Training Sparse Mixture Of Experts Text Embedding Models

Zach Nussbaum [*1] **Brandon Duderstadt** [1]

Abstract

Transformer-based text embedding models have improved their performance on benchmarks like MIRACL and BEIR by increasing their parameter counts. However, this scaling approach introduces significant deployment challenges, including increased inference latency and memory usage. These challenges are particularly severe in retrieval-augmented generation (RAG) applications, where large models' increased memory requirements constrain dataset ingestion capacity, and their higher latency directly impacts query-time performance. While causal language models have addressed similar efficiency challenges using Mixture of Experts (MoE) architectures, this approach hasn't been successfully adapted to the general text embedding setting. In this paper, we introduce Nomic Embed v2, the first general purpose MoE text embedding model. Our model outperforms models in the same parameter class on both monolingual and multilingual benchmarks while also maintaining competitive performance with models twice its size. We open-source all code, models, and evaluation data to ensure full reproducibility of our training pipeline at https://github.com/nomic-ai/contrastors.

1. Introduction

Transformer-based biencoders are the standard architecture for training dense sentence embedding models for text retrieval (Reimers & Gurevych, 2019). In the monolingual setting, these models are trained on curated internet-scale data (Wang et al., 2024a; Xiao et al., 2024; Günther et al., 2023; Nussbaum et al., 2024; Li et al., 2023), and sometimes augmented with task-specific instructions (Su et al., 2023a). While models like mE5 (Wang et al., 2024b), BGE-M3 (Chen et al., 2024), mGTE (Zhang et al., 2024), and Jina V3 (Günther et al., 2024) make strides towards a unified embedding space across languages, they underperform their parameter-equivalent monolingual counterparts on English benchmarks. Multilingual models primarily close this performance gap by increasing their parameter counts, often through the use of large, pretrained multilingual Language Models fine-tuned for retrieval applications (Jiang et al., 2023; Lee et al., 2024b).

The large size of multilingual embedding models creates significant deployment challenges. Their substantial memory requirements and increased inference latency particularly impact retrieval-augmented generation (RAG) applications, where they constrain both dataset ingestion capacity and query-time performance.

While causal language models have addressed similar efficiency challenges using Mixture of Experts (MoE) architectures, this approach has not yet been adapted for text embeddings.

In this work, we introduce the first general-purpose Mixture of Experts text embedding model. We demonstrate that scaling text embedding models with Mixture of Experts in both monolingual and multilingual settings outperforms existing approaches while using fewer active parameters.

2. Related Work

2.1. Mixture of Experts

The Mixture of Experts (MoE) architecture was first introduced by Shazeer et al. (2017) as a method to increase model capacity and performance without a proportional increase in computation by stacking sparsely gated LSTM blocks (Hochreiter & Schmidhuber, 1997). Lepikhin et al. (2020) utilized MoE layers in Transformers for machine translation and showed improvements in multilingual translation as the model size increased, while only incurring a sublinear increase in training time. Fedus et al. (2022) simplified the routing, reduced training instability, and reduced communication costs to achieve a 7x improvement in pre-training speed. Zoph et al. (2022) found that MoEs frequently experienced training instabilities, and introduced an auxiliary loss to stabilize the model training without harming its quality.

Recent advances in MoE training, such as upcycling from pretrained transformers (Komatsuzaki et al., 2023) and efficient block-sparse implementations (Gale et al., 2022), have

[1]Nomic AI, New York, NY, USA. Correspondence to: Zach Nussbaum <zach@nomic.ai>.

Under Review

Table 1. Evaluation of Multilingual Text Embedding Models

Model	Params (M)	Emb Dim	BEIR	MIRACL	Pretrain Data	Finetune Data	Code
mE5 Base	278	768	48.88	62.30	No	No	No
mGTE Base	305	768	51.10	63.40	No	No	No
Arctic Embed v2 Base	305	768	**55.40**	59.90	No	No	No
Nomic Embed v2	305	768	52.86	**65.80**	**Yes**	**Yes**	**Yes**
BGE M3	568	1024	48.80	**69.20**	No	**Yes**	No
Arctic Embed v2 Large	568	1024	**55.65**	66.00	No	No	No
mE5 Large	560	1024	51.40	66.50	No	No	No
mE5 Large Instruct	560	1024	52.64	65.70	No	No	No
Jina Embed v3	572	1024	53.88	61.20	No	No	No

made MoE training even more efficient. However, these advances have primarily focused on language modeling tasks. While Hallee et al. (2024) explored domain-specific MoE embeddings and Li & Zhou (2024) investigated using MoE language model states as embeddings, our work is the first to develop a general-purpose MoE architecture specifically for text embeddings. Concurrent work GRITLM (Muennighoff et al., 2024) demonstrates that MoE models like Mixtral 8x7B can effectively handle both embedding and generation tasks through instruction tuning. In contrast, our work focuses on optimizing MoE architectures for embedding efficiency through large-scale contrastive pretraining and finetuning.

2.2. Monolingual Text Embeddings

Modern monolingual text embedders typically follow a two-stage approach: contrastive pretraining on large weakly-supervised datasets, followed by contrastive finetuning on human-labeled data (Wang et al., 2022; Li et al., 2023; Günther et al., 2023; Nussbaum et al., 2024). Recent work has focused on scaling and data curation (Xiao et al., 2023; Wang et al., 2022; Li et al., 2023; Günther et al., 2023; Nussbaum et al., 2024; Merrick et al., 2024; Yu et al., 2024) or adapting decoder-only LLMs for embedding tasks (Wang et al., 2023; Lee et al., 2024b).

2.3. Multilingual Text Embeddings

While multilingual encoders like mBert (Devlin et al., 2019) and XLM-Roberta (Conneau et al., 2020) provide a foundation for cross-lingual representation, they require additional training for high-quality sentence embeddings. Current approaches either rely on translation data (Reimers & Gurevych, 2020) or scale up model size (Wang et al., 2024b; Chen et al., 2024), typically requiring 3-5x more parameters than monolingual models to achieve comparable English performance - a phenomenon known as the "curse of multilinguality."

