[![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-Process / Multi-GPU Encoding](../../applications/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](../../applications/semantic-search/README.html)
* [Background](../../applications/semantic-search/README.html#background)
* [Symmetric vs. Asymmetric Semantic Search](../../applications/semantic-search/README.html#symmetric-vs-asymmetric-semantic-search)
* [Manual Implementation](../../applications/semantic-search/README.html#manual-implementation)
* [Optimized Implementation](../../applications/semantic-search/README.html#optimized-implementation)
* [Speed Optimization](../../applications/semantic-search/README.html#speed-optimization)
* [Elasticsearch](../../applications/semantic-search/README.html#elasticsearch)
* [Approximate Nearest Neighbor](../../applications/semantic-search/README.html#approximate-nearest-neighbor)
* [Retrieve & Re-Rank](../../applications/semantic-search/README.html#retrieve-re-rank)
* [Examples](../../applications/semantic-search/README.html#examples)
* [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)
* [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)

* [Scalar (int8) 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-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)

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

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

Cross Encoder

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#den

oisingautoencoderloss)

*

[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/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#rerankingevaluator)
*
[SentenceEvaluator](../../../docs/package_reference/sentence_transformer/evaluation.html#sentenceevaluator)
*
[SequentialEvaluator](../../../docs/package_reference/sentence_transformer/evaluation.html#sequentialevaluator)
*
[TranslationEvaluator](../../../docs/package_reference/sentence_transformer/evaluation.html#translationevaluator)
*
[TripletEvaluator](../../../docs/package_reference/sentence_transformer/evaluation.html#tripletevaluator)
    * [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#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)

*

[`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_tra

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

* Natural Language Inference

* [Edit on GitHub](https://github.com/UKPLab/sentence-transformers/blob/master/examples/training/nli/README.md)

* * *

Natural Language Inferenceïƒ•

Given two sentence (premise and hypothesis), Natural Language Inference (NLI) is the task of deciding if the premise entails the hypothesis, if they are contradiction, or if they are neutral. Commonly used NLI dataset are [SNLI](https://huggingface.co/datasets/stanfordnlp/snli) and [MultiNLI](https://huggingface.co/datasets/nyu-mll/multi_nli).

[Conneau et al.](https://arxiv.org/abs/1705.02364) showed that NLI data can be quite useful when training Sentence Embedding methods. We also found this in our [Sentence-BERT-Paper](https://arxiv.org/abs/1908.10084) and often use NLI as a first fine-tuning step for sentence embedding methods.

To train on NLI, see the following example files:

1. **[training_nli.py](https://github.com/UKPLab/sentence-transformers/tree/master/examples/training/nli/training_nli.py)** :

This example uses

[`SoftmaxLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.SoftmaxLoss
"sentence_transformers.losses.SoftmaxLoss") as described in the original
[Sentence Transformers paper](<https://arxiv.org/abs/1908.10084>).

2.

[training_nli_v2.py](https://github.com/UKPLab/sentence-transformers/tree/master/examples/training/nli/training_nli_v2.py) :

The
[`SoftmaxLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.SoftmaxLoss
"sentence_transformers.losses.SoftmaxLoss") as used in our original SBERT
paper does not yield optimal performance. A better loss is
[`MultipleNegativesRankingLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.MultipleNegativesRankingLoss
"sentence_transformers.losses.MultipleNegativesRankingLoss"), where we provide
pairs or triplets. In this script, we provide a triplet of the format:
(anchor, entailment_sentence, contradiction_sentence). The NLI data provides
such triplets. The
[`MultipleNegativesRankingLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.MultipleNegativesRankingLoss
"sentence_transformers.losses.MultipleNegativesRankingLoss") yields much
higher performances and is more intuitive than
[`SoftmaxLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.SoftmaxLoss
"sentence_transformers.losses.SoftmaxLoss"). We have used this loss to train

the paraphrase model in our [Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation](https://arxiv.org/abs/2004.09813) paper.

3.

