[![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)

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)

* Training with PEFT Adapters

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

* * *

Training with PEFT Adaptersïƒ•

Sentence Transformers has been integrated with [PEFT](https://huggingface.co/docs/peft/en/index) (Parameter-Efficient Fine-Tuning), allowing you to finetune embedding models without fine-tuning all of the model parameters. Instead, with PEFT methods you are only finetuning a fraction of (extra) model parameters with only a minor hit in performance compared to full model finetuning.

PEFT Adapter models can be loaded just like any others, for example [tomaarsen/bert-base-uncased-gooaq-peft](https://huggingface.co/tomaarsen/bert-base-uncased-gooaq-peft) which does not contain a `model.safetensors` but only a tiny `adapter_model.safetensors`:

```
from sentence_transformers import SentenceTransformer

# Download from the ðŸ¤— Hub
```

```
model = SentenceTransformer("tomaarsen/bert-base-uncased-gooaq-peft")
# Run inference
sentences = [
    "is toprol xl the same as metoprolol?",
    "Metoprolol succinate is also known by the brand name Toprol XL. It is the extended-release form of metoprolol. Metoprolol succinate is approved to treat high blood pressure, chronic chest pain, and congestive heart failure.",
    "Metoprolol starts to work after about 2 hours, but it can take up to 1 week to fully take effect. You may not feel any different when you take metoprolol, but this doesn't mean it's not working. It's important to keep taking your medicine"
]
embeddings = model.encode(sentences)
print(embeddings.shape)
# [3, 768]

# Get the similarity scores for the embeddings
similarities = model.similarity(embeddings[0], embeddings[1:])
print(similarities)
# tensor([[0.7913, 0.4976]])
```

Compatibility Methodsïƒ•

The [`SentenceTransformer`](../../../docs/package_reference/sentence_transformer/SentenceTransformer.html#sentence_transformers.SentenceTransformer "sentence_transformers.SentenceTransformer") supports 7 methods for

interacting with the PEFT Adapters:

> *
>
[`add_adapter()`](../../../docs/package_reference/sentence_transformer/SentenceTransformer.html#sentence_transformers.SentenceTransformer.add_adapter
> "sentence_transformers.SentenceTransformer.add_adapter"): Adds a fresh new
> adapter to the current model for training.
>
> *
>
[`load_adapter()`](../../../docs/package_reference/sentence_transformer/SentenceTransformer.html#sentence_transformers.SentenceTransformer.load_adapter
> "sentence_transformers.SentenceTransformer.load_adapter"): Load adapter
> weights from a file or Hugging Face Hub repository.
>
> *
>
[`active_adapters()`](../../../docs/package_reference/sentence_transformer/SentenceTransformer.html#sentence_transformers.SentenceTransformer.active_adapters
> "sentence_transformers.SentenceTransformer.active_adapters"): Gets the
> current active adapters.
>
> *
>
[`set_adapter()`](../../../docs/package_reference/sentence_transformer/SentenceTransformer.html#sentence_transformers.SentenceTransformer.set_adapter

> "sentence_transformers.SentenceTransformer.set_adapter"): Tell your model to

> use a specific adapter and disable all others.

>

> *

>

[`enable_adapters()`](../../../docs/package_reference/sentence_transformer/SentenceTransformer.html#sentence_transformers.SentenceTransformer.enable_adapters

> "sentence_transformers.SentenceTransformer.enable_adapters"): Enable all

> adapters.

>

> *

>

[`disable_adapters()`](../../../docs/package_reference/sentence_transformer/SentenceTransformer.html#sentence_transformers.SentenceTransformer.disable_adapters

> "sentence_transformers.SentenceTransformer.disable_adapters"): Disable all

> adapters.

>

> *

>

[`get_adapter_state_dict()`](../../../docs/package_reference/sentence_transformer/SentenceTransformer.html#sentence_transformers.SentenceTransformer.get_adapter_state_dict

> "sentence_transformers.SentenceTransformer.get_adapter_state_dict"): Get the

> adapter state dict with the weights.