Recent work like Arctic Embed 2.0 (Yu et al., 2024) demonstrates that multilingual models can achieve strong English performance without compromising multilingual capability. However, existing approaches still face fundamental challenges with efficiency: state-of-the-art models require large parameter counts and generate large embedding vectors, increasing both computational and economic costs of dense retrieval.

Our MoE-based approach directly addresses this efficiency challenge, maintaining strong performance across both English and multilingual tasks while significantly reducing the active parameter count during inference. This represents a fundamental shift from previous scaling approaches that relied solely on increasing dense model capacity.

Table 2. MLM Hyperparameters

Hyperparameter	**Value**
Batch Size	4,096
Peak Learning Rate	4e-4
Warmup Steps	500
Total Steps	10,000
Grad. Accumulation Steps	8
Learning Rate Schedule	Linear
Sequence Length	2,048
Rotary Base	10,000
MLM Probability	0.3
Language Sampling α	0.3
Max Grad Norm	1.0

3. Background

3.1. Masked Language Modeling

Masked language modeling (MLM), a self-supervised pre-training objective introduced by Devlin et al. (2019), trains a model to recover masked tokens from input sequences. MLM was applied to both monolingual and multilingual

Table 3. Hyperparameters used for finetuning all models on GLUE benchmark tasks. For mGTE, warmup percentage is set to 6% and max gradient norm to 1.

					Single Sentence		Paraphrase and Similarity			Natural Language Inference		
Model	**Params**	**Pos.**	**Seq.**	**Avg.**	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE
XLM-R-Base	279M	Abs.	512	82.35	46.95	92.54	87.37	89.32	90.69	84.34	90.35	77.26
mNomic-BERT	279M	RoPE	2048	81.63	44.69	91.97	87.50	88.48	90.93	83.59	89.38	76.54
mGTE-Base	306M	RoPE	8192	80.77	27.22	91.97	89.71	89.55	91.20	85.16	90.91	80.41

datasets resulting in BERT and mBERT, with the latter demonstrating the potential of cross-lingual representation learning. However, Conneau et al. (2020) identified that these models were undertrained and introduced XLM-RoBERTa, which achieved performance comparable to monolingual models by training on CC100, a diverse dataset spanning 100 languages from CommonCrawl.

3.2. Mixture of Experts (MoE)

Dense models activate all parameters for every input. In contrast, Sparse Mixture of Experts (MoE) models activate only a subset of parameters for each input, reducing computational requirements while maintaining model capacity (Shazeer et al., 2017).

In MoE architectures, standard MLP layers are replaced with MoE blocks consisting of multiple "expert" networks and a router. The router dynamically assigns each input token to a subset of experts using Top-K routing: the router outputs logits for all experts, applies softmax normalization, and routes each token to the top_k experts with the highest probabilities (Fedus et al., 2022).

A key challenge in training MoE models is expert collapse, where certain experts receive disproportionate traffic and others remain underutilized. This is typically addressed through an auxiliary load balancing loss (Zoph et al., 2022):

$$\mathcal{L}_{balance} = \alpha \sum_{i=1}^{E} (r_i \cdot p_i) \tag{1}$$

where r_i is the fraction of tokens routed to expert i and p_i is the mean routing probability for that expert across a batch of tokens. The coefficient α controls the strength of the balancing loss relative to the main objective.

3.3. Contrastive Learning

3.3.1. Training Text Embedding Models

Text embedding models are generally trained in two stages: weakly-supervised contrastive pretraining and contrastive finetuning (Reimers & Gurevych, 2019).

The contrastive pretraining stage uses the InfoNCE objective (van den Oord et al., 2019) to train a biencoder to distinguish relevant text pairs from irrelevant pairs. Given a batch $B = (q_0, d_0), (q_1, d_1)...(q_n, d_n)$, the objective is:

$$\mathcal{L}_C = -\frac{1}{n} \sum_i \log \frac{e^{s(q_i,d_i)/\tau}}{e^{s(q_i,d_i)/\tau} + \sum_{j \neq i}^{n} e^{s(q_i,d_j)/\tau}} \tag{2}$$

where $s(q, d)$ is the learned score between query q and document d and τ is the temperature. Contrastive finetuning incorporates high-quality human labeled datasets and hard negatives to improve retrieval performance (Wang et al., 2022). The InfoNCE objective is adapted to include these hard negatives:

$$Z_i = e^{s(q_i,d_i)/\tau} + \sum_{j \neq i}^{n} e^{s(q_i,d_j)/\tau} + \sum_{m=1}^{H} e^{s(q_i,d_{hn}(1,m))/\tau} \tag{3}$$

$$\mathcal{L}_C = -\frac{1}{n} \sum_i \log \frac{e^{s(q_i,d_i)/\tau}}{Z_i} \tag{4}$$

To reduce the storage costs of embedding vectors, which scale with embedding dimension, recent works have applied Matryoshka Representation Learning (Kusupati et al., 2024) during both training stages (Lee et al., 2024c). This enables more efficient storage of the computed embeddings by encouraging a rank ordering over the information content of successive embedding subspaces

3.3.2. Consistency Filtering

Consistency filtering improves dataset quality by removing potential false positives from weakly supervised data (Wang et al., 2022). In this approach, each dataset is divided into shards of 1-3M samples. An existing text embedding model first embeds all queries and documents. Query-document pairs are then discarded if a ground truth document does not appear among the top-k most similar documents to query.

Initially developed for English text embeddings (Günther et al., 2024; Nussbaum et al., 2024), consistency filtering has been adapted for multilingual data by Yu et al. (2024) using multilingual-E5-small (Wang et al., 2024b) with 3M samples per shard and a top-20 filtering threshold.

