Improving Text Embeddings with Large Language Models

Liang Wang, Nan Yang, Xiaolong Huang,
Linjun Yang, **Rangan Majumder**, **Furu Wei**
Microsoft Corporation
{wangliang,nanya,xiaolhu,yang.linjun,ranganm,fuwei}@microsoft.com

Abstract

In this paper, we introduce a novel and simple method for obtaining high-quality text embeddings using only synthetic data and less than 1k training steps. Unlike existing methods that often depend on multi-stage intermediate pre-training with billions of weakly-supervised text pairs, followed by fine-tuning with a few labeled datasets, our method does not require building complex training pipelines or relying on manually collected datasets that are often constrained by task diversity and language coverage. We leverage proprietary LLMs to generate diverse synthetic data for hundreds of thousands of text embedding tasks across 93 languages. We then fine-tune open-source decoder-only LLMs on the synthetic data using standard contrastive loss. Experiments demonstrate that our method achieves strong performance on highly competitive text embedding benchmarks without using any labeled data. Furthermore, when fine-tuned with a mixture of synthetic and labeled data, our model sets new state-of-the-art results on the BEIR and MTEB benchmarks.

1 Introduction

Text embeddings are vector representations of natural language that encode its semantic information. They are widely used in various natural language processing (NLP) tasks, such as information retrieval (IR), question answering, semantic textual similarity, bitext mining, item recommendation, etc. In the field of IR, the first-stage retrieval often relies on text embeddings to efficiently recall a small set of candidate documents from a large-scale corpus using approximate nearest neighbor search techniques. Embedding-based retrieval is also a crucial component of retrieval-augmented generation (RAG) (Lewis et al., 2020), which is an emerging paradigm that enables large language models (LLMs) to access dynamic external knowledge without modifying the model parameters. Source

attribution of generated text is another important application of text embeddings (Gao et al., 2023) that can improve the interpretability and trustworthiness of LLMs.

Previous studies have demonstrated that weighted average of pre-trained word embeddings (Pennington et al., 2014; Arora et al., 2017) is a strong baseline for measuring semantic similarity. However, these methods fail to capture the rich contextual information of natural language. With the advent of pre-trained language models (Devlin et al., 2019), Sentence-BERT (Reimers and Gurevych, 2019) and SimCSE (Gao et al., 2021) have been proposed to learn text embeddings by fine-tuning BERT on natural language inference (NLI) datasets. To further enhance the performance and robustness of text embeddings, state-of-the-art methods like E5 (Wang et al., 2022b) and BGE (Xiao et al., 2023) employ a more complex multi-stage training paradigm that first pre-trains on billions of weakly-supervised text pairs, and then fine-tunes on several high-quality labeled datasets.

Existing multi-stage approaches suffer from several drawbacks. Firstly, they entail a complex multi-stage training pipeline that demands substantial engineering efforts to curate large amounts of relevance pairs. Secondly, they rely on manually collected datasets that are often constrained by the diversity of tasks and the coverage of languages. For instance, Instructor (Su et al., 2023) is only trained on instructions from 330 English datasets, whereas BGE (Xiao et al., 2023) only focuses on high-resource languages such as English and Chinese. Moreover, most existing methods employ BERT-style encoders as the backbone, neglecting the recent advances of training better LLMs and related techniques such as context length extension (Rozière et al., 2023).

In this paper, we propose a novel method for text embeddings that leverages LLMs to overcome the

limitations of existing approaches. We use proprietary LLMs to generate synthetic data for a diverse range of text embedding tasks in 93 languages, covering hundreds of thousands of embedding tasks. Specifically, we use a two-step prompting strategy that first prompts the LLMs to brainstorm a pool of candidate tasks, and then prompts the LLMs to generate data conditioned on a given task from the pool. To cover various application scenarios, we design multiple prompt templates for each task type and combine the generated data from different templates to boost diversity. For the text embedding models, we opt for fine-tuning powerful open-source LLMs rather than small BERT-style models. Since LLMs such as Mistral (Jiang et al., 2023) have been extensively pre-trained on web-scale data, contrastive pre-training that proves to be important for BERT models (Wang et al., 2022b) offers little additional benefit.

We demonstrate that Mistral-7B, when fine-tuned solely on synthetic data, attains competitive performance on the BEIR (Thakur et al., 2021) and MTEB (Muennighoff et al., 2023) benchmarks. This is particularly intriguing considering that this setting does not involve any labeled data. When fine-tuned on a mixture of synthetic and labeled data, our model achieves new state-of-the-art results, surpassing previous methods by a significant margin (+2%). The entire training process requires less than 1k steps.

Moreover, we empirically validate that our model can effectively perform personalized passkey retrieval for inputs up to 32k tokens by altering the rotation base of the position embeddings, extending the context length beyond the conventional 512 token limit. Regarding its multilinguality, our model excels on high-resource languages. However, for low-resource languages, there is still room for improvement as current open-source LLMs are not adequately pre-trained on them.

2 Related Work

Text Embeddings are continuous low-dimensional representations of text and have been extensively applied to various downstream tasks such as information retrieval, question answering, and retrieval-augmented generation (RAG). Early work on text embeddings includes latent semantic indexing (Deerwester et al., 1990) and weighted average of word embeddings (Mikolov et al.,

2013). More recent methods exploit supervision from natural language inference (Bowman et al., 2015) and labeled query-document pairs, such as the MS-MARCO passage ranking dataset (Campos et al., 2016), to train text embeddings (Reimers and Gurevych, 2019; Conneau et al., 2017; Gao et al., 2021). However, labeled data are often limited in terms of task diversity and language coverage. To address this challenge, methods like Contriever (Izacard et al., 2021), OpenAI Embeddings (Neelakantan et al., 2022), E5 (Wang et al., 2022b), and BGE (Xiao et al., 2023) adopt a multi-stage training paradigm. They first pre-train on large-scale weakly-supervised text pairs using contrastive loss and then fine-tune on small-scale but high-quality datasets. In this paper, we demonstrate that it is possible to obtain state-of-the-art text embeddings with single-stage training.

Synthetic Data Synthetic data generation is a widely studied topic in information retrieval research, with various methods proposed to enhance retrieval systems with artificially created data. For instance, Doc2query (Nogueira et al., 2019), InPars (Bonifacio et al., 2022), and Promptagator (Dai et al., 2022) generate synthetic queries for unlabeled documents, which are then leveraged for document expansion or model training. GPL (Wang et al., 2022a) employs a cross-encoder to produce pseudo-labels for query-document pairs. Similarly, Query2doc (Wang et al., 2023) generates pseudo-documents for query expansion by few-shot prompting LLMs. Unlike these methods, our approach does not rely on any unlabeled documents or queries and thus can generate more diverse synthetic data.

