Jina Embeddings: A Novel Set of High-Performance Sentence Embedding Models

Michael Günther Louis Milliken Jonathan Geuter Georgios Mastrapas Bo Wang Han Xiao

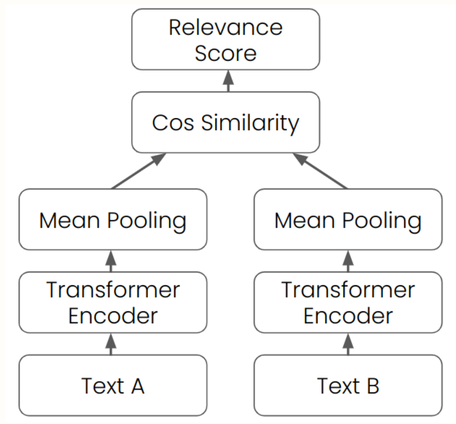
Jina AI
Ohlauer Str. 43, 10999 Berlin, Germany

Introduction

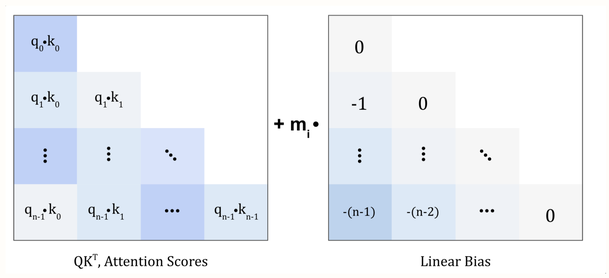
Jina Embeddings are a set of high-performance sentence embedding models adept at translating texts into vector representations that capture their semantics. These models excel in dense retrieval and semantic textual similarity applications. To build the Jina Embeddings v1 model suite, we have focused on setting up an effective data-cleaning pipeline and exploring novel strategies to let embedding models become better at differentiating between entailed and negated statements. We have also put our efforts into optimizing the training process and building models that support encoding long text sequences. These efforts have borne fruit in the Jina Embeddings v2 suite, which encodes up to 8192 tokens into a single embedding representation.

Architecture

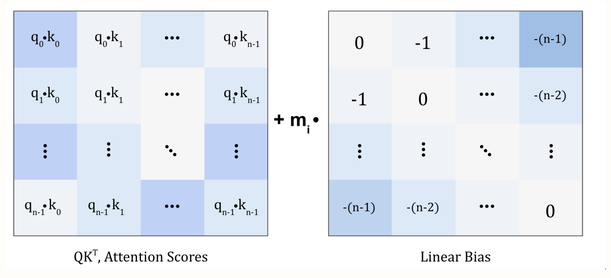
The architecure of our models largely replicates the SentenceBERT architecture proposed in [1]:



Jina Embeddings v1 relies on the encoder part of the T5 Model [2], while Jina Embeddings v2 is based on our own Jina BERT model, which encodes long text sequences of up to 8192 tokens. To support such sequences, we have abandoned traditional positional encoding and use the AliBi approach [3] to inject position information as linear bias in the self-attention calculation.



(a) Original Causal ALiBi



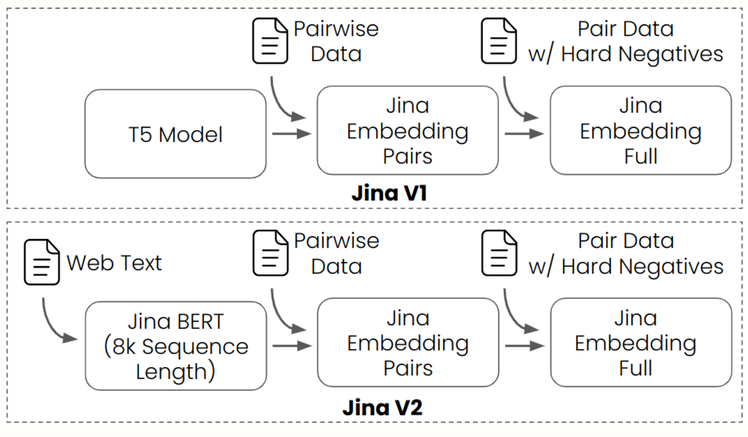
(b) Modified ALiBi for Encoders

Position Encoding with AliBi

AliBi biases were originally designed for decoder-only models and thus only consider one causal direction. To make it effective for general-purpose encoder models, JBert applies AliBi in both directions.