Table 4. XTREME-R Benchmark

Model	Avg.	XNLI Acc.	XCOPA Acc.	UDPOS F1	WikiANN F1	XQuAD F1	MLQA F1	TyDiQA-GoldP F1	Mewsli-X mAP@20	LAReQA mAP@20	Tatoeba Acc.
XLM-R-Base	62.31	74.49	51.8	74.33	60.99	72.96	61.45	54.31	42.45	63.49	66.79
mNomic-BERT	62.70	73.57	61.71	74.92	60.96	71.13	59.61	43.46	43.27	67.49	70.82
mGTE-Base	64.63	73.58	63.62	73.52	60.72	74.71	63.88	49.68	44.58	71.90	70.07

3.3.3. Hard Negative Mining

Text embedding models are typically finetuned with hard negatives mined by an existing retriever (Nussbaum et al., 2024; Yu et al., 2024). While traditional approaches use the top-k most similar documents as hard negatives, this can introduce false negatives. To address this, Moreira et al. (2024) introduced positive-aware hard negative mining:

$$threshold = pos\_sim * percentage\_margin \quad (5)$$

where $percentage\_margin$ (typically 95%) creates a threshold below which negatives are accepted, reducing false negatives. Recent work has shown that using stronger teacher models for mining yields higher quality finetuning datasets (Moreira et al., 2024; Yu et al., 2024).

4. Methods

4.1. Adapting XLM-Roberta for Long-Context

To extend document-level capabilities to multilingual settings, we modify XLM-Roberta Base (Conneau et al., 2020) to handle longer sequences as XLM-Roberta's absolute positional encodings restrict inputs to 512 tokens.

Following Gumma et al. (2024), we replace the absolute positional encodings with Rotary Positional Embeddings (RoPE) (Su et al., 2023b). We set the RoPE base parameter to 10,000, enabling the model to extrapolate to longer sequences while maintaining stable performance. While recent work (Liu et al., 2024; Xiong et al., 2023) suggests using larger RoPE bases, our experiments showed degraded performance on GLUE and XTREME-R benchmarks with larger values. This difference might stem from our training approach – unlike Zhang et al. (2024), who first train with shorter sequences (2,048 tokens) before scaling up, we maintain consistent sequence lengths throughout training.

We use 2048-token segments from a reconstructed CC100 dataset [1]. Following the original XLM-Roberta training protocol, we set the language sampling temperature to 0.3. We train for 10,000 steps with hyperparameters detailed in Table 2.

[1] https://huggingface.co/datasets/statmt/cc100

We refer to our adapted model as mNomic-BERT.

4.2. Consistency Filtering

To ensure high-quality training data, we implement retrieval-based consistency filtering on our multilingual corpus consisting of data from mC4 and multilingual CC News. This approach, established in recent work (Yu et al., 2024; Nussbaum et al., 2024), helps eliminate low-quality or misaligned text pairs from the training set.

For each language in our corpus, we divide the dataset into segments of 1 million examples. Using the multilingual E5 small embedding model (Wang et al., 2024b), we compute similarity between query-document pairs. We retain only pairs where the document ranks among the top 2 most similar documents for its corresponding query, following similar filtering approaches in (Wang et al., 2022; Günther et al., 2023). For English-language data, we utilize the pre-filtered dataset from Nussbaum et al. (2024).

This filtering process yields a final training dataset of 1.6 billion high-quality pairs. The distribution of data across different languages is detailed in Appendix A.

4.3. Weakly-Supervised Contrastive Pretraining

For our contrastive pretraining phase, we initialize a bi-encoder with mNomic-BERT and train it on our filtered contrastive dataset for one epoch. Following Komatsuzaki et al. (2023), we transform every alternate MLP layer into an MoE layer with 8 experts and top-2 routing, starting from the second layer. This results in a model with 475M total parameters, of which only 305M are active during inference. We set the load balancing loss coefficient α from Equation 1 to 1.

For training, we use the InfoNCE contrastive loss (van den Oord et al., 2019) with a temperature of $\tau = 0.02$. Following recent work (Nussbaum et al., 2024; Merrick et al., 2024), we process one dataset per batch with a batch size of 16,384, using random batch sampling. Similar to Yu et al. (2024), we set maximum sequence lengths of 32 and 256 tokens for queries and documents respectively due to computational constraints.

We train the model using 16 H100 GPUs with distributed data-parallel training and activation checkpointing. Our optimization uses a peak learning rate of 8e-5 with 1,000 warmup steps and cosine decay.

4.4. Hard Negative Mining

For each query in our dataset, we mine hard negatives using a margin-based approach defined in Equation 5. We use the data from Chen et al. (2024) and BGE M3 for filtering both English and multilingual data.

4.5. Contrastive Finetuning

We finetune the pretrained biencoder from Section 4.3 using our mined hard negatives. For each query, we incorporate 10 hard negative examples during training. We train for one epoch using a batch size of 256, with a peak learning rate of 2e-5, 400 warmup steps, and linear decay. Compared to pretraining, we increase both query and document maximum lengths to 512 tokens.

To enable efficient inference at multiple dimensions, we incorporate Matryoshka Representation Learning (Kusupati et al., 2024), training the model to produce effective embeddings at dimensions 768 and 256. The distribution of our finetuning data is detailed in Appendix C.

We refer to this final model as Nomic Embed v2.

5. Experimental Setup

5.1. GLUE Evaluation Protocol

We evaluate mNomic-BERT on the GLUE benchmark (Wang et al., 2019), following the evaluation protocol from Nussbaum et al. (2024). We train each model on 8 GLUE tasks for 3 epochs across 5 random seeds, varying batch sizes (16, 32) and learning rates (1e-5, 2e-5, 3e-5). For mGTE evaluation, we modify these parameters to use 6% warmup and max gradient norm of 1, matching Zhang et al. (2024). Following standard practice (Liu et al., 2019), we initialize RTE, STSB, and MRPC tasks from an MNLI checkpoint. Table 6 details the complete hyperparameter configuration.