Another related line of work focuses on knowledge distillation from black-box LLMs by training on synthetic data generated from them. DINO (Schick and Schütze, 2021) generates synthetic text pairs for semantic textual similarity. Unnatural Instructions (Honovich et al., 2022) is a synthetic instruction following dataset by prompting existing LLMs. Orca (Mukherjee et al., 2023) and Phi (Gunasekar et al., 2023) propose to train better small language models by using high-quality synthetic data from GPT-3.5/4 (OpenAI, 2023).

Large Language Models With the popularization of ChatGPT, large language models (LLMs) have demonstrated remarkable capabilities in in-

Brainstorm a list of potentially useful text retrieval tasks.
Here are a few examples for your reference:
  - Provided a scientific claim as query, retrieve documents that help verify or refute the claim.
  - Search for documents that answers a FAQ-style query on children's nutrition.
Please adhere to the following guidelines:
  - Specify what the query is, and what the desired documents are.
  - Each retrieval task should cover a wide range of queries, and should not be too specific.
Your output should always be a python list of strings only, with about 20 elements, and each element corresponds to a distinct retrieval task in one sentence. Do not explain yourself or output anything else. Be creative!

["Retrieve company's financial reports for a given stock ticker symbol.",
"Given a book name as a query, retrieve reviews, ratings and summaries of that book.",
"Search for scientific research papers supporting a medical diagnosis for a specified disease."
... (omitted for space)]

---------------------- new session ----------------------

You have been assigned a retrieval task: *{task}*
Your mission is to write one text retrieval example for this task in JSON format. The JSON object must contain the following keys:
  - **"user_query"**: a string, a random user search query specified by the retrieval task.
  - **"positive_document"**: a string, a relevant document for the user query.
  - **"hard_negative_document"**: a string, a hard negative document that only appears relevant to the query.
Please adhere to the following guidelines:
  - The "user_query" should be *{query_type}*, *{query_length}*, *{clarity}*, and diverse in topic.
  - All documents should be at least *{num_words}* words long.
  - Both the query and documents should be in *{language}*.
  ... (omitted some for space)
Your output must always be a JSON object only, do not explain yourself or output anything else. Be creative!

{**"user_query":** "How to use Microsoft Power BI for data analysis",
"positive_document": "Microsoft Power BI is a sophisticated tool that requires time and practice to master. In this tutorial, we'll show you how to navigate Power BI ... *(omitted)* ",
"hard_negative_document": "Excel is an incredibly powerful tool for managing and analyzing large amounts of data. Our tutorial series focuses on how you...*(omitted)*" }
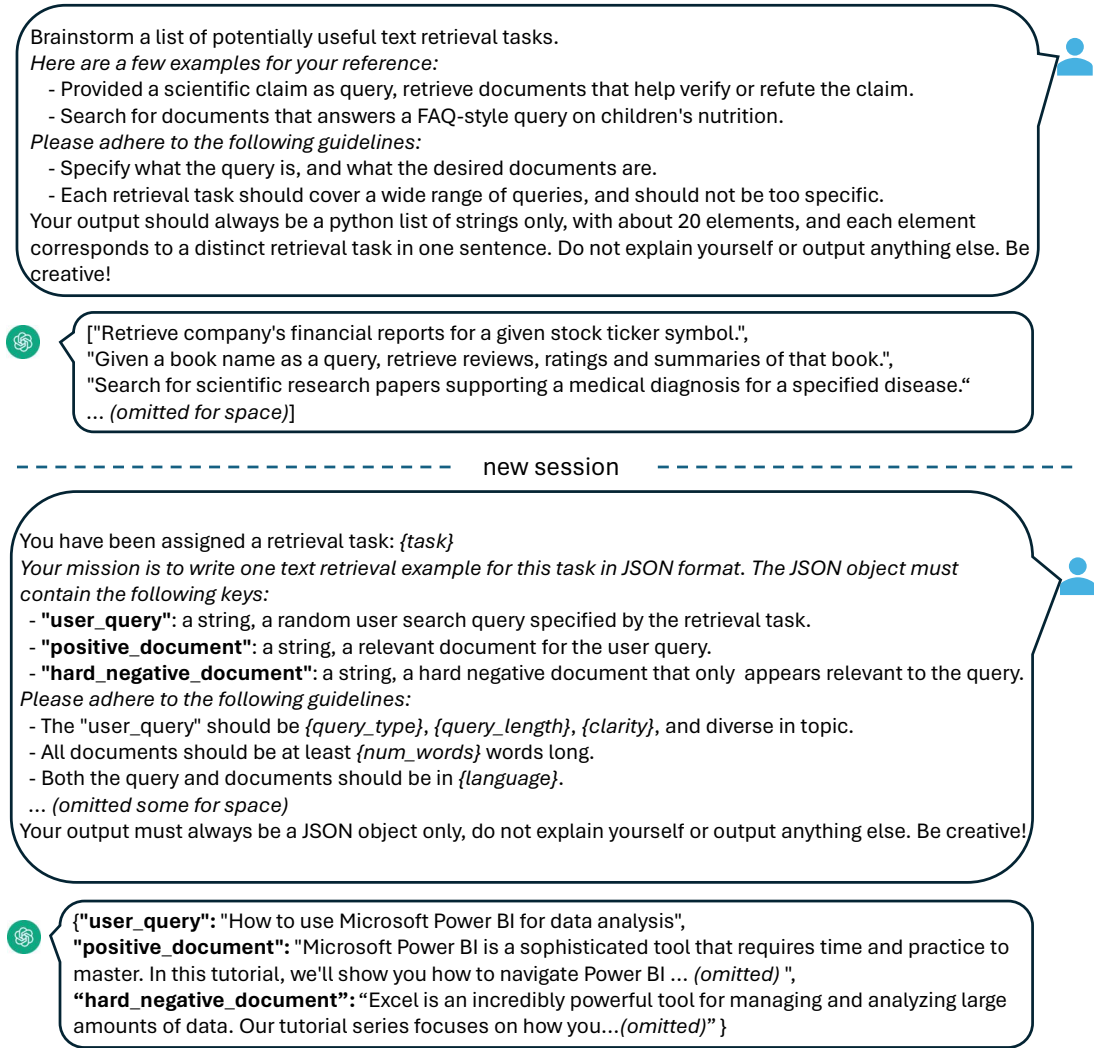
Figure 1: An example two-step prompt template for generating synthetic data with GPT-4. We first prompt GPT-4 to brainstorm a list of potential retrieval tasks, and then generate *(query, positive, hard negative)* triplets for each task. "*{...}*" denotes a placeholder that will be replaced by sampling from a predefined set of values. Full prompts are available in Appendix C.