Training Process

The training process has three phases: (1) training the backbone (only for V2), (2) unsupervised training on pairs, and (3) supervised training with additional hard negatives:



Training on Pairs

For training the model to produce good embeddings, we use InfoNCELoss $\mathcal{L}^{\text{pairs}}(\mathrm{B})$. It contrasts the similarity of the members of a pair to the similarity values between embeddings of texts that do not form pairs:

$$\mathcal{L}^{\text{pairs}}(\mathrm{B}) := \mathcal{L}^{\text{pairs}}_{\text{NCE}}(\mathrm{B}) + \mathcal{L}^{\text{pairs}}_{\text{NCE}}(\mathrm{B}), \text{ where}$$

$$\mathcal{L}^{\text{pairs}}_{\text{NCE}}(\mathrm{B}) := \mathbb{E}_{(\mathrm{q},\mathrm{p})\sim\mathrm{B}}\left[-\ln \frac{e^{\mathrm{s}(\mathrm{q},\mathrm{p})/\tau}}{\sum_{\mathrm{i}=1}^{\mathrm{k}} e^{\mathrm{s}(\mathrm{q},\mathrm{p}_\mathrm{i})/\tau}}\right]$$

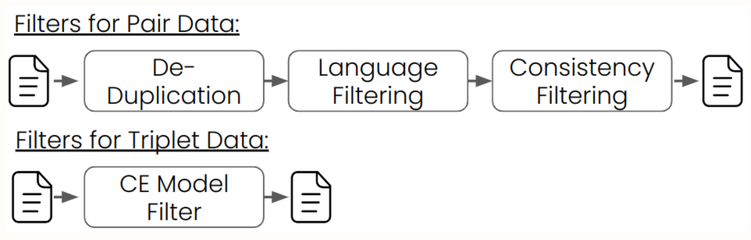
$$\mathcal{L}^{\text{pairs}}_{\text{NCE}}(\mathrm{B}) := \mathbb{E}_{(\mathrm{q},\mathrm{p})\sim\mathrm{B}}\left[-\ln \frac{e^{\mathrm{s}(\mathrm{p},\mathrm{q})/\tau}}{\sum_{\mathrm{i}=1}^{\mathrm{k}} e^{\mathrm{s}(\mathrm{p},\mathrm{q}_\mathrm{i})/\tau}}\right]$$

Training with Hard Negatives

InfoNCE loss treats all combinations of texts that do not form a pair as negatives. For this stage of training, we also use additional hard negatives.

Dataset Preparation

We train our models on a diverse ensemble of datasets, cleaned up by a pipeline of filters:



To validate our filter pipeline, we fine-tuned a small model on just one dataset, applying the pipeline's steps individually to evaluate their effect:

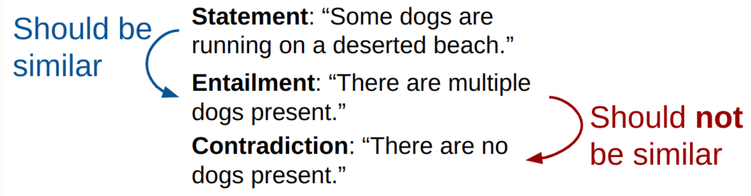
Data Preparation	Quora*	SciFact*	Trec-Cov*	STS12**
No Extra Filter	0.734	0.218	0.242	0.558
Language	0.741	0.218	0.250	0.561
Consistency	0.805	**0.381**	0.297	0.652
Language + Consistency	**0.806**	0.379	**0.306**	**0.653**

(∗) nDCG@10
(∗∗) Spearman Correlation

This demonstrates that language and consistency filtering are effective. Combining both leads to the highest performance across the majority of benchmarks.

Negation Dataset

Embedding models often treat negated statements as more similar to non-negated ones with the opposite meaning than to entailed, non-negated statements with the same meaning. To address this, we have extended some existing datasets (SNLI, Multi-NLI, sentence-compression, Simple Wikipedia, and COCO Captions) with negations generated by an LLM. This allows us to evaluate how models handle negations and to fine-tune them to improve their performance distinguishing negations from entailments.