5.2. XTREME-R Evaluation Setup

We evaluate mNomic-BERT on XTREME-R (Ruder et al., 2021), a comprehensive benchmark consisting of 10 tasks designed to assess multilingual natural language understanding capabilities. All experiments follow a zero-shot cross-lingual transfer protocol: models are trained exclusively on English data and evaluated on multilingual and cross-lingual tasks. We utilize the evaluation pipeline from Zhang et al. (2024)[2] to ensure fair comparison with baseline models XLM-R-Base and mGTE-Base.

[2] `https://github.com/izhx/nlu-evals`

5.3. Text Embedding Benchmark Setup

We evaluate our model on two retrieval benchmarks: (1) BEIR, the retrieval subset of MTEB (Muennighoff et al., 2023), which focuses on English-only retrieval, and (2) MIRACL (Zhang et al., 2022), which evaluates multilingual retrieval capabilities. For all experiments, we:

- Prepend task-specific prefixes “search_query” and “search_document” to queries and documents
- Truncate all inputs to 512 tokens
- Measure performance using nDCG@10

For reproducibility, we conduct all evaluations using the FlagEmbedding framework[3], except for mE5 results which are taken directly from Wang et al. (2024b). Note that mE5 results for German (de) and Yoruba (yo) languages were not reported in the original paper.

6. Results

6.1. mNomic-BERT GLUE Results

Our approach achieves strong performance across the GLUE benchmark, as shown in Table 3. Specifically, mNomic-BERT achieves comparable performance to XLM-R-Base across all tasks, demonstrating that our RoPE-based positional encoding modification and lightweight finetuning preserve the model’s capabilities. Notably, mNomic-BERT matches mGTE-Base performance while requiring only 3% of mGTE-Base’s pretraining steps, suggesting that our lightweight finetuning approach effectively extends context length without extensive pretraining.

While Zhang et al. (2024) reported lower CoLA scores for XLM-Roberta, our hyperparameter search revealed that this task is particularly sensitive to configuration choices. We successfully reproduced mGTE-Base’s reported CoLA performance but found significant variance across different hyperparameter settings, resulting in a lower median score.

6.2. XTREME-R Results

Table 4 presents the performance of mNomic-BERT compared to XLM-R-Base and mGTE-Base across XTREME-R tasks. mNomic-BERT achieves an average score of 62.70, which is comparable to XLM-R-Base’s 62.31 but falls slightly behind mGTE-Base’s 64.63. This pattern is consistent across most individual tasks, with mNomic-BERT and XLM-R-Base showing similar performance levels. These results suggest that our approach maintains the cross-lingual capabilities of the base architecture while extending the con-

[3] https://github.com/FlagOpen/FlagEmbedding

Table 5. MIRACL Performance Across Different Languages. Numbers for E5 taken from Wang et al. (2024b).

Model	Avg (18)	Avg (16)	NDCG@10 Per Language																	
			ar	bn	de	en	es	fa	fi	fr	hi	id	ja	ko	ru	sw	te	th	yo	zh
Arctic M v2.0	60.6	59.9	69.7	67.7	**56.7**	**55.7**	55.4	52.6	68.4	54.0	53.7	48.3	58.3	59.7	58.8	52.3	81.7	74.3	75.6	48.3
mGTE Base	63.6	63.4	71.4	72.9	49.7	54.0	51.8	54.0	73.5	54.5	51.9	50.3	65.8	62.9	63.2	69.9	**83.1**	74.0	79.3	**61.8**
mE5 Base	62.2	62.3	71.6	70.2	51.9	51.2	51.5	57.4	74.4	49.7	58.4	51.1	64.7	62.2	61.5	**71.1**	75.2	75.2	70.7	51.5
Nomic Embed v2	**66.0**	**65.8**	**76.7**	**73.6**	56.6	54.7	**56.3**	**59.2**	**77.1**	**55.8**	**60.5**	**54.2**	**67.0**	**65.9**	**65.2**	66.3	82.6	**78.3**	**78.3**	59.5
Arctic L v2.0	66.3	66.0	76.1	74.4	**58.6**	53.7	55.6	60.3	77.1	56.7	58.4	52.3	66.5	66.3	67.1	70.8	83.5	77.5	78.3	59.9
mE5 Large	66.6	66.5	76.0	75.9	56.4	52.9	52.9	59.0	77.8	54.5	**62.0**	52.9	70.6	66.5	67.4	74.9	**84.6**	80.2	78.3	56.0
E5 Large Instr.	66.1	65.7	76.8	73.9	55.7	51.5	53.7	59.4	77.3	53.7	60.3	52.1	69.0	65.3	67.9	72.5	83.4	78.6	81.6	56.2
BGE M3	**69.2**	**69.2**	**78.5**	**79.9**	56.8	**56.9**	56.1	**60.9**	**78.6**	**58.2**	59.5	**56.0**	**72.8**	**69.6**	**70.1**	**78.6**	**86.2**	**82.6**	**81.8**	**62.6**

Table 6. GLUE Fintuning Hyperparameters

Hyperparameter	**Value**
Epochs	3
Sequence Length	128
Batch Size	16, 32
Learning Rate	1, 2, 3e-5
Learning Rate Schedule	Linear
Warmup Pct	0
Max Grad Norm	0

text length of multilingual text encoders, complementing recent work by Gumma et al. (2024).

6.3. Text Embedding Benchmark

We evaluate performance on BEIR, the retrieval subset of MTEB (Muennighoff et al., 2023), an English-only benchmark, and MIRACL (Zhang et al., 2022), a multilingual retrieval benchmark. Results can be found in Table 1 and 5.

Compared to similarly sized parameter models, Nomic Embed v2 outperforms all models on BEIR and MIRACL except Arctic Embed v2 Base. However, Yu et al. (2024) do not release any of their training data of which a large percentage consists of private web search data.

Despite being 2x smaller, Nomic Embed v2 outperforms all multilingual models on BEIR, except Arctic Embed v2 Large, and is competitive with all models on MIRACL.

