[![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](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](usage/semantic_textual_similarity.html)
* [Similarity Calculation](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](usage/efficiency.html)
    * [PyTorch](usage/efficiency.html#pytorch)
    * [ONNX](usage/efficiency.html#onnx)
    * [OpenVINO](usage/efficiency.html#openvino)
    * [Benchmarks](usage/efficiency.html#benchmarks)
  * [Creating Custom Models](usage/custom_models.html)
    * [Structure of Sentence Transformer Models](usage/custom_models.html#structure-of-sentence-transformer-models)

    * [Datasets on the Hugging Face Hub](dataset_overview.html#datasets-on-the-hugging-face-hub)
    * [Pre-existing Datasets](dataset_overview.html#pre-existing-datasets)
  * Loss Overview
    * Loss modifiers
    * Distillation
    * Commonly used Loss Functions
    * Custom Loss Functions
  * [Training Examples](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)

* [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](training/distributed.html)
  * [Comparison](training/distributed.html#comparison)
  * [FSDP](training/distributed.html#fsdp)

Cross Encoder

* [Usage](../cross_encoder/usage/usage.html)
  * [Retrieve & Re-Rank](../../examples/applications/retrieve_rerank/README.html)
    * [Retrieve & Re-Rank Pipeline](../../examples/applications/retrieve_rerank/README.html#retrieve-re-rank-pipeline)
    * [Retrieval: Bi-Encoder](../../examples/applications/retrieve_rerank/README.html#retrieval-bi-encoder)
    * [Re-Ranker: Cross-Encoder](../../examples/applications/retrieve_rerank/README.html#re-ranker-cross-encoder)
  * [Example Scripts](../../examples/applications/retrieve_rerank/README.html#example-scripts)
    * [Pre-trained Bi-Encoders (Retrieval)](../../examples/applications/retrieve_rerank/README.html#pre-trained-bi-encoders-retrieval)
    * [Pre-trained Cross-Encoders (Re-Ranker)](../../examples/applications/retrieve_rerank/README.html#pre-trained-cross-encoders-re-ranker)
* [Pretrained Models](../cross_encoder/pretrained_models.html)
  * [MS MARCO](../cross_encoder/pretrained_models.html#ms-marco)
  * [SQuAD (QNLI)](../cross_encoder/pretrained_models.html#squad-qnli)
  * [STSbenchmark](../cross_encoder/pretrained_models.html#stsbenchmark)
  * [Quora Duplicate Questions](../cross_encoder/pretrained_models.html#quora-duplicate-questions)

    * [NLI](../cross_encoder/pretrained_models.html#nli)
    * [Community Models](../cross_encoder/pretrained_models.html#community-models)
  * [Training Overview](../cross_encoder/training_overview.html)
  * [Training Examples](../cross_encoder/training/examples.html)
    * [MS MARCO](../../examples/training/ms_marco/cross_encoder_README.html)
    * [Cross-Encoder](../../examples/training/ms_marco/cross_encoder_README.html#cross-encoder)
    * [Cross-Encoder Knowledge Distillation](../../examples/training/ms_marco/cross_encoder_README.html#cross-encoder-knowledge-distillation)

Package Reference

  * [Sentence Transformer](../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#embeddingsimilarityevaluator)

*

[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#parallelsentencesdataset)
*
[SentenceLabelDataset](../package_reference/sentence_transformer/datasets.html#sentencelabeldataset)
*
[DenoisingAutoEncoderDataset](../package_reference/sentence_transformer/datasets.html#denoisingautoencoderdataset)
*
[NoDuplicatesDataLoader](../package_reference/sentence_transformer/datasets.html#noduplicatesd

ataloader)
    * [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#cecorrelationevaluat

or)
    * [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)
  * Loss Overview
  * [Edit on GitHub](https://github.com/UKPLab/sentence-transformers/blob/master/docs/sentence_transformer/loss_overview.md)

* * *

Loss Overviewïƒ•

Loss functions play a critical role in the performance of your fine-tuned model. Sadly, there is no â€œone size fits allâ€• loss function. Ideally, this table should help narrow down your choice of loss function(s) by matching them to your data formats.

Note

You can often convert one training data format into another, allowing more loss functions to be viable for your scenario. For example, `(sentence_A, sentence_B) pairs` with `class` labels can be converted into `(anchor, positive, negative) triplets` by sampling sentences with the same or different

classes.

Inputs | Labels | Appropriate Loss Functions

---|---|---

`single sentences` | `class` | [`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)

`single sentences` | `none` | [`ContrastiveTensionLoss`](../package_reference/sentence_transformer/losses.html#contrastivetensionloss) [`DenoisingAutoEncoderLoss`](../package_reference/sentence_transformer/losses.html#denoisingautoencoderloss)

`(anchor, anchor) pairs` | `none` | [`ContrastiveTensionLossInBatchNegatives`](../package_reference/sentence_transformer/losses.html#contrastivetensionlossinbatchnegatives)

`(damaged_sentence, original_sentence) pairs` | `none` | [`DenoisingAutoEncoderLoss`](../package_reference/sentence_transformer/losses.html#denoisingautoencoderloss)

`(sentence_A, sentence_B) pairs` | `class` | [`SoftmaxLoss`](../package_reference/sentence_transformer/losses.html#softmaxloss)