struction following and few-shot in-context learning (Brown et al., 2020). However, the most advanced LLMs such as GPT-4 (OpenAI, 2023) are proprietary and have little technical details disclosed. To bridge the gap between proprietary and open-source LLMs, several notable efforts have been made, such as LLaMA-2 (Touvron et al., 2023) and Mistral (Jiang et al., 2023) models. A major limitation of LLMs is that they lack awareness of recent events and private knowledge. This issue can be partly mitigated by augmenting LLMs with information retrieved from external sources, a technique known as retrieval-augmented generation (RAG). On the other hand, LLMs can also serve as foundation models to enhance text embeddings. RepLLaMA (Ma et al., 2023) proposes to

fine-tune LLaMA-2 with bi-encoder architecture for ad-hoc retrieval. SGPT (Muennighoff, 2022), GTR (Ni et al., 2022b), and Udever (Zhang et al., 2023a) demonstrate the scaling law of text embeddings empirically, but their performance still falls behind small bidirectional encoders such as E5 (Wang et al., 2022b) and BGE (Xiao et al., 2023). In this paper, we present a novel approach to train state-of-the-art text embeddings by exploiting the latest advances of LLMs and synthetic data.

3 Method

3.1 Synthetic Data Generation

Utilizing synthetic data generated by advanced LLMs such as GPT-4 presents a compelling opportunity, especially in terms of enhancing diversity

across a multitude of tasks and languages. Such diversity is essential for developing robust text embeddings that can perform well across different tasks, be it semantic retrieval, textual similarity, or clustering.

To generate diverse synthetic data, we propose a simple taxonomy that categorizes embedding tasks into several groups, and then apply different prompt templates to each group.

Asymmetric Tasks This category comprises tasks where the query and document are semantically related but are not paraphrases of each other. Depending on the length of the query and document, we further divide asymmetric tasks into four subgroups: short-long match, long-short match, short-short match, and long-long match. For instance, short-long match tasks involve a short query and a long document, which is a typical scenario in commercial search engines. For each subgroup, we design a two-step prompt template that first prompts LLMs brainstorm a list of tasks, and then generates a concrete example conditioned on the task definition. In Figure 1, we show an example prompt for the short-long match subgroup. The full output is available in Table 16. The outputs from GPT-4 are mostly coherent and of high quality. In our preliminary experiments, we also attempted to generate the task definition and query-document pairs using a single prompt, but the data diversity was not as satisfactory as the proposed two-step approach.

Symmetric Tasks Symmetric tasks involve queries and documents that have similar semantic meanings but different surface forms. We examine two application scenarios: monolingual semantic textual similarity (STS) and bitext retrieval. We design two distinct prompt templates for each scenario, tailored to their specific objectives. Since the task definition is straightforward, we omit the brainstorming step for symmetric tasks.

To further boost the diversity of the prompts and thus the synthetic data, we incorporate several placeholders in each prompt template, whose values are randomly sampled at runtime. For example, in Figure 1, the value of "*{query_length}*" is sampled from the set "*{less than 5 words, 5-10 words, at least 10 words}*".

To generate multilingual data, we sample the value of "*{language}*" from the language list of

XLM-R (Conneau et al., 2020), giving more weight to high-resource languages. Any generated data that does not conform to the predefined JSON format are discarded during the parsing process. We also remove duplicates based on exact string matching.

3.2 Training

Given a relevant query-document pair (q^+, d^+), we first apply the following instruction template to the original query q^+ to generate a new one q^+_{inst}:

$$q^+_{\text{inst}} = \text{Instruct: \{task\_definition\}} \backslash n \text{ Query: } \{q^+\} \tag{1}$$

where "*{task_definition}*" is a placeholder for a one-sentence description of the embedding task. For generated synthetic data, we use the outputs from the brainstorming step. For other datasets, such as MS-MARCO, we manually craft the task definitions and apply them to all the queries in the dataset. We do not modify the document side with any instruction prefix. In this way, the document index can be prebuilt, and we can customize the task to perform by changing only the query side.

Given a pretrained LLM, we append an [EOS] token to the end of the query and document, and then feed them into the LLM to obtain the query and document embeddings $(\mathbf{h}_{q^+_{\text{inst}}}, \mathbf{h}_{d^+})$ by taking the last layer [EOS] vector. To train the embedding model, we adopt the standard InfoNCE loss $\mathbb{L}$ over the in-batch negatives and hard negatives:

$$\min \quad \mathbb{L} = -\log \frac{\phi(q^+_{\text{inst}}, d^+)}{\phi(q^+_{\text{inst}}, d^+) + \sum_{n_i \in \mathbb{N}} (\phi(q^+_{\text{inst}}, n_i))} \tag{2}$$

where $\mathbb{N}$ denotes the set of all negatives, and $\phi(q, d)$ is a function that computes the matching score between query q and document d. In this paper, we adopt the temperature-scaled cosine similarity function as follows:

$$\phi(q, d) = \exp(\frac{1}{\tau} \cos(\mathbf{h}_q, \mathbf{h}_d)) \tag{3}$$

τ is a temperature hyper-parameter, which is fixed to 0.02 in our experiments.

4 Experiments

4.1 Statistics of the Synthetic Data

Figure 2 presents the statistics of our generated synthetic data. We manage to generate 500k examples with 150k unique instructions using Azure
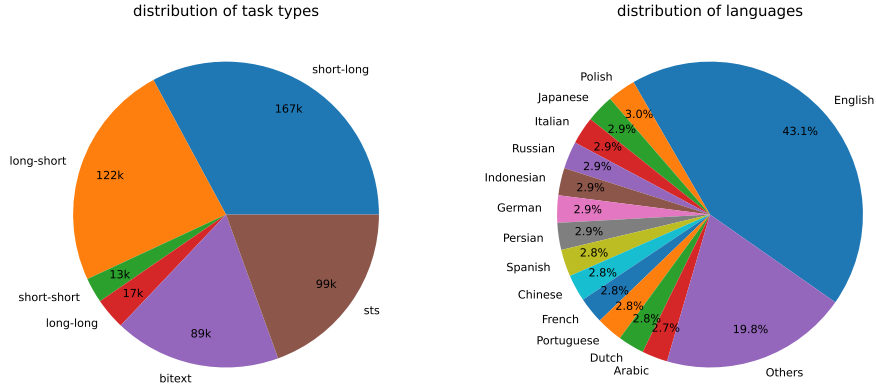
Figure 2: Task type and language statistics of the generated synthetic data (see Section 3.1 for task type definitions). The "Others" category contains the remaining languages from the XLM-R language list.