7. Analysis

7.1. Effectiveness of MoEs for Text Embeddings

We compare monolingual MoE and dense text embedding models by pretraining them on 235M weakly-supervised contrastive pairs from Nussbaum et al. (2024). For evaluation, we use the BEIR benchmark (Thakur et al., 2021) across varying batch sizes, with a fixed maximum sequence length of 128 tokens. Our MoE model (Nomic BERT MoE) is created by upcycling alternate layers of Nomic BERT following Komatsuzaki et al. (2023). The model uses token choice routing with TopK Routing ($\mathbf{k} = \mathbf{1}$, also known as Switch Routing (Fedus et al., 2022)) and 8 experts. We compare this against two baselines: the original Nomic BERT and BERT Large (Devlin et al., 2019).

Figure 1 shows that Nomic BERT MoE consistently outperforms the original Nomic BERT across all batch sizes, despite maintaining a similar number of active parameters. Notably, our MoE model achieves comparable performance to BERT Large, despite the latter having 3x more active parameters, demonstrating the efficiency of the MoE architecture.

Table 7. **Impact of Upcycled Layers on Model Performance.** BEIR scores across batch sizes and upcycled layers. 6-layer models outperform 12-layer variants at larger batches, suggesting selective upcycling is more effective than full model conversion.

MoE Layers	Batch Size	BEIR
6	2048	44.13
	4096	**45.36**
	8192	**45.89**
12	2048	44.28
	4096	44.89
	8192	45.48

Table 7 presents an ablation study on the number of upcycled layers. Converting all 12 layers to MoE layers actually reduces performance compared to converting only 6 layers, particularly at larger batch sizes. This suggests that selective layer upcycling provides a better balance between model capacity and optimization stability.

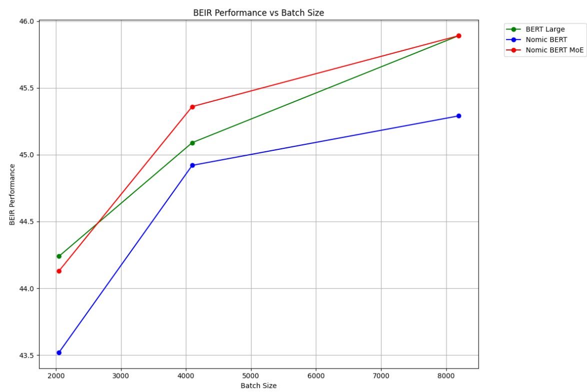

Figure 1. **Impact of Model Size and Batch Size on Retrieval Performance**. NDCG@10 scores on BEIR benchmark across different batch sizes and model architectures. The upcycled MoE model's performance approaches that of a model with 3x more active parameters as batch size increases, demonstrating efficient scaling behavior.

7.2. Effectiveness of MoEs for Multilingual Text Embeddings

We extend our analysis to the multilingual setting by incorporating an additional 65M weakly-supervised contrastive pairs from mC4 (Xue et al., 2021) and Multilingual CC News (Wang et al., 2024b). For a controlled ablation study, we focus on six languages spanning different language families: English, Chinese, Arabic, Hindi, Spanish, and Swahili. This selection includes both high-resource and low-resource languages, with Swahili representing the latter category. We evaluate three models: XLM-RoBERTa Base (Conneau et al., 2020), our MoE variant (XLM-RoBERTa MoE Base), and XLM-RoBERTa Large. Performance is measured using NDCG@10 on both BEIR (Thakur et al., 2021) and MIRACL (Zhang et al., 2022) benchmarks across different batch sizes.

Table 9 presents our multilingual evaluation results. While our MoE model consistently outperforms its dense counterpart across all batch sizes on both BEIR and MIRACL benchmarks, it does not match the performance of the larger model—a notable departure from our monolingual findings.

Our experiments reveal that data scale significantly impacts the performance of XLM-RoBERTa MoE Base. Initial experiments with a smaller dataset of 100M total contrastive pairs showed the MoE model consistently underperforming its parameter-equivalent dense counterpart. This aligns with findings from Krajewski et al. (2024), who observed that MoE models tend to underperform dense models under limited training regimes.

Table 8. Evaluation of different teacher models and thresholds for hard negative mining

Teacher Model	Margin	Negatives	Avg	FiQA	HotpotQA	NQ
Arctic Embed Large	None	4	52.87	42.68	59.47	56.45
Arctic Embed Large	0.95	4	55.20	44.98	62.75	57.88
NVEmbed v1	0.95	4	54.94	45.00	58.29	61.56
Stella 1.5B	0.95	4	57.22	45.18	64.06	62.42
Stella 1.5B	0.98	4	57.20	45.67	63.65	62.29
Stella 1.5B	0.95	7	57.31	45.10	64.27	62.58
Stella 1.5B	0.95	10	57.45	45.17	64.48	62.69

7.3. Hard Negative Mining

We investigate the impact of different teacher models and margin thresholds for hard negative mining, following the approach of Moreira et al. (2024). We initialize our model from E5-Large Unsupervised (Wang et al., 2022)[4] and mine negatives using Equation 5. Our training data comprises approximately 500k examples from three sources: StackExchange Title-Body pairs[5], SQuAD (Rajpurkar et al., 2016), and Natural Questions (NQ) (Kwiatkowski et al., 2019). We evaluate performance on three BEIR datasets: NQ, FiQA, and HotpotQA (Thakur et al., 2021). For teacher models, we compare NVEmbed v1 (Lee et al., 2024a), Arctic Embed Large (Merrick et al., 2024), and Stella 1.5B v5 (Zhang & FulongWang, 2024).

Table 8 presents our findings across different teacher models and mining parameters. Several key trends emerge: Positive aware hard negative mining with consistently improves performance, as shown by the 2.33 point average improvement when using Arctic Embed Large with a margin of 0.95 compared to no margin. Surprisingly, Stella 1.5B outperforms NVEmbed v1 even though it is a 7x smaller model. Increasing the number of negative examples from 4 to 10 with Stella 1.5B yields modest but consistent improvements, with the best average performance of 57.45 achieved using 10 negatives. However, the gains diminish with each additional negative, suggesting a potential plateau in the benefits of increased negative examples. Finally, varying the margin threshold between 0.95 and 0.98 shows minimal impact on overall performance, indicating that the mining process is relatively robust to this hyperparameter within this range.

