```
[![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](../../applications/computing-embeddings/README.html)
 - * [Initializing a Sentence Transformer

Model](../../applications/computing-embeddings/README.html#initializing-a-sentence-transformer-model)

[Calculating

Embeddings](../../applications/computing-embeddings/README.html#calculating-embeddings)

- * [Prompt Templates](../../applications/computing-embeddings/README.html#prompt-templates)
- * [Input Sequence Length](../../applications/computing-embeddings/README.html#id1)

| | * | [Multi-Proc | ess | / | Multi-GPU |
|--|---------------|--------------|--------------|-------------|---------------|
| Encoding](//applications/computing-embedd | lings/READM | 1E.html#m | ulti-proces | ss-multi-g | pu-encodin |
| g) | | | | | |
| | * | | [Semantic | ; | Textual |
| Similarity](//docs/sentence_transformer/us | sage/semanti | c_textual_ | similarity.l | html) | |
| | | | * | | [Similarity |
| Calculation](//docs/sentence_transformer/ | 'usage/sema | ntic_textua | l_similarit | y.html#si | milarity-calc |
| ulation) | | | | | |
| * [Semantic Search](//applications/seman | tic-search/Rl | EADME.ht | ml) | | |
| * [Background](//applications/semantic-s | search/REAC | ME.html#l | oackgroun | ıd) | |
| * [| [Symmetric | VS. | Asymr | netric | Semantic |
| Search](//applications/semantic-search/REA | NDME.html#s | symmetric- | vs-asymm | etric-sem | nantic-searc |
| h) | | | | | |
| | | | * | · | [Manual |
| Implementation](//applications/semantic-sea | arch/READM | E.html#ma | nual-imple | ementatic | on) |
| | | | * | | [Optimized |
| Implementation](//applications/semantic-sea | arch/READM | E.html#opt | imized-im | plementa | ition) |
| * [Speed Optimization](//applications/ser | mantic-searc | h/READMI | E.html#sp∈ | eed-optin | nization) |
| * [Elasticsearch](//applications/semantic | -search/REA | DME.html | #elasticse | arch) | |
| | | * | [Approxim | nate | Nearest |
| Neighbor](//applications/semantic-search/R | EADME.html | #approxim | ate-neare | st-neighb | or) |
| * [Retrieve & Re-Rank](//applications/se | mantic-searc | :h/READM | E.html#re | trieve-re- | rank) |
| * [Examples](//applications/semantic-sea | arch/READM | E.html#ex | amples) | | |
| * [Retrieve & Re-Rank](//applications/retri | eve_rerank/l | README. | ntml) | | |
| | * | [Retrie | eve | & | Re-Rank |
| Pipeline](//applications/retrieve_rerank/REA | DME.html#re | etrieve-re-r | ank-pipeli | ne) | |
| * [Retrieval: Bi-Encoder](//applications/re | etrieve_reran | k/READM | E.html#ret | trieval-bi- | encoder) |

| [Re-Ranker: | |
|-------------|--|
| ·) | |

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

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

[Pre-trained Bi-Encoders

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

[Pre-trained Cross-Encoders

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

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

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

* [Translated Sentence Mining](../../applications/parallel-sentence-mining/README.html)

* [Margin Based

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

- * [Examples](../../applications/parallel-sentence-mining/README.html#examples)
- * [Image Search](../../applications/image-search/README.html)
 - * [Installation](../../applications/image-search/README.html#installation)
 - * [Usage](../../applications/image-search/README.html#usage)
 - * [Examples](../../applications/image-search/README.html#examples)
- * [Embedding Quantization](../../applications/embedding-quantization/README.html)

[Binary

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

(int8) [Scalar Quantization](../../applications/embedding-quantization/README.html#scalar-int8-quantization) [Additional extensions](../../applications/embedding-quantization/README.html#additional-extensions) * [Demo](../../applications/embedding-quantization/README.html#demo) * [Try it yourself](../../applications/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-transfo rmer-models) Transformer [Sentence Model from **Transformers** Model](../../docs/sentence_transformer/usage/custom_models.html#sentence-transformer-model-f rom-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-ga-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
Components](../../docs/sentence_transformer/training_overview.html#training-components)
  * [Dataset](../../docs/sentence transformer/training overview.html#dataset)
   * [Dataset Format](../../docs/sentence_transformer/training_overview.html#dataset-format)
  * [Loss Function](../../docs/sentence_transformer/training_overview.html#loss-function)
                                                                                         [Training
Arguments](../../docs/sentence_transformer/training_overview.html#training-arguments)
  * [Evaluator](../../docs/sentence transformer/training overview.html#evaluator)
  * [Trainer](../../docs/sentence transformer/training overview.html#trainer)
   * [Callbacks](../../docs/sentence_transformer/training_overview.html#callbacks)
                                                                                    [Multi-Dataset
Training](../../docs/sentence_transformer/training_overview.html#multi-dataset-training)
                                                                                      [Deprecated
Training](../../docs/sentence_transformer/training_overview.html#deprecated-training)
                                                       [Best
                                                                       Base
                                                                                       Embedding
Models](../../docs/sentence_transformer/training_overview.html#best-base-embedding-models)
```