# of datasets →	Class. 12	Clust. 11	PairClass. 3	Rerank 4	Retr. 15	STS 10	Summ. 1	Avg 56
Unsupervised Models								
Glove (Pennington et al., 2014)	57.3	27.7	70.9	43.3	21.6	61.9	28.9	42.0
SimCSE_bert-unsup (Gao et al., 2021)	62.5	29.0	70.3	46.5	20.3	74.3	31.2	45.5
Supervised Models								
SimCSE_bert-sup (Gao et al., 2021)	67.3	33.4	73.7	47.5	21.8	79.1	23.3	48.7
Contriever (Izacard et al., 2021)	66.7	41.1	82.5	53.1	41.9	76.5	30.4	56.0
GTR_xxl (Ni et al., 2022b)	67.4	42.4	86.1	56.7	48.5	78.4	30.6	59.0
Sentence-T5_xxl (Ni et al., 2022a)	73.4	43.7	85.1	56.4	42.2	82.6	30.1	59.5
E5_large-v2 (Wang et al., 2022b)	75.2	44.5	86.0	56.6	50.6	82.1	30.2	62.3
GTE_large (Li et al., 2023)	73.3	46.8	85.0	59.1	52.2	83.4	31.7	63.1
BGE_large-en-v1.5 (Xiao et al., 2023)	76.0	46.1	87.1	60.0	54.3	83.1	31.6	64.2
Ours								
E5_mistral-7b + full data	**78.5**	50.3	**88.3**	**60.2**	**56.9**	**84.6**	31.4	**66.6**
w/ synthetic data only	78.2	**50.5**	86.0	59.0	46.9	81.2	31.9	63.1
w/ synthetic + msmarco	78.3	49.9	87.1	59.5	52.2	81.2	**32.7**	64.5

Table 1: Results on the MTEB benchmark (Muennighoff et al., 2023) (56 datasets in the English subset). The numbers are averaged for each category. Please refer to Table 17 for the scores per dataset.

OpenAI Service [1], among which 25% are generated by *GPT-35-Turbo* and others are generated by *GPT-4*. The total token consumption is about 180M. The predominant language is English, with coverage extending to a total of 93 languages. For the bottom 75 low-resource languages, there are about 1k examples per language on average. Please see Table 16 in the appendix for examples of synthetic data.

In terms of data quality, we find that a portion of *GPT-35-Turbo* outputs do not strictly follow the guidelines specified in the prompt templates. Nevertheless, the overall quality remains acceptable, and preliminary experiments have demonstrated the benefits of incorporating this data subset.

---

[1] https://oai.azure.com/

4.2 Model Fine-tuning and Evaluation

The pretrained Mistral-7b (Jiang et al., 2023) checkpoint is fine-tuned for 1 epoch using the loss in Equation 2. We follow the training recipe from RankLLaMA (Ma et al., 2023) and utilize LoRA (Hu et al., 2022) with rank 16. To further reduce GPU memory requirement, techniques including gradient checkpointing, mixed precision training, and DeepSpeed ZeRO-3 are applied.

For the training data, we utilize both the generated synthetic data and a collection of 13 public datasets, yielding approximately 1.8M examples after sampling. More details are available in Appendix A. To provide a fair comparison with some previous work, we also report results when the only labeled supervision is the MS-MARCO passage

	High-resource Languages				Low-resource Languages			
	en	fr	es	ru	te	hi	bn	sw
BM25 (Zhang et al., 2023b)	35.1	18.3	31.9	33.4	49.4	45.8	50.8	38.3
mDPR (Zhang et al., 2023b)	39.4	43.5	47.8	40.7	35.6	38.3	44.3	29.9
mE5$_{base}$ (Wang et al., 2024)	51.2	49.7	51.5	61.5	75.2	58.4	70.2	71.1
mE5$_{large}$ (Wang et al., 2024)	52.9	54.5	**52.9**	67.4	**84.6**	**62.0**	**75.9**	**74.9**
E5$_{mistral-7b}$ + full data	**57.3**	**55.2**	52.2	**67.7**	73.9	52.1	70.3	68.4

Table 2: nDCG@10 on the dev set of the MIRACL dataset for both high-resource and low-resource languages. We select the 4 high-resource languages and the 4 low-resource languages according to the number of candidate documents. The numbers for BM25 and mDPR come from Zhang et al. (2023b). For the complete results on all 16 languages, please see Table 6.
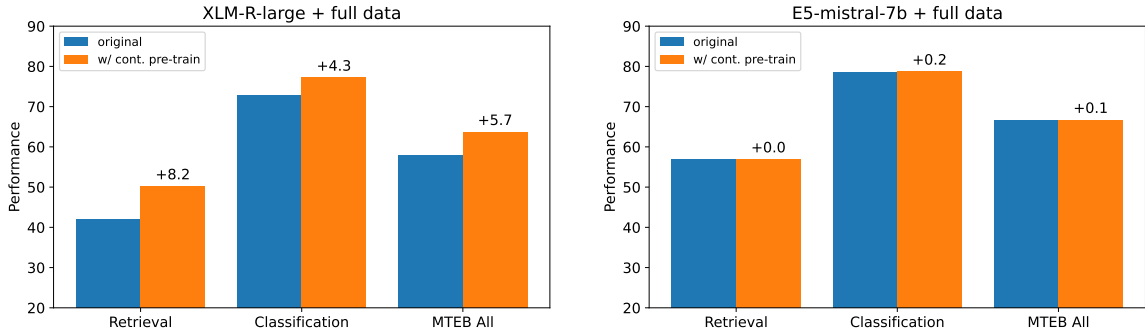
Figure 3: Effects of contrastive pre-training. Detailed numbers are in Appendix Table 7.

ranking (Campos et al., 2016) dataset.

We evaluate the trained model on the MTEB benchmark (Muennighoff et al., 2023). Note that the retrieval category in MTEB corresponds to the 15 publicly available datasets in the BEIR benchmark (Thakur et al., 2021). Evaluation of one model takes about 3 days on 8 V100 GPUs due to the need to encode a large number of documents. Although our model can accommodate sequence length beyond 512, we only evaluate on the first 512 tokens for efficiency. Official metrics are reported for each category. For more details about the evaluation protocol, please refer to the original papers (Muennighoff et al., 2023; Thakur et al., 2021).