We also compared our best-performing mined dataset against a filtered version of the finetuning data released by Chen et al. (2024). Using BGE M3 to filter negatives based on Equation 5, this approach achieved 1 point higher NDCG@10 on BEIR, suggesting filtering potential negatives from an existing mined dataset is also a viable option.

[4] https://huggingface.co/intfloat/e5-large-unsupervised

[5] https://huggingface.co/datasets/sentence-transformers/embedding-training-data

Table 9. **Performance Comparison of Multilingual Models.** BEIR and MIRACL scores across different model architectures and batch sizes. XLM-R Large (561M parameters) consistently outperforms both the MoE variants and the base model (XLM-B, 278M parameters). MoE models show improved performance with increased batch sizes, particularly when using k=2 experts.

Model	Params	BEIR	MIRACL
Batch Size: 2048			
XLM-R Large	561M	**45.86**	38.49
XLM-MoE (k=1)	278M	43.63	**39.17**
XLM-B	278M	43.51	34.14
Batch Size: 4096			
XLM-R Large	561M	**46.26**	**38.99**
XLM-MoE (k=1)	278M	44.37	37.08
XLM-B	278M	43.78	37.32
Batch Size: 8192			
XLM-R Large	561M	**46.91**	**42.71**
XLM-MoE (k=2)	300M	45.00	39.81
XLM-MoE (k=1)	278M	44.11	39.17
XLM-B	278M	43.96	37.92

8. Conclusion

We introduce Nomic Embed v2, the first Mixture of Expert Embedding Model. Nomic Embed v2 outperforms similarly sized and larger embedding models in both English and Multilingual Retrieval benchmarks while being trained only publicly available data. Nomic Embed v2 proves a successful alternative to scaling text embedding models without increasing computational costs.

9. Limitations and Future Work

Our work with Nomic Embed v2 demonstrates the advantages of MoE architectures over dense models for text embeddings. However, this represents only an initial exploration of MoE applications in this domain. Several promising research directions emerge: investigating the optimal scaling of expert count and active parameters, exploring alternative routing mechanisms, and examining how loss-free routing could leverage the bidirectional nature of these models. Furthermore, techniques for distilling MoE models back into dense architectures could make these improvements more widely deployable.

Beyond architectural choices, understanding the fundamental scaling relationships between dataset size, model parameters, and embedding dimension would provide valuable insights for the field. This could help establish whether the benefits of MoE architectures persist or even compound at larger scales.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Chen, J., Xiao, S., Zhang, P., Luo, K., Lian, D., and Liu, Z. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2024. URL https://arxiv.org/abs/2402.03216.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. Unsupervised cross-lingual representation learning at scale, 2020. URL https://arxiv.org/abs/1911.02116.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Fedus, W., Zoph, B., and Shazeer, N. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, 2022. URL https://arxiv.org/abs/2101.03961.

Gale, T., Narayanan, D., Young, C., and Zaharia, M. Megablocks: Efficient sparse training with mixture-of-experts, 2022. URL https://arxiv.org/abs/2211.15841.

Gumma, V., Chitale, P. A., and Bali, K. Towards inducing document-level abilities in standard multilingual neural machine translation models, 2024. URL https://arxiv.org/abs/2408.11382.

Günther, M., Milliken, L., Geuter, J., Mastrapas, G., Wang, B., and Xiao, H. Jina embeddings: A novel set of high-performance sentence embedding models, 2023.

Günther, M., Ong, J., Mohr, I., Abdessalem, A., Abel, T., Akram, M. K., Guzman, S., Mastrapas, G., Sturua, S., Wang, B., Werk, M., Wang, N., and Xiao, H. Jina embeddings 2: 8192-token general-purpose text embeddings for long documents, 2024.

Hallee, L., Kapur, R., Patel, A., Gleghorn, J. P., and Khomtchouk, B. Contrastive learning and mixture of experts enables precise vector embeddings, 2024. URL https://arxiv.org/abs/2401.15713.

Hochreiter, S. and Schmidhuber, J. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735.

URL https://doi.org/10.1162/neco.1997.9.8.1735.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. Mistral 7b, 2023.

Komatsuzaki, A., Puigcerver, J., Lee-Thorp, J., Ruiz, C. R., Mustafa, B., Ainslie, J., Tay, Y., Dehghani, M., and Houlsby, N. Sparse upcycling: Training mixture-of-experts from dense checkpoints, 2023. URL https://arxiv.org/abs/2212.05055.

Krajewski, J., Ludziejewski, J., Adamczewski, K., Pióro, M., Krutul, M., Antoniak, S., Ciebiera, K., Król, K., Odrzygóźdź, T., Sankowski, P., Cygan, M., and Jaszczur, S. Scaling laws for fine-grained mixture of experts, 2024. URL https://arxiv.org/abs/2402.07871.

Kusupati, A., Bhatt, G., Rege, A., Wallingford, M., Sinha, A., Ramanujan, V., Howard-Snyder, W., Chen, K., Kakade, S., Jain, P., and Farhadi, A. Matryoshka representation learning, 2024. URL https://arxiv.org/abs/2205.13147.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Kelcey, M., Devlin, J., Lee, K., Toutanova, K. N., Jones, L., Chang, M.-W., Dai, A., Uszkoreit, J., Le, Q., and Petrov, S. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

Lee, C., Roy, R., Xu, M., Raiman, J., Shoeybi, M., Catanzaro, B., and Ping, W. Nv-embed: Improved techniques for training llms as generalist embedding models. *arXiv preprint arXiv:2405.17428*, 2024a.

Lee, C., Roy, R., Xu, M., Raiman, J., Shoeybi, M., Catanzaro, B., and Ping, W. Nv-embed: Improved techniques for training llms as generalist embedding models, 2024b. URL https://arxiv.org/abs/2405.17428.