* [Dataset Overview](../../docs/sentence_transformer/dataset_overview.html) [Datasets on the Hugging Face Hub](../../docs/sentence transformer/dataset overview.html#datasets-on-the-hugging-face-hub) [Pre-existing Datasets](../../docs/sentence_transformer/dataset_overview.html#pre-existing-datasets) * [Loss Overview](../../docs/sentence_transformer/loss_overview.html) * [Loss modifiers](../../docs/sentence_transformer/loss_overview.html#loss-modifiers) * [Distillation](../../docs/sentence_transformer/loss_overview.html#distillation) [Commonly used Loss Functions](../../docs/sentence_transformer/loss_overview.html#commonly-used-loss-functions) [Custom Loss Functions](../../docs/sentence_transformer/loss_overview.html#custom-loss-functions) * [Training Examples](../../docs/sentence_transformer/training/examples.html) * [Semantic Textual Similarity](../sts/README.html) * [Training data](../sts/README.html#training-data) * [Loss Function](../sts/README.html#loss-function) * [Natural Language Inference](../nli/README.html) * [Data](../nli/README.html#data) * [SoftmaxLoss](../nli/README.html#softmaxloss) * [MultipleNegativesRankingLoss](../nli/README.html#multiplenegativesrankingloss) * [Paraphrase Data](../paraphrases/README.html) * [Pre-Trained Models](../paraphrases/README.html#pre-trained-models) * Quora Duplicate Questions * Training * MultipleNegativesRankingLoss * Pretrained Models

* [MS MARCO](../ms_marco/README.html)

- * [Bi-Encoder](../ms marco/README.html#bi-encoder)
- * [Matryoshka Embeddings](../matryoshka/README.html)
 - * [Use Cases](../matryoshka/README.html#use-cases)
 - * [Results](../matryoshka/README.html#results)
 - * [Training](../matryoshka/README.html#training)
 - * [Inference](../matryoshka/README.html#inference)
- * [Code Examples](../matryoshka/README.html#code-examples)
- * [Adaptive Layers](../adaptive_layer/README.html)
 - * [Use Cases](../adaptive layer/README.html#use-cases)
 - * [Results](../adaptive layer/README.html#results)
 - * [Training](../adaptive_layer/README.html#training)
 - * [Inference](../adaptive_layer/README.html#inference)
 - * [Code Examples](../adaptive_layer/README.html#code-examples)
- * [Multilingual Models](../multilingual/README.html)
 - * [Extend your own models](../multilingual/README.html#extend-your-own-models)
 - * [Training](../multilingual/README.html#training)
 - * [Datasets](../multilingual/README.html#datasets)
 - * [Sources for Training Data](../multilingual/README.html#sources-for-training-data)
 - * [Evaluation](../multilingual/README.html#evaluation)
 - * [Available Pre-trained Models](../multilingual/README.html#available-pre-trained-models)
 - * [Usage](../multilingual/README.html#usage)
 - * [Performance](../multilingual/README.html#performance)
 - * [Citation](../multilingual/README.html#citation)
- * [Model Distillation](../distillation/README.html)
 - * [Knowledge Distillation](../distillation/README.html#knowledge-distillation)
 - * [Speed Performance Trade-Off](../distillation/README.html#speed-performance-trade-off)
 - * [Dimensionality Reduction](../distillation/README.html#dimensionality-reduction)