4.3 Main Results

In Table 1, our model "E5$_{mistral-7b}$ + full data" attains the highest average score on the MTEB benchmark, outperforming the previous state-of-the-art model by 2.4 points. In the "w/ synthetic data only" setting, no labeled data is used for training, and yet the performance remains quite competitive. We posit that generative language modeling and text embeddings are the two sides of the same coin, with both tasks requiring the model to have a deep understanding of the natural language. Given

an embedding task definition, a truly robust LLM should be able to generate training data on its own and then be transformed into an embedding model through light-weight fine-tuning. Our experiments shed light on the potential of this direction, and more research is needed to fully explore it.

Model	BEIR	MTEB
OpenAI text-embedding-3-large	55.4	64.6
Cohere-embed-english-v3.0	55.0	64.5
voyage-lite-01-instruct	55.6	64.5
UAE-Large-V1	54.7	64.6
E5$_{mistral-7b}$ + full data	**56.9**	**66.6**

Table 3: Comparison with commercial models and the model that tops the MTEB leaderboard (as of 2023-12-22) (Li and Li, 2023). "BEIR" is the average nDCG@10 score over 15 public datasets in the BEIR benchmark (Thakur et al., 2021). "MTEB" is the average score over 56 datasets in the English subset of the MTEB benchmark (Muennighoff et al., 2023). For the commercial models listed here, little details are available on their model architectures and training data.

In Table 3, we also present a comparison with several commercial text embedding models. However, due to the lack of transparency and documentation about these models, a fair comparison is not feasible. We focus especially on the retrieval per-

Query: what is the pass key for Malayah Graves?

Doc1: <prefix filler> Malayah Graves's pass key is 123. Remember it. 123 is the pass key for Malayah Graves. <suffix filler>
Doc2: <prefix filler> Cesar McLean's pass key is 456. Remember it. 456 is the pass key for Cesar McLean. <suffix filler>
......

Figure 4: Illustration of the *personalized passkey retrieval* task adapted from Mohtashami and Jaggi (2023). The "*<prefix filler>*" and "*<suffix filler>*" are repeats of "*The grass is green. The sky is blue. The sun is yellow. Here we go. There and back again.*" In addition, each document has a unique person name and a random passkey inserted at a random position. The task is to retrieve the document that contains the given person's passkey from 100 candidates.
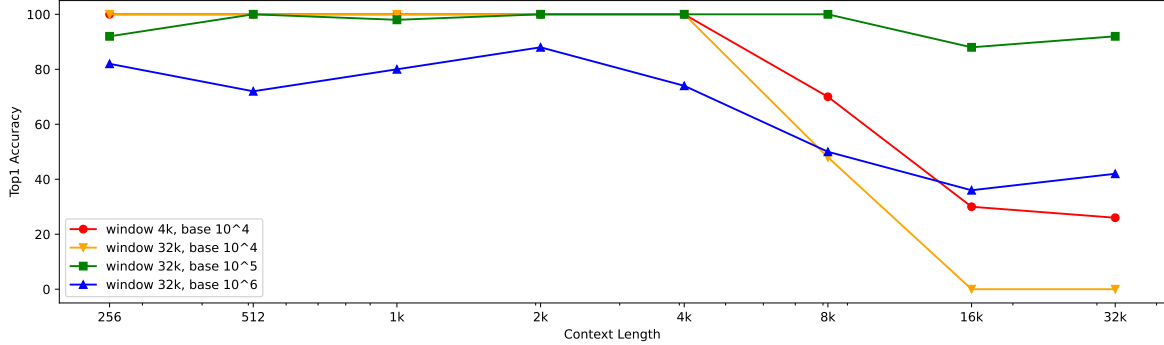
Figure 5: Accuracy of personalized passkey retrieval as a function of input context length. For each context length, we randomly generate 50 queries and compute the top-1 accuracy.

formance on the BEIR benchmark, since retrieval-augmented generation is an emerging technique to enhance LLM with external knowledge and proprietary data. As Table 3 shows, our model outperforms the current commercial models by a significant margin.

4.4 Multilingual Retrieval

To assess the multilingual capabilities of our model, we conduct an evaluation on the MIRACL dataset (Zhang et al., 2023b), which comprises human-annotated queries and relevance judgments across 18 languages. The validation set contains labels for 16 languages. As shown in Table 2, our model surpasses mE5$_{large}$ on high-resource languages, notably on English. Nevertheless, for low-resource languages, our model remains suboptimal compared to mE5$_{base}$. We attribute this to the fact that Mistral-7B is predominantly pre-trained on English data, and we anticipate that future multilingual LLMs will leverage our method to bridge this gap.

To evaluate our model's cross-lingual retrieval capability, we report Bitext mining results in Table 4. For baselines including mContriever (Izacard et al., 2021), LaBSE (Feng et al., 2022), and mE5 (Wang et al., 2024), we evaluate the results using publicly available checkpoints. Our observa-

	BUCC 2018	Tatoeba
	4 langs	112 langs
mContriever	93.7	37.7
LaBSE	98.8	**81.1**
mE5$_{base}$	98.1	68.1
mE5$_{large}$	98.6	75.7
E5$_{mistral-7b}$	**98.9**	70.1

Table 4: Bitext mining results. BUCC 2018 (Zweigenbaum et al., 2018) contains 4 high-resource languages. Tatoeba (Artetxe and Schwenk, 2019) consists of 112 English-centric language pairs.

tions indicate that, similar to the MIRACL retrieval, E5$_{mistral-7b}$ excels in bitext mining for high-resource languages only.

5 Analysis

5.1 Is Contrastive Pre-training Necessary?

Weakly-supervised contrastive pre-training is one of the key factors behind the success of existing text embedding models. For instance, Contriever (Izacard et al., 2021) treats random cropped spans as positive pairs for pre-training, while E5 (Wang et al., 2022b) and BGE (Xiao et al., 2023) collect and filter text pairs from various sources.

