[![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](../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](../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)

* [Masked Language Model (MLM)](../../unsupervised_learning/README.html#masked-language-model-mlm)
    * [GenQ](../../unsupervised_learning/README.html#genq)
    * [GPL](../../unsupervised_learning/README.html#gpl)
* [Performance Comparison](../../unsupervised_learning/README.html#performance-comparison)
  * [Domain Adaptation](../../domain_adaptation/README.html)
* [Domain Adaptation vs. Unsupervised Learning](../../domain_adaptation/README.html#domain-adaptation-vs-unsupervised-learning)
    * [Adaptive Pre-Training](../../domain_adaptation/README.html#adaptive-pre-training)
* [GPL: Generative Pseudo-Labeling](../../domain_adaptation/README.html#gpl-generative-pseudo-labeling)
  * [Hyperparameter Optimization](../hpo/README.html)
    * [HPO Components](../hpo/README.html#hpo-components)
    * [Putting It All Together](../hpo/README.html#putting-it-all-together)
    * [Example Scripts](../hpo/README.html#example-scripts)
  * [Distributed Training](../../../docs/sentence_transformer/training/distributed.html)
    * [Comparison](../../../docs/sentence_transformer/training/distributed.html#comparison)
    * [FSDP](../../../docs/sentence_transformer/training/distributed.html#fsdp)

Cross Encoder

* [Usage](../../../docs/cross_encoder/usage/usage.html)
  * [Retrieve & Re-Rank](../../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)
* [Cross-Encoder Knowledge Distillation](../ms_marco/cross_encoder_README.html#cross-encoder-knowledge-distillation)

Package Reference

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

html)

*

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

*

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

*

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

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

*

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

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

*

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

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

*

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

*

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

*

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

tripletloss)

*

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

*

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

*

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

*

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

*

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

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

*

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

*

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

*

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

*

[CachedGISTEmbedLoss](../../../docs/package_reference/sentence_transformer/losses.html#cachedgistembedloss)
    * [MSELoss](../../../docs/package_reference/sentence_transformer/losses.html#mseloss)
* [MarginMSELoss](../../../docs/package_reference/sentence_transformer/losses.html#marginmseloss)
* [MatryoshkaLoss](../../../docs/package_reference/sentence_transformer/losses.html#matryoshkaloss)
* [Matryoshka2dLoss](../../../docs/package_reference/sentence_transformer/losses.html#matryoshka2dloss)
* [AdaptiveLayerLoss](../../../docs/package_reference/sentence_transformer/losses.html#adaptivelayerloss)
* [MegaBatchMarginLoss](../../../docs/package_reference/sentence_transformer/losses.html#megabatchmarginloss)
* [MultipleNegativesRankingLoss](../../../docs/package_reference/sentence_transformer/losses.html#multiplenegativesrankingloss)
* [CachedMultipleNegativesRankingLoss](../../../docs/package_reference/sentence_transformer/losses.html#cachedmultiplenegativesrankingloss)
* [MultipleNegativesSymmetricRankingLoss](../../../docs/package_reference/sentence_transformer/losses.html#multiplenegativessymmetricrankingloss)

*
[CachedMultipleNegativesSymmetricRankingLoss](../../../docs/package_reference/sentence_transformer/losses.html#cachedmultiplenegativessymmetricrankingloss)
    * [SoftmaxLoss](../../../docs/package_reference/sentence_transformer/losses.html#softmaxloss)
    * [TripletLoss](../../../docs/package_reference/sentence_transformer/losses.html#tripletloss)
  * [Samplers](../../../docs/package_reference/sentence_transformer/sampler.html)
*
[BatchSamplers](../../../docs/package_reference/sentence_transformer/sampler.html#batchsamplers)
*
[MultiDatasetBatchSamplers](../../../docs/package_reference/sentence_transformer/sampler.html#multidatasetbatchsamplers)
  * [Evaluation](../../../docs/package_reference/sentence_transformer/evaluation.html)
*
[BinaryClassificationEvaluator](../../../docs/package_reference/sentence_transformer/evaluation.html#binaryclassificationevaluator)
*
[EmbeddingSimilarityEvaluator](../../../docs/package_reference/sentence_transformer/evaluation.html#embeddingsimilarityevaluator)
*
[InformationRetrievalEvaluator](../../../docs/package_reference/sentence_transformer/evaluation.html#informationretrievalevaluator)
*
[NanoBEIREvaluator](../../../docs/package_reference/sentence_transformer/evaluation.html#nanobeirevaluator)
*
[MSEEvaluator](../../../docs/package_reference/sentence_transformer/evaluation.html#mseevaluator