- * [Quantization](../distillation/README.html#quantization)
- * [Augmented SBERT](../data_augmentation/README.html)
 - * [Motivation](../data_augmentation/README.html#motivation)
- * [Extend to your own datasets](../data_augmentation/README.html#extend-to-your-own-datasets)
 - * [Methodology](../data_augmentation/README.html#methodology)
- * [Scenario 1: Limited or small annotated datasets (few labeled sentence-pairs)](../data_augmentation/README.html#scenario-1-limited-or-small-annotated-dataset s-few-labeled-sentence-pairs)
- * [Scenario 2: No annotated datasets (Only unlabeled sentence-pairs)](../data_augmentation/README.html#scenario-2-no-annotated-datasets-only-unlab eled-sentence-pairs)
 - * [Training](../data_augmentation/README.html#training)
 - * [Citation](../data augmentation/README.html#citation)
 - * [Training with Prompts](../prompts/README.html)
 - * [What are Prompts?](../prompts/README.html#what-are-prompts)
- * [Why would we train with

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

- * [How do we train with Prompts?](../prompts/README.html#how-do-we-train-with-prompts)
- * [Training with PEFT Adapters](../peft/README.html)
 - * [Compatibility Methods](../peft/README.html#compatibility-methods)
 - * [Adding a New Adapter](../peft/README.html#adding-a-new-adapter)
 - * [Loading a Pretrained Adapter](../peft/README.html#loading-a-pretrained-adapter)
 - * [Training Script](../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/REA[| OME.html#ct) | | | |
|------|--------------------------------------|-----------------|------------------|----------------|--------------|
| | | * | [CT | (In-Batch | Negative |
| Sam | oling)](//unsupervised_learning/REA | ADME.html#ct- | -in-batch-nega | tive-sampling) |) |
| | | * | [Masked | Languag | e Mode |
| (MLN | l)](//unsupervised_learning/READM | IE.html#maske | ed-language-m | odel-mlm) | |
| * | [GenQ](//unsupervised_learning/RE | EADME.html# | genq) | | |
| * | [GPL](//unsupervised_learning/REA | ADME.html#gp | ol) | | |
| | | | | * | [Performance |
| Com | parison](//unsupervised_learning/RI | EADME.html# | performance-c | omparison) | |
| * [| Domain Adaptation](//domain_adap | tation/READM | IE.html) | | |
| | * | [Domain | Adaptation | VS. | Unsupervised |
| Lear | ning](//domain_adaptation/README | E.html#domair | n-adaptation-vs | -unsupervise | d-learning) |
| * | [Adaptive Pre-Training](//domain_a | daptation/RE/ | ADME.html#ad | aptive-pre-tra | ining) |
| | | | * | [GPL: | Generative |
| Pseu | do-Labeling](//domain_adaptation/F | README.html | #gpl-generativ | e-pseudo-labe | eling) |
| * [| Hyperparameter Optimization](/hpo/F | README.html |) | | |
| * | [HPO Components](/hpo/README.h | ntml#hpo-com | ponents) | | |
| * | [Putting It All Together](/hpo/READM | 1E.html#puttin | g-it-all-togethe | r) | |
| * | [Example Scripts](/hpo/README.htr | ml#example-so | cripts) | | |
| * [| Distributed Training](//docs/senter | nce_transform | er/training/dist | ributed.html) | |
| * | [Comparison](//docs/sentence_tra | ansformer/traiı | ning/distributed | d.html#compa | rison) |
| * | [FSDP](//docs/sentence_transform | mer/training/di | stributed.html# | fsdp) | |
| | | | | | |
| Cros | s Encoder | | | | |
| | | | | | |
| * [U | sage](//docs/cross_encoder/usag | e/usage.html) | | | |