Lee, J., Dai, Z., Ren, X., Chen, B., Cer, D., Cole, J. R., Hui, K., Boratko, M., Kapadia, R., Ding, W., Luan, Y., Duddu, S. M. K., Abrego, G. H., Shi, W., Gupta, N., Kusupati, A., Jain, P., Jonnalagadda, S. R., Chang, M.-W., and Naim, I. Gecko: Versatile text embeddings distilled from large language models, 2024c. URL https://arxiv.org/abs/2403.20327.

Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., Krikun, M., Shazeer, N., and Chen, Z. Gshard: Scaling giant models with conditional computation and automatic sharding, 2020. URL https://arxiv.org/abs/2006.16668.

Li, Z. and Zhou, T. Your mixture-of-experts llm is secretly an embedding model for free, 2024. URL https://arxiv.org/abs/2410.10814.

Li, Z., Zhang, X., Zhang, Y., Long, D., Xie, P., and Zhang, M. Towards general text embeddings with multi-stage contrastive learning, 2023.

Liu, X., Yan, H., Zhang, S., An, C., Qiu, X., and Lin, D. Scaling laws of rope-based extrapolation, 2024. URL https://arxiv.org/abs/2310.05209.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach, 2019.

Merrick, L., Xu, D., Nuti, G., and Campos, D. Arctic-embed: Scalable, efficient, and accurate text embedding models, 2024. URL https://arxiv.org/abs/2405.05374.

Moreira, G. d. S. P., Osmulski, R., Xu, M., Ak, R., Schifferer, B., and Oldridge, E. Nv-retriever: Improving text embedding models with effective hard-negative mining. *arXiv preprint arXiv:2407.15831*, 2024.

Muennighoff, N., Tazi, N., Magne, L., and Reimers, N. Mteb: Massive text embedding benchmark, 2023.

Muennighoff, N., Su, H., Wang, L., Yang, N., Wei, F., Yu, T., Singh, A., and Kiela, D. Generative representational instruction tuning, 2024. URL https://arxiv.org/abs/2402.09906.

Nussbaum, Z., Morris, J. X., Duderstadt, B., and Mulyar, A. Nomic embed: Training a reproducible long context text embedder, 2024. URL https://arxiv.org/abs/2402.01613.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. SQuAD: 100,000+ questions for machine comprehension of text. In Su, J., Duh, K., and Carreras, X. (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264.

Reimers, N. and Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.

Reimers, N. and Gurevych, I. Making monolingual sentence embeddings multilingual using knowledge distillation, 2020. URL https://arxiv.org/abs/2004.09813.

Ruder, S., Constant, N., Botha, J., Siddhant, A., Firat, O., Fu, J., Liu, P., Hu, J., Garrette, D., Neubig, G., and Johnson, M. Xtreme-r: Towards more challenging and nuanced multilingual evaluation, 2021. URL https://arxiv.org/abs/2104.07412.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017. URL https://arxiv.org/abs/1701.06538.

Su, H., Shi, W., Kasai, J., Wang, Y., Hu, Y., Ostendorf, M., tau Yih, W., Smith, N. A., Zettlemoyer, L., and Yu, T. One embedder, any task: Instruction-finetuned text embeddings, 2023a.

Su, J., Lu, Y., Pan, S., Murtadha, A., Wen, B., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding, 2023b.

Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., and Gurevych, I. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models, 2021.

van den Oord, A., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding, 2019.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. GLUE: A multi-task benchmark and analysis platform for natural language understanding. 2019. In the Proceedings of ICLR.

Wang, L., Yang, N., Huang, X., Jiao, B., Yang, L., Jiang, D., Majumder, R., and Wei, F. Text embeddings by weakly-supervised contrastive pre-training, 2022.

Wang, L., Yang, N., Huang, X., Yang, L., Majumder, R., and Wei, F. Improving text embeddings with large language models, 2023.

Wang, L., Yang, N., Huang, X., Jiao, B., Yang, L., Jiang, D., Majumder, R., and Wei, F. Text embeddings by weakly-supervised contrastive pre-training, 2024a. URL https://arxiv.org/abs/2212.03533.

Wang, L., Yang, N., Huang, X., Yang, L., Majumder, R., and Wei, F. Multilingual e5 text embeddings: A technical report, 2024b. URL https://arxiv.org/abs/2402.05672.

Xiao, S., Liu, Z., Zhang, P., and Muennighoff, N. C-pack: Packaged resources to advance general chinese embedding, 2023.

Xiao, S., Liu, Z., Zhang, P., Muennighoff, N., Lian, D., and Nie, J.-Y. C-pack: Packed resources for general chinese embeddings, 2024. URL https://arxiv.org/abs/2309.07597.

Xiong, W., Liu, J., Molybog, I., Zhang, H., Bhargava, P., Hou, R., Martin, L., Rungta, R., Sankararaman, K. A., Oguz, B., Khabsa, M., Fang, H., Mehdad, Y., Narang, S., Malik, K., Fan, A., Bhosale, S., Edunov, S., Lewis, M., Wang, S., and Ma, H. Effective long-context scaling of foundation models, 2023. URL https://arxiv.org/abs/2309.16039.

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., and Raffel, C. mt5: A massively multilingual pre-trained text-to-text transformer, 2021. URL https://arxiv.org/abs/2010.11934.

Yu, P., Merrick, L., Nuti, G., and Campos, D. Arctic-embed 2.0: Multilingual retrieval without compromise, 2024. URL https://arxiv.org/abs/2412.04506.

Zhang, D. and FulongWang. Jasper and stella: distillation of sota embedding models, 2024. URL https://arxiv.org/abs/2412.19048.

Zhang, X., Thakur, N., Ogundepo, O., Kamalloo, E., Alfonso-Hermelo, D., Li, X., Liu, Q., Rezagholizadeh, M., and Lin, J. Making a miracl: Multilingual information retrieval across a continuum of languages, 2022. URL https://arxiv.org/abs/2210.09984.

