformers.cross_encoder.CrossEncoder.fit")

```
    model.fit(
        train_dataloader=train_dataloader,
        evaluator=evaluator,
        epochs=num_epochs,
        warmup_steps=warmup_steps,
        output_path=model_save_path,
```

)

[Previous](pretrained_models.html "Pretrained Models") [Next](training/examples.html "Training Examples")

* * *