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

[Retrieve Re-Rank & Pipeline](../../applications/retrieve_rerank/README.html#retrieve-re-rank-pipeline) * [Retrieval: Bi-Encoder](../../applications/retrieve rerank/README.html#retrieval-bi-encoder) [Re-Ranker: Cross-Encoder](../../applications/retrieve_rerank/README.html#re-ranker-cross-encoder) * [Example Scripts](../../applications/retrieve_rerank/README.html#example-scripts) [Pre-trained **Bi-Encoders** (Retrieval)](../../applications/retrieve_rerank/README.html#pre-trained-bi-encoders-retrieval) [Pre-trained Cross-Encoders (Re-Ranker)](../../applications/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](../ms_marco/cross_encoder_README.html)

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

Package Reference

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

[Cross-Encoder

Knowledge

* [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/Sente nceTransformer.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#se ntencetransformertrainer) * [Training Arguments](../../docs/package_reference/sentence_transformer/training_args.html) [SentenceTransformerTrainingArguments](../../docs/package_reference/sentence_transformer/trainingArguments] ning_args.html#sentencetransformertrainingarguments) * [Losses](../../../docs/package_reference/sentence_transformer/losses.html) [BatchAllTripletLoss](../../docs/package_reference/sentence_transformer/losses.html#batchalltriple tloss) [BatchHardSoftMarginTripletLoss](../../../docs/package_reference/sentence_transformer/losses.html

#batchhardsoftmargintripletloss)

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

tripletloss)

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

*

[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#contras tivetensionloss)