)

*

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

*

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

*

[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_transformers.backend.export_static_quantized_openvino_model)
* [Similarity Metrics](../../../docs/package_reference/util.html#module-sentence_transformers.util)
* [`cos_sim()`](../../../docs/package_reference/util.html#sentence_transformers.util.cos_sim)
* [`dot_score()`](../../../docs/package_reference/util.html#sentence_transformers.util.dot_score)
*

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

*

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

*

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

*

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

*

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

*

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

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

* [](../../../index.html)
* [Training Examples](../../../docs/sentence_transformer/training/examples.html)
* Augmented SBERT

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

* * *

Augmented SBERTïƒ•

Motivationïƒ•

Bi-encoders (a.k.a. sentence embeddings models) require substantial training data and fine-tuning over the target task to achieve competitive performances. However, in many scenarios, there is only little training data available.

To solve this practical issue, we release an effective data-augmentation strategy known as **Augmented SBERT** where we utilize a high performing and slow cross-encoder (BERT) to label a larger set of input pairs to augment the training data for the bi-encoder (SBERT).

For more details, refer to our publication - [Augmented SBERT: Data Augmentation Method for Improving Bi-Encoders for Pairwise Sentence Scoring Tasks](https://arxiv.org/abs/2010.08240) which is a joint effort by Nandan Thakur, Nils Reimers and Johannes Daxenberger of UKP Lab, TU Darmstadt.

Chien Vu also wrote a nice blog article on this technique: [Advance BERT model via transferring knowledge from Cross-Encoders to Bi-Encoders](https://towardsdatascience.com/advance-nlp-model-via-transferring-knowledge-from-cross-encoders-to-bi-encoders-3e0fc564f554)

Extend to your own datasetsïƒ•

Scenario 1: Limited or small annotated datasets (few labeled sentence-pairs (1k-3k))

If you have specialized datasets in your company or research which are small-sized or contain labeled few sentence-pairs. You can extend the idea of Augmented SBERT (in-domain) strategy by training a cross-encoder over your small gold dataset and use BM25 sampling to generate combinations not seen earlier. Use the cross-encoder to label these unlabeled pairs to create the silver dataset. Finally train a bi-encoder (i.e. SBERT) over your extended dataset (gold+silver) dataset as shown in [train_sts_indomain_bm25.py](https://github.com/UKPLab/sentence-transformers/tree/master/examples/training/data_augmentation/train_sts_indomain_bm25.py).

Scenario 2: No annotated datasets (Only unlabeled sentence-pairs)

If you have specialized datasets in your company or research which only contain unlabeled sentence-pairs. You can extend the idea of Augmented SBERT (domain-transfer) strategy by training a cross-encoder over a source dataset which is annotated (for eg. QQP). Use this cross-encoder to label your specialised unlabeled dataset i.e. target dataset. Finally train a bi-encoder i.e. SBERT over your labeled target dataset as shown in [train_sts_qqp_crossdomain.py](https://github.com/UKPLab/sentence-transformers/tree/master/examples/training/data_augmentation/train_sts_qqp_crossdomain.py).

Methodologyïƒ•

There are two major scenarios for the Augmented SBERT approach for pairwise-sentence regression or classification tasks.

Scenario 1: Limited or small annotated datasets (few labeled sentence-pairs)ïƒ•

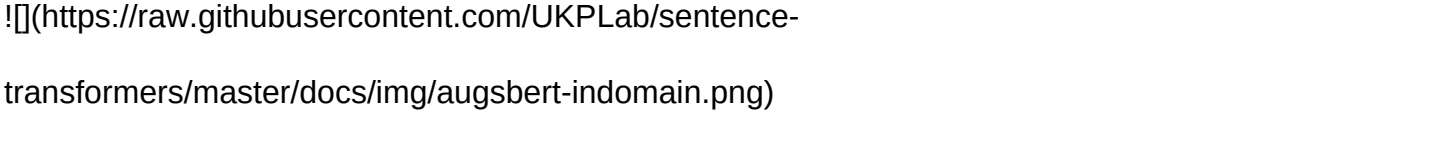
We apply the Augmented SBERT (**In-domain**) strategy, it involves three steps -

* Step 1: Train a cross-encoder (BERT) over the small (gold or annotated) dataset

* Step 2.1: Create pairs by recombination and reduce the pairs via BM25 or semantic search

* Step 2.2: Weakly label new pairs with cross-encoder (BERT). These are silver pairs or (silver) dataset

* Step 3: Finally, train a bi-encoder (SBERT) on the extended (gold + silver) training dataset