	Error Rate [%]	
Model	Before Fine-tuning	After Fine-tuning
jina-small-v1	7.9	5.7
jina-base-v1	6.4	3.9
jina-large-v1	6.2	3.1
all-MiniLM-L6-v2	8	5
all-mpnet-base-v2	6.3	3.1
sentence-t5-base	5.3	2.9
sentence-t5-large	4.3	2.3

Evaluation Result

We evaluated our models with the MTEB benchmark, which encompasses more than 50 embedding evaluation tasks with different objectives:

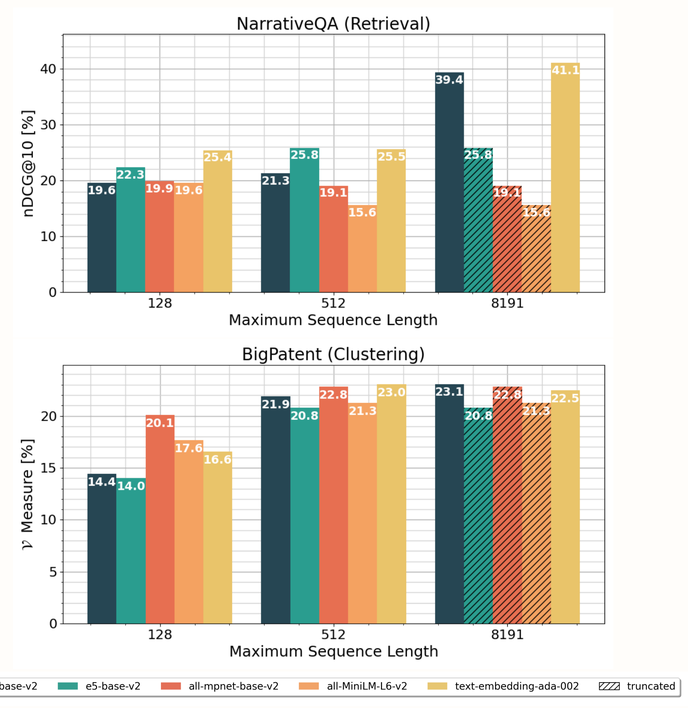
Model	CF	CL	PC	RR	RT	STS	SM	Average
text-emb-ada-002	70.93	45.9	84.89	56.32	49.25	80.97	30.8	60.99
e5-base-v2	73.84	43.8	85.73	55.91	50.29	81.05	30.28	61.5
all-MiniLM-L6-v2	63.05	42.35	82.37	58.04	41.95	78.9	30.81	56.26
all-mpnet-base-v2	65.07	43.69	83.04	59.36	43.81	80.28	27.49	57.78
jina-small-v1	60.56	32.56	79.22	53.07	38.91	78.06	31.25	52.33
jina-base-v1	66.07	35.88	83.04	55.84	44.03	79.93	30.71	56.26
jina-large-v1	67.76	37.15	84.8	56.42	44.81	80.96	29.85	57.38
jina-small-v2	68.82	40.08	84.44	55.09	45.14	80	30.56	58
jina-base-v2	73.45	41.73	85.38	56.98	47.87	80.7	31.6	60.38

CF: Classification Accuracy [%] CL: Clustering $\mathcal{V}$ measure[%]
PC: Pair Classification Average Precision [%] RR: Reranking MAP [%]
RT: Retrieval nDCG@10 STS: Sentence Similarity Spearman Corr. [%]
SM: Summarization Spearman Correlation [%]

Even though most tasks only involve short texts, these results show that our models are on par with the state-of-the-art.

Influence of the Sequence Length

To test the influence of the sequence length, we apply models to datasets with long texts after truncating the inputs at different token positions:



Further Information

Jina Embedding Models Version 1:
Günther, M., Milliken, L., Geuter, J., Mastrapas, G., Wang, B., Xiao, H. (2023). Jina Embeddings: A Novel Set of High-Performance Sentence Embedding Models. arXiv:2307.11224.
https://huggingface.co/jinaai/jina-embedding-b-en-v1
Jina Embedding Models Version 2:
Günther, M., Ong, J., Mohr, I., Abdessalem, A., Abel, T., Akram, M. K., ... Xiao, H. (2023). Jina Embeddings 2: 8192-Token General-Purpose Text Embeddings for Long Documents. arXiv:2310.19923.
https://huggingface.co/jinaai/jina-embeddings-v2-base-en
Negation Dataset:
https://huggingface.co/datasets/jinaai/negation-dataset-v2
Embedding API:
https://jina.ai/embeddings/

Huggingface page for Jina Embeddings models

References

[1] Reimers, N., Gurevych, I. (2019, November). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *EMNLP-IJCNLP 2019*.

[2] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. The JMLR, 21(1)

[3] Press, O., Smith, N., Lewis, M. (2021, October). Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation. In ICLR.

3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS) 6 Dec 2023 @ EMNLP 2023 in Singapore