Zhang, X., Zhang, Y., Long, D., Xie, W., Dai, Z., Tang, J., Lin, H., Yang, B., Xie, P., Huang, F., Zhang, M., Li, W., and Zhang, M. mgte: Generalized long-context text representation and reranking models for multilingual text retrieval, 2024. URL https://arxiv.org/abs/2407.19669.

Zoph, B., Bello, I., Kumar, S., Du, N., Huang, Y., Dean, J., Shazeer, N., and Fedus, W. St-moe: Designing stable and transferable sparse expert models, 2022. URL https://arxiv.org/abs/2202.08906.

A. Weakly Supervised Contrastive Pretraining Dataset Distribution

The full pretraining dataset distribution can be see in Table A.

Table 10. Dataset Distribution of 1.6B pairs for weakly supervised contrastive pretraining

Code	Language	Pairs	Code	Language	Pairs	Code	Language	Pairs
en	English	234,553,344	be	Belarusian	589,824	my	Burmese	147,456
es	Spanish	210,010,112	ml	Malayalam	557,056	km	Khmer	131,072
fr	French	172,769,280	kn	Kannada	524,288	mg	Malagasy	131,072
de	German	169,426,944	mk	Macedonian	425,984	pa	Punjabi	131,072
it	Italian	104,251,392	ur	Urdu	409,600	ru-Latn	Russian (Latin)	131,072
pt	Portuguese	87,982,080	fy	Frisian	393,216	sn	Shona	131,072
pl	Polish	63,209,472	fil	Filipino	360,448	zh-Latn	Chinese (Latin)	131,072
nl	Dutch	50,118,656	te	Telugu	360,448	ha	Hausa	98,304
tr	Turkish	49,053,696	eu	Basque	344,064	he	Hebrew	98,304
ja	Japanese	43,433,984	sw	Swahili	327,680	hmn	Hmong	98,304
vi	Vietnamese	40,058,880	so	Somali	294,912	ht	Haitian	98,304
ru	Russian	38,731,776	sd	Sindhi	262,144	ja-Latn	Japanese (Latin)	98,304
id	Indonesian	36,470,784	uz	Uzbek	262,144	su	Sundanese	98,304
ar	Arabic	33,800,192	co	Corsican	245,760	bg-Latn	Bulgarian (Latin)	65,536
cs	Czech	29,966,336	hr	Croatian	245,760	gd	Scots Gaelic	65,536
ro	Romanian	24,772,608	gu	Gujarati	229,376	ny	Nyanja	65,536
sv	Swedish	24,608,768	hi-Latn	Hindi (Latin)	229,376	ps	Pashto	65,536
el	Greek	22,773,760	ceb	Cebuano	196,608	ku	Kurdish	49,152
uk	Ukrainian	19,841,024	eo	Esperanto	196,608	sh	Serbo-Croatian	49,152
zh	Chinese	18,661,376	jv	Javanese	196,608	am	Amharic	32,768
hu	Hungarian	18,448,384	la	Latin	196,608	ig	Igbo	32,768
da	Danish	14,548,992	zu	Zulu	196,608	lo	Lao	32,768
no	Norwegian	12,812,288	mn	Mongolian	180,224	mi	Maori	32,768
hi	Hindi	12,713,984	si	Sinhala	180,224	nn	Norwegian Nynorsk	32,768
fi	Finnish	12,697,600	el-Latn	Greek (Latin)	163,840	sm	Samoan	32,768
bg	Bulgarian	12,042,240	ga	Irish	163,840	yi	Yiddish	32,768
ko	Korean	10,354,688	ky	Kyrgyz	163,840	st	Sotho	16,384
sk	Slovak	8,962,048	tg	Tajik	163,840	tl	Tagalog	16,384
th	Thai	7,602,176				xh	Xhosa	16,384
iw	Hebrew	5,783,552				yo	Yoruba	16,384
ca	Catalan	5,701,632						
lt	Lithuanian	5,242,880						
fa	Persian	5,177,344						
ms	Malay	4,325,376						
sl	Slovenian	4,259,840						
lv	Latvian	3,211,264						
mr	Marathi	2,588,672						
bn	Bengali	2,457,600						
sq	Albanian	2,113,536						
cy	Welsh	2,048,000						

B. Contrastive Finetuning Dataset Distribution

Full finetuning data distribution can be found in Table C. We train on the training sets of BEIR and MIRACL as well as SQuAD and Stackoverflow.

C. BEIR Retrieval Performance

The full BEIR results broken down by task can be found in Table C. Nomic Embed v2 at 256 dimensions performs competitively to full dimensionality.

Table 11. Contrastive finetuning data distribution

Dataset	**Number of Samples**
English Datasets	
MSMARCO	485,120
Stack	249,856
SQuAD	87,552
HotPot	82,432
NQ	57,856
FEVER	28,672
MIRACL Train Datasets	
Russian	4,608
Indonesian	3,840
Arabic	3,328
Japanese	3,328
Telugu	3,328
Finnish	2,816
Thai	2,816
English	2,560
Spanish	2,048
Persian	2,048
Swahili	1,792
Bengali	1,536
Chinese	1,280
French	1,024
Hindi	1,024
Korean	768
Total	**1,029,632**

Table 12. BEIR Retrieval Performance

Dataset	Nomic Embed v2	Nomic Embed v2 256
Average	52.86	49.63
ArguAna	55.73	51.97
ClimateFEVER	33.38	29.58
CQADupstack	42.64	40.25
DBPedia	41.44	37.51
FEVER	87.17	84.64
FiQA2018	38.74	35.71
HotpotQA	68.53	63.67
MSMARCO	40.89	38.90
NFCorpus	34.60	31.29
NQ	60.41	56.12
QuoraRetrieval	87.95	87.49
SCIDOCS	19.25	17.71
SciFact	72.89	66.31
TRECCOVID	78.78	74.22
Touche2020	30.55	29.03