![Logo](_static/logo.png)

Getting Started

Sentence Transformer

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)

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)

      * [Semantic Similarity Models](docs/sentence_transformer/pretrained_models.html#semantic-similarity-models)

      * [Bitext Mining](docs/sentence_transformer/pretrained_models.html#bitext-mining)

    * [Image & Text-Models](docs/sentence_transformer/pretrained_models.html#image-text-models)

    * [INSTRUCTOR models](docs/sentence_transformer/pretrained_models.html#instructor-models)

    * [Scientific Similarity Models](docs/sentence_transformer/pretrained_models.html#scientific-similarity-models)

  * [Training Overview](docs/sentence_transformer/training_overview.html)

    * [Why Finetune?](docs/sentence_transformer/training_overview.html#why-finetune)

    * [Training Components](docs/sentence_transformer/training_overview.html#training-components)

    * [Dataset](docs/sentence_transformer/training_overview.html#dataset)

      * [Dataset Format](docs/sentence_transformer/training_overview.html#dataset-format)

    * [Loss Function](docs/sentence_transformer/training_overview.html#loss-function)

    * [Training Arguments](docs/sentence_transformer/training_overview.html#training-arguments)

    * [Evaluator](docs/sentence_transformer/training_overview.html#evaluator)

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

      * [Callbacks](docs/sentence_transformer/training_overview.html#callbacks)

    * [Multi-Dataset Training](docs/sentence_transformer/training_overview.html#multi-dataset-training)

    * [Deprecated Training](docs/sentence_transformer/training_overview.html#deprecated-training)

    * [Best Base Embedding

Models](docs/sentence_transformer/training_overview.html#best-base-embedding-models)

  * [Dataset Overview](docs/sentence_transformer/dataset_overview.html)

    * [Datasets on the Hugging Face Hub](docs/sentence_transformer/dataset_overview.html#datasets-on-the-hugging-face-hub)

    * [Pre-existing Datasets](docs/sentence_transformer/dataset_overview.html#pre-existing-datasets)

  * [Loss Overview](docs/sentence_transformer/loss_overview.html)

    * [Loss modifiers](docs/sentence_transformer/loss_overview.html#loss-modifiers)

    * [Distillation](docs/sentence_transformer/loss_overview.html#distillation)

    * [Commonly used Loss Functions](docs/sentence_transformer/loss_overview.html#commonly-used-loss-functions)

    * [Custom Loss Functions](docs/sentence_transformer/loss_overview.html#custom-loss-functions)

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

    * [Semantic Textual Similarity](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)

      *

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](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](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](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](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](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#simil

arityfunction)

  * [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#contrastivete

nsionloss)

* [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#megabatchma

rginloss)

*

[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

* * SentenceTransformers Documentation

* [Edit on GitHub](https://github.com/UKPLab/sentence-transformers/blob/master/index.rst)

* * *

Tip

You can now use ONNX and OpenVINO backends for Sentence Transformer models. Read [SentenceTransformer > Usage > Speeding up Inference](docs/sentence_transformer/usage/efficiency.html) to learn more about the new backends and what they can mean for your inference speed.

Tip

Sentence Transformers v3.4 just released, introducing documentation for training with PEFT. Read [SentenceTransformer > Training Examples > Training with PEFT Adapters](examples/training/peft/README.html) to learn more about how you can use train embedding models without finetuning all model parameters.

# SentenceTransformers Documentationïƒ•

Sentence Transformers (a.k.a. SBERT) is the go-to Python module for accessing, using, and training state-of-the-art text and image embedding models. It can be used to compute embeddings using Sentence Transformer models ([quickstart](docs/quickstart.html#sentence-transformer)) or to calculate similarity scores using Cross-Encoder models ([quickstart](docs/quickstart.html#cross-encoder)). This unlocks a wide range of applications, including [semantic search](examples/applications/semantic-search/README.html), [semantic textual similarity](docs/usage/semantic_textual_similarity.html), and [paraphrase mining](examples/applications/paraphrase-mining/README.html).

A wide selection of over [5,000 pre-trained Sentence Transformers models](https://huggingface.co/models?library=sentence-transformers) are available for immediate use on ðŸ¤— Hugging Face, including many of the state-of-the-art models from the [Massive Text Embeddings Benchmark (MTEB) leaderboard](https://huggingface.co/spaces/mteb/leaderboard). Additionally, it is easy to [train or finetune your own models](docs/sentence_transformer/training_overview.html) using Sentence Transformers, enabling you to create custom models for your specific use cases.