*

[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#cache
dgistembedloss)
   * [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#matryoshka2
dloss)
[AdaptiveLayerLoss](../../docs/package_reference/sentence_transformer/losses.html#adaptivelaye
rloss)
[MegaBatchMarginLoss](../../docs/package_reference/sentence_transformer/losses.html#megabat
chmarginloss)
[MultipleNegativesRankingLoss](../../docs/package_reference/sentence_transformer/losses.html#
multiplenegativesrankingloss)
[CachedMultipleNegativesRankingLoss](../../docs/package_reference/sentence_transformer/losse
s.html#cachedmultiplenegativesrankingloss)
```

*

[MultipleNegativesSymmetricRankingLoss](../../docs/package_reference/sentence_transformer/los ses.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)

ŧ

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

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

t

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

ŀ

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

t

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

r

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

| irevaluator) |
|--|
| [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#rerankingevaluator) |
| [SentenceEvaluator](//docs/package_reference/sentence_transformer/evaluation.html#sentenceevaluator) |
| [SequentialEvaluator](//docs/package_reference/sentence_transformer/evaluation.html#sequen tialevaluator) |
| [TranslationEvaluator](//docs/package_reference/sentence_transformer/evaluation.html#translationevaluator) |
| [TripletEvaluator](//docs/package_reference/sentence_transformer/evaluation.html#tripletevaluation) * [Datasets](//docs/package_reference/sentence_transformer/datasets.html) |

[ParallelSentencesDataset](../../docs/package_reference/sentence_transformer/datasets.html#par allelsentencesdataset)

*

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

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

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

* [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#s entence_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.ht ml#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#cebina ryaccuracyevaluator) [CEBinaryClassificationEvaluator](../../docs/package_reference/cross_encoder/evaluation.html#ce binaryclassificationevaluator) [CECorrelationEvaluator](../../docs/package_reference/cross_encoder/evaluation.html#cecorrelatio nevaluator) * [CEF1Evaluator](../../docs/package_reference/cross_encoder/evaluation.html#cef1evaluator) [CESoftmaxAccuracyEvaluator](../../docs/package_reference/cross_encoder/evaluation.html#ceso ftmaxaccuracyevaluator) [CERerankingEvaluator](../../docs/package_reference/cross_encoder/evaluation.html#cereranking evaluator) * [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.comm

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

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

unity_detection)

ing_available)

*

[Model

[`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)

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

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

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

[`export_dynamic_quantized_onnx_model()`](../../docs/package_reference/util.html#sentence_tran sformers.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_tra nsformers.backend.export_static_quantized_openvino_model)

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

| * [`cos_sim()`](//do | ocs/package_referen | ce/util.html#sente | nce_transformers.util.c | os_sim) |
|---|----------------------|---------------------|-------------------------|---------------------|
| * [`dot_score()`](// | docs/package_refere | nce/util.html#sent | tence_transformers.util | .dot_score) |
| [`euclidean_sim()`](//d | locs/package_referer | nce/util.html#sente | ence_transformers.util. | euclidean_si |
| [`manhattan_sim()`](/// sim) | docs/package_refere | ence/util.html#sen | tence_transformers.util | .manhattan_ |
| [`pairwise_cos_sim()`](/ cos_sim) | //docs/package_refe | erence/util.html#s | entence_transformers.u | util.pairwise_ * |
| [`pairwise_dot_score()`](/ _dot_score) | ///docs/package_re | eference/util.html# | sentence_transformers | s.util.pairwise |
| [`pairwise_euclidean_sim(wise_euclidean_sim) |)`](//docs/packaç | ge_reference/util.h | ntml#sentence_transfor | mers.util.pair |
| [`pairwise_manhattan_sim rwise_manhattan_sim) | i()`](//docs/packa | ge_reference/util. | .html#sentence_transfo | |
| [Sentence Transformers | s](//index.html) | | | |
| * [](//index.html) * [Training Examples](/. | .//docs/sentence_tra | ansformer/training | ı/examples.html) | |
| * Quora Duplicate Questi | | _ | | |
| · | * | Γ | Edit | on |

GitHub](https://github.com/UKPLab/sentence-transformers/blob/master/examples/training/quora_duplicate_questions/README.md)

* * *

Quora Duplicate Questionsïf•

This folder contains scripts that demonstrate how to train

SentenceTransformers for **Information Retrieval**. As a simple example, we will use the [Quora Duplicate Questions

dataset](https://huggingface.co/datasets/sentence-transformers/quora-duplicates). It contains over 500,000 sentences with over 400,000 pairwise annotations whether two questions are a duplicate or not.

Models trained on this dataset can be used for mining duplicate questions, i.e., given a large set of sentences (in this case questions), identify all pairs that are duplicates. See [Paraphrase Mining](../../applications/paraphrase-mining/README.html) for an example how to use sentence transformers to mine for duplicate questions / paraphrases. This approach can be scaled to hundred thousands of sentences.

Trainingïf•

Choosing the right loss function is crucial for finetuning useful models. For the given task, two loss functions are especially suitable:

[`OnlineContrastiveLoss`](../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.OnlineContrastiveLoss

"sentence_transformers.losses.OnlineContrastiveLoss") and

 $[`MultipleNegativesRankingLoss`] (.../.../docs/package_reference/sentence_transformer/losses.html) and the properties of the properties$

#sentence_transformers.losses.MultipleNegativesRankingLoss

"sentence_transformers.losses.MultipleNegativesRankingLoss").

Contrastive Lossïf•

For the complete training example, see

[training OnlineContrastiveLoss.py](https://github.com/UKPLab/sentence-

 $transformers/tree/master/examples/training/quora_duplicate_questions/training_OnlineContrastiveL\\$

oss.py).

The Quora Duplicates dataset has a [pair-class

subset](https://huggingface.co/datasets/sentence-transformers/quora-

duplicates/viewer/pair-class) which consists of question pairs and labels: 1

for duplicate and 0 for different.

As shown by our [Loss

Overview](../../docs/sentence_transformer/loss_overview.md), this allows us

to use

[`ContrastiveLoss`](../../docs/package_reference/sentence_transformer/losses.html#sentence_transfor

sformers.losses.ContrastiveLoss

"sentence_transformers.losses.ContrastiveLoss"). Similar pairs with label 1

are pulled together, so that they are close in vector space, while dissimilar

pairs that are closer than a defined margin are pushed away in vector space.

An improved version is

[`OnlineContrastiveLoss`](../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.OnlineContrastiveLoss

"sentence_transformers.losses.OnlineContrastiveLoss"). This loss looks which negative pairs have a lower distance than the largest positive pair and which positive pairs have a higher distance than the lowest distance of negative pairs. I.e., this loss automatically detects the hard cases in a batch and computes the loss only for these cases.

The loss can be used like this:

from datasets import load_dataset