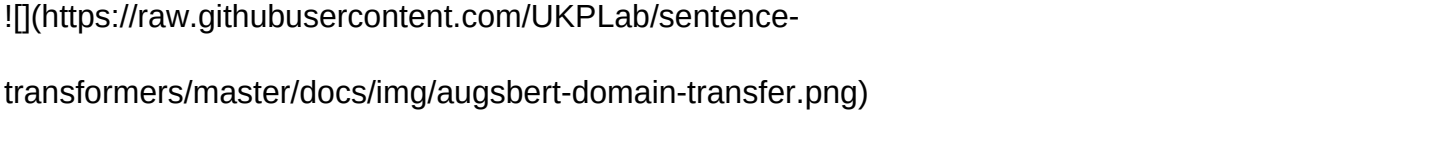
Scenario 2: No annotated datasets (Only unlabeled sentence-pairs)ïƒ•

We apply the Augmented SBERT (**Domain-Transfer**) strategy, it involves three steps -

* Step 1: Train from scratch a cross-encoder (BERT) over a source dataset, for which we contain annotations

* Step 2: Use this cross-encoder (BERT) to label your target dataset i.e. unlabeled sentence pairs

* Step 3: Finally, train a bi-encoder (SBERT) on the labeled target dataset



Trainingïƒ•

The [examples/training/data_augmentation](https://github.com/UKPLab/sentence-transformers/blob/master/examples/training/data_augmentation/) folder contains simple training examples for each scenario explained below:

*

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

* This script trains a bi-encoder (SBERT) model from scratch for STS benchmark dataset with seed-optimization.

* Seed optimization technique is inspired from [(Dodge et al., 2020)](https://arxiv.org/abs/2002.06305).

* For Seed opt., we train our bi-encoder for various seeds and evaluate using an early stopping algorithm.

* Finally, measure dev performance across the seeds to get the highest performing seeds.

*

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

* This script trains a bi-encoder (SBERT) model from scratch for STS benchmark dataset using easy data augmentation.

* Data augmentation strategies are used from popular [nlpaug](https://github.com/makcedward/nlpaug) package.

* Augment single sentences with synonyms using (word2vec, BERT or WordNet). Forms our silver dataset.

* Train bi-encoder model on both original small training dataset and synonym based silver dataset.

*

[train_sts_indomain_bm25.py](https://github.com/UKPLab/sentence-transformers/tree/master/examples/training/data_augmentation/train_sts_indomain_bm25.py)

* Script initially trains a cross-encoder (BERT) model from scratch for small STS benchmark dataset.

* Recombine sentences from our small training dataset and form lots of sentence-pairs.

* Limit number of combinations with BM25 sampling using [Elasticsearch](https://www.elastic.co/).

* Retrieve top-k sentences given a sentence and label these pairs using the cross-encoder (silver dataset).

* Train a bi-encoder (SBERT) model on both gold + silver STSb dataset. (Augmented SBERT (In-domain) Strategy).

*

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

* This script initially trains a cross-encoder (BERT) model from scratch for small STS benchmark dataset.

* We recombine sentences from our small training dataset and form lots of sentence-pairs.

* Limit number of combinations with Semantic Search sampling using pretrained SBERT model.

* Retrieve top-k sentences given a sentence and label these pairs using the cross-encoder (silver dataset).

* Train a bi-encoder (SBERT) model on both gold + silver STSb dataset. (Augmented SBERT (In-domain) Strategy).

*

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

* This script initially trains a cross-encoder (BERT) model from scratch for STS benchmark dataset.

* Label the Quora Questions Pair (QQP) training dataset (Assume no labels present) using the cross-encoder.

* Train a bi-encoder (SBERT) model on the QQP dataset. (Augmented SBERT (Domain-Transfer) Strategy).

Citationïƒ•

If you use the code for augmented sbert, feel free to cite our publication
[Augmented SBERT: Data Augmentation Method for Improving Bi-Encoders for
Pairwise Sentence Scoring Tasks](https://arxiv.org/abs/2010.08240):

```
@article{thakur-2020-AugSBERT,
    title = "Augmented SBERT: Data Augmentation Method for Improving Bi-Encoders for Pairwise Sentence Scoring Tasks",
    author = "Thakur, Nandan and Reimers, Nils and Daxenberger, Johannes and Gurevych, Iryna",
    journal= "arXiv preprint arXiv:2010.08240",
    month = "10",
    year = "2020",
    url = "https://arxiv.org/abs/2010.08240",
}
```

[Previous](../distillation/README.html "Model Distillation") [Next

](../prompts/README.html "Training with Prompts")

* * *