This section re-evaluates the necessity of con-

Datasets	Class.	Clust.	PairClass.	Rerank	Retr.	STS	Summ.	Avg
E5$_{mistral-7b}$	78.3	49.9	87.1	**59.5**	52.2	81.2	**32.7**	64.5
w/ LLaMA-2 7b init.	76.2	48.1	85.1	58.9	49.6	81.2	30.8	62.9$^{-1.6}$
w/ msmarco data only	71.6	47.1	86.1	58.8	54.4	79.5	31.7	62.7$^{-1.8}$
pooling type								
w/ mean pool	77.0	48.9	86.1	59.2	52.4	**81.4**	30.8	64.1$^{-0.4}$
w/ weighted mean	77.0	49.0	86.1	59.2	52.0	**81.4**	30.2	64.0$^{-0.5}$
LoRA rank								
w/ r=8	**78.4**	**50.3**	87.1	59.3	**53.0**	81.0	31.7	**64.8**$^{+0.3}$
w/ r=32	**78.4**	**50.3**	**87.4**	**59.5**	52.2	81.2	30.6	64.6$^{+0.1}$
instruction type								
w/o instruction	72.3	47.1	82.6	56.3	48.2	76.7	30.7	60.3$^{-4.2}$
w/ task type prefix	71.1	46.5	79.7	54.0	52.7	73.8	30.0	60.3$^{-4.2}$

Table 5: Results on the MTEB benchmark with various hyperparameters. The first row corresponds to the default setting, which employs last-token pooling, LoRA rank 16, and natural language instructions. Unless otherwise stated, all models are trained on the synthetic and MS-MARCO passage ranking data.

trastive pre-training for LLMs, particularly those that have been pre-trained on trillions of tokens. Figure 3 shows that contrastive pre-training benefits XLM-R$_{large}$, enhancing its retrieval performance by 8.2 points when fine-tuned on the same data, which aligns with prior findings. However, for Mistral-7B based models, contrastive pre-training has negligible impact on the model quality. This implies that extensive auto-regressive pre-training enables LLMs to acquire good text representations, and only minimal fine-tuning is required to transform them into effective embedding models.

5.2 Extending to Long Text Embeddings

Existing evaluation datasets for text embedding models are typically short, to evaluate the long-context capability of our model, we introduce a novel synthetic task called *personalized passkey retrieval*, which is illustrated in Figure 4. This task requires encoding the passkey information in a long context into the embeddings. We compare the performance of different variants by changing the sliding window size and the RoPE rotation base (Su et al., 2024) in Figure 5. The results show that the default configuration with 4k sliding window attains 100% accuracy within 4k tokens, but the accuracy deteriorates quickly as the context length grows. Naively extending the sliding window size to 32k results in worse performance. By changing the RoPE rotation base to 10^5, the model can achieve over 90% accuracy within 32k tokens. However, this entails a minor trade-off in performance for shorter contexts. A potential avenue for future research is to efficiently adapt

the model to longer contexts through lightweight post-training (Zhu et al., 2023).

5.3 Analysis of Training Hyperparameters

Table 5 presents the results under different configurations. We notice that the Mistral-7B initialization holds an advantage over LLaMA-2 7B, in line with the findings from Mistral-7B technical report (Jiang et al., 2023). The choice of pooling types and LoRA ranks does not affect the overall performance substantially, hence we adhere to the default setting despite the marginal superiority of LoRA rank 8. On the other hand, the way of adding instructions has a considerable impact on the performance. We conjecture that natural language instructions better inform the model regarding the embedding task at hand, and thus enable the model to generate more discriminative embeddings. Our framework also provides a way to customize the behavior of text embeddings through instructions without the need to fine-tune the model or re-build document index.

6 Conclusion

This paper shows that the quality of text embeddings can be substantially enhanced by exploiting LLMs. We prompt proprietary LLMs such as GPT-4 to generate diverse synthetic data with instructions in many languages. Combined with the strong language understanding capability of the Mistral model, we establish new state-of-the-art results for nearly all task categories on the competitive MTEB benchmark. The training process is much more streamlined and efficient than existing multi-stage

approaches, thereby obviating the need for intermediate pre-training.

For future work, we aim to further improve the multilingual performance of our model and explore the possibility of using open-source LLMs to generate synthetic data.

Limitations

In comparison to the mainstream BERT-style encoders, the employment of LLMs, such as Mistral-7B, for text embeddings results in a significantly increased inference cost. The development of more advanced GPUs and better kernel implementations may enhance the efficiency of the inference process. With regards to storage cost, our model is comparatively more expensive, with embeddings of 4096 dimensions. Early successes in reducing embedding dimensions while maintaining competitive performance have been demonstrated through techniques such as Matryoshka representation learning (Kusupati et al., 2022).

For synthetic data generation, we rely on manual prompt engineering to elicit high-quality outputs from proprietary LLMs. Automatic prompt optimization presents a promising avenue for improving the quality of synthetic data.

Acknowledgements

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Luiz Henrique Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Unsupervised dataset generation for information retrieval. *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Daniel Fernando Campos, Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, Li Deng, and Bhaskar Mitra. 2016. Ms marco: A human generated machine reading comprehension dataset. *ArXiv preprint*, abs/1611.09268.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith Hall, and Ming-Wei Chang. 2022. Promptagator: Few-shot dense retrieval from 8 examples. In *The Eleventh International Conference on Learning Representations*.

DataCanary, hilfialkaff, Lili Jiang, Meg Risdal, Nikhil Dandekar, and tomtung. 2017. Quora question pairs.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the*

American society for information science, 41(6):391–407.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic bert sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023. Enabling large language models to generate text with citations. *ArXiv preprint*, abs/2305.14627.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio Cesar Teodoro Mendes, Allison Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero C. Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, S. Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuan-Fang Li. 2023. Textbooks are all you need. *ArXiv preprint*, abs/2306.11644.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2022. Unnatural instructions: Tuning language models with (almost) no human labor. *ArXiv preprint*, abs/2212.09689.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Towards unsupervised dense information retrieval with contrastive learning. *ArXiv preprint*, abs/2112.09118.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *ArXiv preprint*, abs/2310.06825.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham M. Kakade, Prateek Jain, and Ali Farhadi. 2022. Matryoshka representation learning. In *Neural Information Processing Systems*.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Xianming Li and Jing Li. 2023. Angle-optimized text embeddings. *ArXiv preprint*, abs/2309.12871.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *ArXiv preprint*, abs/2308.03281.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2023. Fine-tuning llama for multi-stage text retrieval. *ArXiv preprint*, abs/2310.08319.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark attention: Random-access infinite context length for transformers. *ArXiv preprint*, abs/2305.16300.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *ArXiv preprint*, abs/2202.08904.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Hassan Awadallah. 2023. Orca: Progressive learning

from complex explanation traces of gpt-4. *ArXiv preprint*, abs/2306.02707.

Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas A. Tezak, Jong Wook Kim, Chris Hallacy, Johannes Heidecke, Pranav Shyam, Boris Power, Tyna Eloundou Nekoul, Girish Sastry, Gretchen Krueger, David P. Schnurr, Felipe Petroski Such, Kenny Sai-Kin Hsu, Madeleine Thompson, Tabarak Khan, Toki Sherbakov, Joanne Jang, Peter Welinder, and Lilian Weng. 2022. Text and code embeddings by contrastive pre-training. *ArXiv preprint*, abs/2201.10005.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022a. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. 2022b. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *ArXiv preprint*, abs/1904.08375.