>

>

Adding a New Adapterïƒ•

Adding a new adapter to a model is as simple as calling
[`add_adapter()`](../../../docs/package_reference/sentence_transformer/SentenceTransformer.html#
sentence_transformers.SentenceTransformer.add_adapter
"sentence_transformers.SentenceTransformer.add_adapter") with a (subclass of)
[`PeftConfig`](https://huggingface.co/docs/peft/main/en/package_reference/config#peft.PeftConfig
"\(in peft vmain\)") on an initialized Sentence Transformer model. In the
following example, we use a
[`LoraConfig`](https://huggingface.co/docs/peft/main/en/package_reference/lora#peft.LoraConfig
"\(in peft vmain\)") instance.

```
from sentence_transformers import SentenceTransformer

# 1. Load a model to finetune with 2. (Optional) model card data
model = SentenceTransformer(
    "all-MiniLM-L6-v2",
    model_card_data=SentenceTransformerModelCardData(
        language="en",
        license="apache-2.0",
        model_name="all-MiniLM-L6-v2 adapter finetuned on GooAQ pairs",
    ),
)

# 3. Create a LoRA adapter for the model & add it
peft_config = LoraConfig(
```

```
        task_type=TaskType.FEATURE_EXTRACTION,
        inference_mode=False,
        r=64,
        lora_alpha=128,
        lora_dropout=0.1,
    )
    model.add_adapter(peft_config)

    # Proceed as usual... See https://sbert.net/docs/sentence_transformer/training_overview.html
```

Loading a Pretrained Adapterï*f*•

Weâ€™ve created a small adapter model called [tomaarsen/bert-base-uncased-gooaq-peft](https://huggingface.co/tomaarsen/bert-base-uncased-gooaq-peft) on top of the [bert-base-uncased](https://huggingface.co/bert-base-uncased) base model.

The `adapter_model.safetensors` is 9.44MB, only 2.14% of the size of the base modelâ€™s `model.safetensors`. To load an adapter model like this one, you can either load this adapter directly:

```
    from sentence_transformers import SentenceTransformer

    model = SentenceTransformer("tomaarsen/bert-base-uncased-gooaq-peft")
```

```
embeddings = model.encode(["This is an example sentence", "Each sentence is converted"])
print(embeddings.shape)
# (2, 768)
```

Or you can load the base model and load the adapter into it:

```
from sentence_transformers import SentenceTransformer

model = SentenceTransformer("bert-base-uncased")
model.load_adapter("tomaarsen/bert-base-uncased-gooaq-peft")
embeddings = model.encode(["This is an example sentence", "Each sentence is converted"])
print(embeddings.shape)
# (2, 768)
```

In most cases, the former is easiest, as it will work regardless of whether
the model is an adapter model or not.

Training Scriptïƒ•

See the following example file for a full example of how PEFT can be used with
Sentence Transformers:

*

[training_gooaq_lora.py](https://github.com/UKPLab/sentence-transformers/tree/master/examples/training/peft/training_gooaq_lora.py) : This is a simple recipe for finetuning [bert-base-uncased](https://huggingface.co/google-bert/bert-base-uncased) on the GooAQ question-answer dataset with the excellent MultipleNegativesRankingLoss, but it has been adapted to use a [LoRA adapter](https://huggingface.co/docs/peft/en/package_reference/lora) from PEFT.

This script was used to train [tomaarsen/bert-base-uncased-gooaq-peft](https://huggingface.co/tomaarsen/bert-base-uncased-gooaq-peft), which reached 0.4705 NDCG@10 on the NanoBEIR benchmark; only marginally behind [tomaarsen/bert-base-uncased-gooaq](https://huggingface.co/tomaarsen/bert-base-uncased-gooaq) which scores 0.4728 NDCG@10 with a modified script that uses full model finetuning.

[Previous](../prompts/README.html "Training with Prompts") [Next](../../unsupervised_learning/README.html "Unsupervised Learning")

* * *