[training_nli_v3.py](https://github.com/UKPLab/sentence-transformers/tree/master/examples/training/nli/training_nli_v3.py)

Following the [GISTEmbed](https://arxiv.org/abs/2402.16829) paper, we can modify the in-batch negative selection from [`MultipleNegativesRankingLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.MultipleNegativesRankingLoss "sentence_transformers.losses.MultipleNegativesRankingLoss") using a guiding model. Candidate negative pairs are ignored during training if the guiding model considers the pair to be too similar. In practice, the [`GISTEmbedLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.GISTEmbedLoss "sentence_transformers.losses.GISTEmbedLoss") tends to produce a stronger training signal than [`MultipleNegativesRankingLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.MultipleNegativesRankingLoss "sentence_transformers.losses.MultipleNegativesRankingLoss") at the cost of some training overhead for running inference on the guiding model.

Dataïƒ•

We combine [SNLI](https://huggingface.co/datasets/stanfordnlp/snli) and

[MultiNLI](https://huggingface.co/datasets/nyu-mll/multi_nli) into a dataset
we call [AllNLI](https://huggingface.co/datasets/sentence-transformers/all-
nli). These two datasets contain sentence pairs and one of three labels:
entailment, neutral, contradiction:

Sentence A (Premise)	Sentence B (Hypothesis)	Label
A soccer game with multiple males playing.	Some men are playing a sport.	entailment
An older and younger man smiling.	Two men are smiling and laughing at the cats playing on the floor.	neutral
A man inspects the uniform of a figure in some East Asian country.	The man is sleeping.	contradiction

We format AllNLI in a few different subsets, compatible with different loss
functions. See for example the [triplet subset of
AllNLI](https://huggingface.co/datasets/sentence-transformers/all-
nli/viewer/triplet).

SoftmaxLossïƒ•

[Conneau et al.](https://arxiv.org/abs/1705.02364) described how a softmax
classifier on top of a [siamese
network](https://en.wikipedia.org/wiki/Siamese_neural_network) can be used to
learn meaningful sentence representation. We can achieve this by using
[`SoftmaxLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transfo
rmers.losses.SoftmaxLoss
"sentence_transformers.losses.SoftmaxLoss"):

![SBERT SoftmaxLoss](https://raw.githubusercontent.com/UKPLab/sentence-transformers/master/docs/img/SBERT_SoftmaxLoss.png)

We pass the two sentences through our SentenceTransformer model and get the sentence embeddings _u_ and _v_. We then concatenate _u_ , _v_ and _|u-v|_ to form one long vector. This vector is then passed to a softmax classifier, which predicts our three classes (entailment, neutral, contradiction).

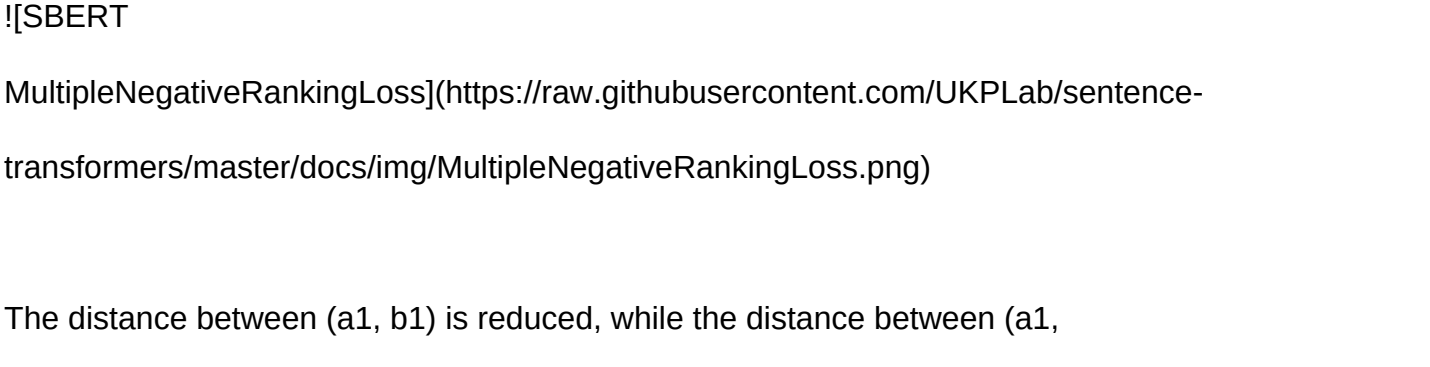
This setup learns sentence embeddings that can later be used for wide variety of tasks.