OpenAI. 2023. Gpt-4 technical report. *ArXiv preprint*, abs/2303.08774.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Yifu Qiu, Hongyu Li, Yingqi Qu, Ying Chen, QiaoQiao She, Jing Liu, Hua Wu, and Haifeng Wang. 2022. DuReader-retrieval: A large-scale Chinese benchmark for passage retrieval from web search engine. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5326–5338, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, I. Evtimov, Joanna Bitton, Manish P Bhatt, Cristian Cantón Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre D'efossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2023. Code llama: Open foundation models for code. *ArXiv preprint*, abs/2308.12950.

Timo Schick and Hinrich Schütze. 2021. Generating datasets with pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6943–6951.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2023. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121, Toronto, Canada. Association for Computational Linguistics.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv preprint*, abs/2307.09288.

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2022a. GPL: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2345–2360, Seattle, United States. Association for Computational Linguistics.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder,

and Furu Wei. 2022b. Text embeddings by weakly-supervised contrastive pre-training. *ArXiv preprint*, abs/2212.03533.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual e5 text embeddings: A technical report. *arXiv preprint arXiv:2402.05672*.

Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query expansion with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423, Singapore. Association for Computational Linguistics.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighof. 2023. C-pack: Packaged resources to advance general chinese embedding. *ArXiv preprint*, abs/2309.07597.

Xiaohui Xie, Qian Dong, Bingning Wang, Feiyang Lv, Ting Yao, Weinan Gan, Zhijing Wu, Xiangsheng Li, Haitao Li, Yiqun Liu, et al. 2023. T2ranking: A large-scale chinese benchmark for passage ranking. *ArXiv preprint*, abs/2304.03679.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Xin Zhang, Zehan Li, Yanzhao Zhang, Dingkun Long, Pengjun Xie, Meishan Zhang, and Min Zhang. 2023a. Language models are universal embedders. *ArXiv preprint*, abs/2310.08232.

Xinyu Zhang, Xueguang Ma, Peng Shi, and Jimmy Lin. 2021. Mr. TyDi: A multi-lingual benchmark for dense retrieval. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 127–137, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xinyu Crystina Zhang, Nandan Thakur, Odunayo Ogundepo, Ehsan Kamalloo, David Alfonso-Hermelo, Xiaoguang Li, Qun Liu, Mehdi Rezagholizadeh, and Jimmy Lin. 2023b. Miracl: A multilingual retrieval dataset covering 18 diverse languages. *Transactions of the Association for Computational Linguistics*, 11:1114–1131.

Dawei Zhu, Nan Yang, Liang Wang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. 2023. Pose: Efficient context window extension of llms via positional skip-wise training. In *The Twelfth International Conference on Learning Representations*.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2018. Overview of the third bucc shared task: Spotting parallel sentences in comparable corpora. In *Proceedings of 11th Workshop on Building and Using Comparable Corpora*, pages 39–42.

A   Implementation Details

Baseline Models   For results with mE5$_{base}$ and mE5$_{large}$, we use the public checkpoints available at `https://huggingface.co/intfloat/multilingual-e5-base` and `https://huggingface.co/intfloat/multilingual-e5-large` respectively. For experiments in Table 5, we follow the SGPT (Muennighoff, 2022) paper for the implementation of weighted mean pooling. For the "w/ task type prefix" setting, we prepend "classify: " for the long-short matching subgroup, and "query: " for other asymmetric tasks. No prefix is added for symmetric tasks.

Training Data   For the "E5$_{mistral-7b}$ + full data" setting, our training data comprises generated synthetic data, ELI5 (Fan et al., 2019)(sample ratio 0.1), HotpotQA (Yang et al., 2018), FEVER (Thorne et al., 2018), MIRACL (Zhang et al., 2023b), MSMARCO passage ranking (sample ratio 0.5) and document ranking (sample ratio 0.2) (Campos et al., 2016), NQ (Karpukhin et al., 2020), NLI (Gao et al., 2021), SQuAD (Karpukhin et al., 2020), TriviaQA (Karpukhin et al., 2020), Quora Duplicate Questions (DataCanary et al., 2017)(sample ratio 0.1), MrTyDi (Zhang et al., 2021), DuReader (Qiu et al., 2022), and T2Ranking (Xie et al., 2023)(sample ratio 0.5) datasets. We only include the training set of each dataset. For the datasets without hard negatives, we use mE5$_{base}$ to mine top 100 hard negatives. After sampling, we obtain approximately 1.8 million examples. The entire training process takes fewer than 1k steps to complete.

Hyperparameters for Fine-tuning   When fine-tuning Mistral-7b [2], the batch size is set to 2048 and the learning rate is 10^{-4} with 100 step warmup and linear decay. The weight decay is 0.1. We add 1 hard negative for each query-document pair. The fine-tuning process takes roughly 18 hours on 32 V100 GPUs with a maximum sequence length 512. We add LoRA adapters to all linear layers, resulting in a total of 42M trainable parameters. Our implementation is based on the HuggingFace PEFT library at `https://github.com/huggingface/peft`.

---

[2]`https://huggingface.co/mistralai/Mistral-7B-v0.1`

	nDCG@10					Recall@100				
	BM25	mDPR	mE5$_{base}$	mE5$_{large}$	E5$_{mistral-7b}$ full	BM25	mDPR	mE5$_{base}$	mE5$_{large}$	E5$_{mistral-7b}$ full
ar	48.1	49.9	71.6	76.0	73.3	88.9	84.1	95.9	97.3	96.0
bn	50.8	44.3	70.2	75.9	70.3	90.9	81.9	96.6	98.2	96.0
en	35.1	39.4	51.2	52.9	57.3	81.9	76.8	86.4	87.6	90.2
es	31.9	47.8	51.5	52.9	52.2	70.2	86.4	88.6	89.1	87.5
fa	33.3	48.0	57.4	59.0	52.1	73.1	89.8	91.2	92.9	88.0
fi	55.1	47.2	74.4	77.8	74.7	89.1	78.8	96.9	98.1	96.7
fr	18.3	43.5	49.7	54.5	55.2	65.3	91.5	90.0	90.6	92.8
hi	45.8	38.3	58.4	62.0	52.1	86.8	77.6	92.6	93.9	89.9
id	44.9	27.2	51.1	52.9	52.7	90.4	57.3	87.4	87.9	88.4
ja	36.9	43.9	64.7	70.6	66.8	80.5	82.5	96.0	97.1	95.1
ko	41.9	41.9	62.2	66.5	61.8	78.3	73.7	91.6	93.4	89.4
ru	33.4	40.7	61.5	67.4	67.7	66.1	79.7	92.7	95.5	95.0
sw	38.3	29.9	71.1	74.9	68.4	70.1	61.6	95.6	96.7	95.5
te	49.4	35.6	75.2	84.6	73.9	83.1	76.2	98.0	99.2	95.1
th	48.4	35.8	75.2	80.2	74.0	88.7	67.8	98.0	98.9	96.5
zh	18.0	51.2	51.5	56.0	54.0	56.0	94.4	92.1	93.3	90.1
Avg	39.3	41.5	62.3	66.5	62.9	78.7	78.8	93.1	94.3	92.6

Table 6: nDCG@10 and Recall@100 on the dev set of the MIRACL dataset for all 16 languages.