`(anchor, positive) pairs` | `none` | [`MultipleNegativesRankingLoss`](../package_reference/sentence_transformer/losses.html#multiple

negativesrankingloss)
[`CachedMultipleNegativesRankingLoss`](../package_reference/sentence_transformer/losses.html#cachedmultiplenegativesrankingloss)
[`MultipleNegativesSymmetricRankingLoss`](../package_reference/sentence_transformer/losses.html#multiplenegativessymmetricrankingloss)
[`CachedMultipleNegativesSymmetricRankingLoss`](../package_reference/sentence_transformer/losses.html#cachedmultiplenegativessymmetricrankingloss)
[`MegaBatchMarginLoss`](../package_reference/sentence_transformer/losses.html#megabatchmarginloss)
[`GISTEmbedLoss`](../package_reference/sentence_transformer/losses.html#gistembedloss)
[`CachedGISTEmbedLoss`](../package_reference/sentence_transformer/losses.html#cachedgistembedloss)
`(anchor, positive/negative) pairs` | `1 if positive, 0 if negative` |
[`ContrastiveLoss`](../package_reference/sentence_transformer/losses.html#contrastiveloss)
[`OnlineContrastiveLoss`](../package_reference/sentence_transformer/losses.html#onlinecontrastiveloss)
`(sentence_A, sentence_B) pairs` | `float similarity score` |
[`CoSENTLoss`](../package_reference/sentence_transformer/losses.html#cosentloss)
[`AnglELoss`](../package_reference/sentence_transformer/losses.html#angleloss)
[`CosineSimilarityLoss`](../package_reference/sentence_transformer/losses.html#cosinesimilarityloss)
`(anchor, positive, negative) triplets` | `none` |
[`MultipleNegativesRankingLoss`](../package_reference/sentence_transformer/losses.html#multiplenegativesrankingloss)
[`CachedMultipleNegativesRankingLoss`](../package_reference/sentence_transformer/losses.html#cachedmultiplenegativesrankingloss)
[`TripletLoss`](../package_reference/sentence_transformer/losses.html#tripletloss)

[`CachedGISTEmbedLoss`](../package_reference/sentence_transformer/losses.html#cachedgistembedloss)
[`GISTEmbedLoss`](../package_reference/sentence_transformer/losses.html#gistembedloss)
`(anchor, positive, negative_1, ..., negative_n)` | `none` |
[`MultipleNegativesRankingLoss`](../package_reference/sentence_transformer/losses.html#multiplenegativesrankingloss)
[`CachedMultipleNegativesRankingLoss`](../package_reference/sentence_transformer/losses.html#cachedmultiplenegativesrankingloss)
[`CachedGISTEmbedLoss`](../package_reference/sentence_transformer/losses.html#cachedgistembedloss)

Loss modifiersïƒ•

These loss functions can be seen as _loss modifiers_ : they work on top of
standard loss functions, but apply those loss functions in different ways to
try and instil useful properties into the trained embedding model.

For example, models trained with
[`MatryoshkaLoss`](../package_reference/sentence_transformer/losses.html#matryoshkaloss)
produce embeddings whose size can be truncated without notable losses in
performance, and models trained with
[`AdaptiveLayerLoss`](../package_reference/sentence_transformer/losses.html#adaptivelayerloss)
still perform well when you remove model layers for faster inference.

Texts	Labels	Appropriate Loss Functions
`any`	`any`

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

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

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

Distillationïƒ•

These loss functions are specifically designed to be used when distilling the knowledge from one model into another. For example, when finetuning a small model to behave more like a larger & stronger one, or when finetuning a model to become multi-lingual.

Texts | Labels | Appropriate Loss Functions

---|---|---

`sentence` | `model sentence embeddings` | [`MSELoss`](../package_reference/sentence_transformer/losses.html#mseloss)

`sentence_1, sentence_2, ..., sentence_N` | `model sentence embeddings` | [`MSELoss`](../package_reference/sentence_transformer/losses.html#mseloss)

`(query, passage_one, passage_two) triplets` | `gold_sim(query, passage_one) - gold_sim(query, passage_two)` | [`MarginMSELoss`](../package_reference/sentence_transformer/losses.html#marginmseloss)

Commonly used Loss Functionsïƒ•

In practice, not all loss functions get used equally often. The most common scenarios are:

* `(anchor, positive) pairs` without any labels:

[`MultipleNegativesRankingLoss`](../package_reference/sentence_transformer/losses.html#multiplenegativesrankingloss) is commonly used to train the top performing embedding models. This data is often relatively cheap to obtain, and the models are generally very performant. [`CachedMultipleNegativesRankingLoss`](../package_reference/sentence_transformer/losses.html#cachedmultiplenegativesrankingloss) is often used to increase the batch size, resulting in superior performance.

* `(sentence_A, sentence_B) pairs` with a `float similarity score`: [`CosineSimilarityLoss`](../package_reference/sentence_transformer/losses.html#cosinesimilarityloss) is traditionally used a lot, though more recently [`CoSENTLoss`](../package_reference/sentence_transformer/losses.html#cosentloss) and [`AnglELoss`](../package_reference/sentence_transformer/losses.html#angleloss) are used as drop-in replacements with superior performance.

Custom Loss Functionsïƒ•

Advanced users can create and train with their own loss functions. Custom loss functions only have a few requirements:

* They must be a subclass of [`torch.nn.Module`](https://pytorch.org/docs/stable/generated/torch.nn.Module.html#torch.nn.Module "\(in PyTorch v2.5\)").

* They must have `model` as the first argument in the constructor.

* They must implement a `forward` method that accepts `sentence_features` and `labels`. The former is a list of tokenized batches, one element for each column. These tokenized batches can be

fed directly to the `model` being trained to produce embeddings. The latter is an optional tensor of labels. The method must return a single loss value.

To get full support with the automatic model card generation, you may also
wish to implement:

* a `get_config_dict` method that returns a dictionary of loss parameters.

* a `citation` property so your work gets cited in all models that train with the loss.

[Previous](dataset_overview.html "Dataset Overview") [Next
](training/examples.html "Training Examples")

* * *