```
train_dataset = load_dataset("sentence-transformers/quora-duplicates", "pair-class", split="train")
# => Dataset({
# features: ['sentence1', 'sentence2', 'label'],
# num_rows: 404290
# })
print(train_dataset[0])
# => {'sentence1': 'What is the step by step guide to invest in share market in india?', 'sentence2':
'What is the step by step guide to invest in share market?', 'label': 0}
train_loss = losses.OnlineContrastiveLoss(model=model, margin=0.5)
```

MultipleNegativesRankingLossïf•

```
For the complete example, see
```

[training_MultipleNegativesRankingLoss.py](https://github.com/UKPLab/sentence-transformers/tree/master/examples/training/quora_duplicate_questions/training_MultipleNegativesRankingLoss.py).

[`MultipleNegativesRankingLoss`](../../../docs/package_reference/sentence_transformer/losses.html #sentence_transformers.losses.MultipleNegativesRankingLoss
"sentence_transformers.losses.MultipleNegativesRankingLoss") is especially
suitable for Information Retrieval / Semantic Search. A nice advantage is that
it only requires positive pairs, i.e., we only need examples of duplicate
questions. See [NLI >
MultipleNegativesRankingLoss](../nli/README.html#multiplenegativesrankingloss)
for more information on how the loss works.

Using the loss is easy and does not require tuning of any hyperparameters:

from datasets import load dataset

```
train_dataset = load_dataset("sentence-transformers/quora-duplicates", "pair", split="train")
# => Dataset({
# features: ['anchor', 'positive'],
# num_rows: 149263
# })
print(train_dataset[0])
# => {'anchor': 'Astrology: I am a Capricorn Sun Cap moon and cap rising...what does that say
```

```
about me?', 'positive': "I'm a triple Capricorn (Sun, Moon and ascendant in Capricorn) What does this say about me?"}

train_loss = losses.MultipleNegativesRankingLoss(model)
```

As 'is_duplicate' is a symmetric relation, we can use not just (anchor, positive) but also (positive, anchor) to our training sample set:

```
from datasets import concatenate_datasets
```

```
train_dataset = concatenate_datasets([
    train_dataset,
    train_dataset.rename_columns({"anchor": "positive", "positive": "anchor"})
])
# Dataset({
# features: ['anchor', 'positive'],
# num_rows: 298526
# })
```

Note

Increasing the batch sizes usually yields better results, as the task gets harder. It is more difficult to identify the correct duplicate question out of a set of 100 questions than out of a set of only 10 questions. So it is

advisable to set the training batch size as large as possible. I trained it with a batch size of 350 on 32 GB GPU memory.

Note

[`MultipleNegativesRankingLoss`](../../.docs/package_reference/sentence_transformer/losses.html #sentence_transformers.losses.MultipleNegativesRankingLoss
"sentence_transformers.losses.MultipleNegativesRankingLoss") only works if
_(a_i, b_j)_ with j != i is actually a negative, non-duplicate question pair.

In few instances, this assumption is wrong. But in the majority of cases, if
we sample two random questions, they are not duplicates. If your dataset
cannot fulfil this property,

[`MultipleNegativesRankingLoss`](../../../docs/package_reference/sentence_transformer/losses.html #sentence_transformers.losses.MultipleNegativesRankingLoss

"sentence_transformers.losses.MultipleNegativesRankingLoss") might not work

well.

Multi-Task-Learningif •

[`ContrastiveLoss`](../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.ContrastiveLoss

"sentence_transformers.losses.ContrastiveLoss") works well for pair classification, i.e., given two pairs, are these duplicates or not. It pushes negative pairs far away in vector space, so that the distinguishing between duplicate and non-duplicate pairs works good.

[`MultipleNegativesRankingLoss`](../../docs/package_reference/sentence_transformer/losses.html

#sentence_transformers.losses.MultipleNegativesRankingLoss

"sentence_transformers.losses.MultipleNegativesRankingLoss") on the other sides mainly reduces the distance between positive pairs out of large set of possible candidates. However, the distance between non-duplicate questions is not so large, so that this loss does not work that well for pair classification.

In [training_multi-task-learning.py](https://github.com/UKPLab/sentence-transformers/tree/master/examples/training/quora_duplicate_questions/training_multi-task-learning.py) I demonstrate how we can train the network with both losses.

The essential code is to define both losses and to pass it to the fit method.