MultipleNegativesRankingLossïƒ•

That the [`SoftmaxLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.SoftmaxLoss "sentence_transformers.losses.SoftmaxLoss") with NLI data produces (relatively) good sentence embeddings is rather coincidental. The [`MultipleNegativesRankingLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.MultipleNegativesRankingLoss "sentence_transformers.losses.MultipleNegativesRankingLoss") is much more intuitive and produces significantly better sentence representations.

The training data for MultipleNegativesRankingLoss consists of sentence pairs [(a1, b1), â€¦, (an, bn)] where we assume that (ai, bi) are similar sentences and (ai, bj) are dissimilar sentences for i != j. The minimizes the distance

between (ai, bi) while it simultaneously maximizes the distance (ai, bj) for all i != j. For example, in the following picture:



The distance between (a1, b1) is reduced, while the distance between (a1, b2â€¦5) will be increased. The same is done for a2, â€¦, a5.

Using [`MultipleNegativesRankingLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.MultipleNegativesRankingLoss "sentence_transformers.losses.MultipleNegativesRankingLoss") with NLI is rather easy: We define sentences that have an _entailment_ label as positive pairs. E.g, we have pairs like (_â€œA soccer game with multiple males playing.â€•_ , _â€œSome men are playing a sport.â€•_) and want that these pairs are close in vector space. The [pair subset of AllNLI](https://huggingface.co/datasets/sentence-transformers/all-nli/viewer/pair) has been prepared in this format.

MultipleNegativesRankingLoss with Hard Negativesïƒ•

We can further improve MultipleNegativesRankingLoss by providing triplets rather than pairs: [(a1, b1, c1), â€¦, (an, bn, cn)]. The samples for ci are so-called hard-negatives: On a lexical level, they are similar to ai and bi, but on a semantic level, they mean different things and should not be close to

ai in the vector space.

For NLI data, we can use the contradiction-label to create such triplets with a hard negative. So our triplets look like this: (â€• _A soccer game with multiple males playing.â€•_ , _â€œSome men are playing a sport.â€•_ , _â€œA group of men playing a baseball game.â€•_). We want the sentences _â€œA soccer game with multiple males playing.â€•_ and _â€œSome men are playing a sport.â€•_ to be close in the vector space, while there should be a larger distance between _â€œA soccer game with multiple males playing.â€•_ and â€œ _A group of men playing a baseball game.â€•_. The [triplet subset of AllNLI](https://huggingface.co/datasets/sentence-transformers/all-nli/viewer/triplet) has been prepared in this format.

GISTEmbedLossïƒ•

[`MultipleNegativesRankingLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.MultipleNegativesRankingLoss "sentence_transformers.losses.MultipleNegativesRankingLoss") can be extended even further by recognizing that the in-batch negative sampling as shown in this example is a bit flawed. In particular, we automatically assume that the pairs (a1, b2), â€¦, (a1, bn) are negative, but that does not strictly have to be true.

To address this, [`GISTEmbedLoss`](../../../docs/package_reference/sentence_transformer/losses.html#sentence_transformers.losses.GISTEmbedLoss "sentence_transformers.losses.GISTEmbedLoss") uses a Sentence Transformer

model to guide the in-batch negative sample selection. In particular, if the guide model considers the similarity of (a1, bn) to be larger than (a1, b1), then the (a1, bn) pair is considered a false negative and consequently ignored in the training process. In essence, this results in higher quality training data for the model.

[Previous](../sts/README.html "Semantic Textual Similarity") [Next](../paraphrases/README.html "Paraphrase Data")

* * *