Sentence Transformers was created by [UKPLab](http://www.ukp.tu-darmstadt.de/) and is being maintained by [ðŸ¤— Hugging Face](https://huggingface.co). Donâ€™t hesitate to open an issue on the [Sentence Transformers repository](https://github.com/UKPLab/sentence-transformers) if something is broken or if you have further questions.

# Usageïƒ•

See also

See the [Quickstart](docs/quickstart.html) for more quick information on how to use Sentence Transformers.

Using Sentence Transformer models is elementary:

Installation

You can install _sentence-transformers_ using pip:

```
pip install -U sentence-transformers
```

We recommend **Python 3.9+** and **PyTorch 1.11.0+**. See [installation](docs/installation.html) for further installation options.

```
from sentence_transformers import SentenceTransformer

# 1. Load a pretrained Sentence Transformer model
model = SentenceTransformer("all-MiniLM-L6-v2")

# The sentences to encode
sentences = [
    "The weather is lovely today.",
    "It's so sunny outside!",
    "He drove to the stadium.",
]

# 2. Calculate embeddings by calling model.encode()
embeddings = model.encode(sentences)
print(embeddings.shape)
```

```
    # [3, 384]

    # 3. Calculate the embedding similarities
    similarities = model.similarity(embeddings, embeddings)
    print(similarities)
    # tensor([[1.0000, 0.6660, 0.1046],
    #         [0.6660, 1.0000, 0.1411],
    #         [0.1046, 0.1411, 1.0000]])
```

# What Next?ïƒ•

Consider reading one of the following sections to answer the related questions:

* How to **use** Sentence Transformer models? [Sentence Transformers > Usage](docs/sentence_transformer/usage/usage.html)

* What Sentence Transformer **models** can I use? [Sentence Transformers > Pretrained Models](docs/sentence_transformer/pretrained_models.html)

* How do I make Sentence Transformer models **faster**? [Sentence Transformers > Usage > Speeding up Inference](docs/sentence_transformer/usage/efficiency.html)

* How do I **train/finetune** a Sentence Transformer model? [Sentence Transformers > Training Overview](docs/sentence_transformer/training_overview.html)

* How to **use** Cross Encoder models? [Cross Encoder > Usage](docs/cross_encoder/usage/usage.html)

* What Cross Encoder **models** can I use? [Cross Encoder > Pretrained Models](docs/cross_encoder/pretrained_models.html)

# Citingïƒ•

If you find this repository helpful, feel free to cite our publication [Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks](https://arxiv.org/abs/1908.10084):

>
> @inproceedings{reimers-2019-sentence-bert,
> title = "Sentence-BERT: Sentence Embeddings using Siamese BERT-
> Networks",
> author = "Reimers, Nils and Gurevych, Iryna",
> booktitle = "Proceedings of the 2019 Conference on Empirical Methods
> in Natural Language Processing",
> month = "11",
> year = "2019",
> publisher = "Association for Computational Linguistics",
> url = "https://arxiv.org/abs/1908.10084",
> }
>

If you use one of the multilingual models, feel free to cite our publication

[Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation](https://arxiv.org/abs/2004.09813):

>
> @inproceedings{reimers-2020-multilingual-sentence-bert,
>   title = "Making Monolingual Sentence Embeddings Multilingual using
> Knowledge Distillation",
>   author = "Reimers, Nils and Gurevych, Iryna",
>   booktitle = "Proceedings of the 2020 Conference on Empirical Methods
> in Natural Language Processing",
>   month = "11",
>   year = "2020",
>   publisher = "Association for Computational Linguistics",
>   url = "https://arxiv.org/abs/2004.09813",
> }
>

If you use the code for [data augmentation](https://github.com/UKPLab/sentence-transformers/tree/master/examples/training/data_augmentation), 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):

>
> @inproceedings{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",
>
>   booktitle = "Proceedings of the 2021 Conference of the North American
>
> Chapter of the Association for Computational Linguistics: Human Language
>
> Technologies",
>
>   month = jun,
>
>   year = "2021",
>
>   address = "Online",
>
>   publisher = "Association for Computational Linguistics",
>
>   url = "https://www.aclweb.org/anthology/2021.naacl-main.28",
>
>   pages = "296--310",
>
>   }
>

[Next](docs/installation.html "Installation")

* * *