```
from datasets import load_dataset

from sentence_transformers.losses import ContrastiveLoss, MultipleNegativesRankingLoss

from sentence_transformers import SentenceTransformerTrainer, SentenceTransformer

model_name = "stsb-distilbert-base"

model = SentenceTransformer(model_name)

# https://huggingface.co/datasets/sentence-transformers/quora-duplicates

mnrl_dataset = load_dataset(
    "sentence-transformers/quora-duplicates", "triplet", split="train"

) # The "pair" subset also works

mnrl_train_dataset = mnrl_dataset.select(range(100000))
```

mnrl_eval_dataset = mnrl_dataset.select(range(100000, 101000))

```
# https://huggingface.co/datasets/sentence-transformers/quora-duplicates
cl_dataset = load_dataset("sentence-transformers/quora-duplicates", "pair-class", split="train")
cl_train_dataset = cl_dataset.select(range(100000))
cl_eval_dataset = cl_dataset.select(range(100000, 101000))
cl_train_loss = ContrastiveLoss(model=model, margin=0.5)
# Create the trainer & start training
trainer = SentenceTransformerTrainer(
  model=model,
  train_dataset={
     "mnrl": mnrl_train_dataset,
     "cl": cl_train_dataset,
  },
  eval_dataset={
     "mnrl": mnrl eval dataset,
     "cl": cl_eval_dataset,
  },
  loss={
     "mnrl": mnrl_train_loss,
     "cl": cl_train_loss,
  },
)
trainer.train()
```

mnrl_train_loss = MultipleNegativesRankingLoss(model=model)

Pretrained Modelsif•

Currently the following models trained on Quora Duplicate Questions are

available:

[distilbert-base-nli-stsb-quora-ranking](https://huggingface.co/sentence-transformers/distilbert-base-

nli-stsb-quora-ranking):

We

extended

the

[distilbert-base-nli-stsb-mean-tokens](https://huggingface.co/sentence-transformers/distilbert-base-nl

i-stsb-mean-tokens) model and trained it with _OnlineContrastiveLoss_ and with

MultipleNegativesRankingLoss on the Quora Duplicate questions dataset. For the code, see

[training_multi-task-learning.py](https://github.com/UKPLab/sentence-transformers/tree/master/exa

mples/training/quora_duplicate_questions/training_multi-task-learning.py)

[distilbert-multilingual-nli-stsb-quora-ranking](https://huggingface.co/sentence-transformers/distilbert-

multilingual-nli-stsb-quora-ranking): Extension of _distilbert-base-nli-stsb-quora-ranking_ to be

multi-lingual. Trained on parallel data for 50 languages.

You can load & use pre-trained models like this:

from sentence_transformers import SentenceTransformer

model = SentenceTransformer("distilbert-base-nli-stsb-quora-ranking")

[Previous](../paraphrases/README.html "Paraphrase Data") [Next](../ms_marco/README.html "MS MARCO")

* * *

(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).