Datasets	Class.	Clust.	PairClass.	Rerank	Retr.	STS	Summ.	Avg
XLM-R$_{large}$ + full data	72.9	38.7	84.5	53.8	42.0	82.3	29.7	58.0
w/ cont. pre-train	77.2	47.3	85.5	58.6	50.2	84.4	30.7	63.7
E5$_{mistral-7b}$ + full data	78.5	50.3	88.3	60.2	56.9	84.6	31.4	66.6
w/ cont. pre-train	78.7	50.1	87.7	60.9	56.9	84.9	30.2	66.7

Table 7: Detailed results for the effects of contrastive pre-training. For the "E5$_{mistral-7b}$ w/ cont. pre-train" setting, we pre-train Mistral-7B following the mE5 recipe for 10k steps.

Artifacts The model and dataset release information is available at https://github.com/microsoft/unilm/tree/master/e5. We release our trained models and evaluation scripts to facilitate reproducibility and further research.

B Test Set Contamination Analysis

To assess the test set contamination on all the datasets in the MTEB benchmark, we perform a string match based analysis between the test set and our training set, disregarding differences in character case and spacing. We categorize the train-test overlaps into three types:

- **Low entropy texts.** These are texts such as "*i need a coffee*" and "*what does that mean*", which are not considered as contamination because they are common expressions that can occur in various contexts.

- **Question overlap.** We identify 4 test set questions in the DBPedia dataset that also appear in the TriviaQA training set. Given that they constitute a minor portion of the test set, their impact on the overall performance is insignificant.

- **Retrieval corpus overlap.** Several retrieval datasets share the same retrieval corpus. For instance, the DBPedia, NQ, and TriviaQA datasets all use Wikipedia passages, even though their query sets are different. This is a standard evaluation practice in the field of information retrieval, and we do not regard it as contamination.

In summary, we did not detect substantial contamination risks that could alter the main findings of this paper.

Another aspect to consider is the possibility of test set contamination in the training data of Mistral-7B and GPT-4. However, since the training data of these models is not publicly accessible, it is challenging to estimate the degree of such contamination. Given their widespread use in the research community, we believe it is still a valid comparison if other works also employ these models.

C Prompts for Synthetic Data Generation

For asymmetric tasks, we list the four prompt templates in Table 8, 9, 10, and 11. For symmetric tasks, the prompts templates are available in Ta-

Brainstorm a list of potentially useful text retrieval tasks.

Here are a few examples for your reference:
- Retrieve relevant documents for a short keyword web search query that asks for weather information.
- Search for documents that answers a FAQ-style query on children's nutrition.

Please adhere to the following guidelines:
- Specify what the query is, and what the desired documents are.
- Each retrieval task should cover a wide range of queries, and should not be too specific.

Your output must always be a python list of strings only, with about 20 elements, and each element corresponds to a distinct retrieval task in one sentence. Do not explain yourself or output anything else. Be creative!

You have been assigned a retrieval task: {task}

Your mission is to write one text retrieval example for this task in JSON format. The JSON object must contain the following keys:
- "user_query": a string, a random user search query specified by the retrieval task.
- "positive_document": a string, a relevant document for the user query.
- "hard_negative_document": a string, a hard negative document that only appears relevant to the query.

Please adhere to the following guidelines:
- The "user_query" should be {query_type}, {query_length}, {clarity}, and diverse in topic.
- All documents must be created independent of the query. Avoid copying the query verbatim. It's acceptable if some parts of the "positive_document" are not topically related to the query.
- All documents should be at least {num_words} words long.
- The "hard_negative_document" contains some useful information, but it should be less useful or comprehensive compared to the "positive_document".
- Both the query and documents should be in {language}.
- Do not provide any explanation in any document on why it is relevant or not relevant to the query.
- Both the query and documents require {difficulty} level education to understand.

Your output must always be a JSON object only, do not explain yourself or output anything else. Be creative!

Table 8: Prompt template for the short-long matching subgroup. For placeholders, "*{query_type}*" ∈ {extremely long-tail, long-tail, common}, "*{query_length}*" ∈ {less than 5 words, 5 to 15 words, at least 10 words}, "*{difficulty}*" ∈ {high school, college, PhD}, "*{clarity}*" ∈ {clear, understandable with some effort, ambiguous}, "*{num_words}*" ∈ {50, 100, 200, 300, 400, 500}.

ble 12 and 13. To generate multilingual data, we sample the value of "*{language}*" from the language list of XLM-R (Conneau et al., 2020) with higher probability for high-resource languages. When prompting GPT-4/3.5, we set the sampling temperature to 1.0 and the top-p hyperparameter to 1.0, which is higher than the default setting to encourage more diversity.

D   Instructions for Training and Evaluation

We manually write instructions for training datasets, as listed in Table 14. For evaluation datasets, the instructions are listed in Table 15.

Brainstorm a list of potentially useful text classification tasks.

Please adhere to the following guidelines:
- Tasks should cover a diverse range of domains and task types.

Your output must always be a python list of strings only, with about 20 elements, and each element corresponds to a distinct text classification task in one sentence. Do not explain yourself or output anything else. Be creative!

You have been assigned a text classification task: {task}

Your mission is to write one text classification example for this task in JSON format. The JSON object must contain the following keys:
- "input_text": a string, the input text specified by the classification task.
- "label": a string, the correct label of the input text.
- "misleading_label": a string, an incorrect label that is related to the task.

Please adhere to the following guidelines:
- The "input_text" should be {num_words} words and diverse in expression.
- The "misleading_label" must be a valid label for the given task, but not as appropriate as the "label" for the "input_text".
- The values for all fields should be in {language}.
- Avoid including the values of the "label" and "misleading_label" fields in the "input_text", that would make the task too easy.
- The "input_text" is {clarity} and requires {difficulty} level education to comprehend.

Your output must always be a JSON object only, do not explain yourself or output anything else. Be creative!

Table 9: Prompt template for the long-short matching subgroup. For placeholders, "*{num_words}*" ∈ {"less than 10", "at least 10", "at least 50", "at least 100", "at least 200"}, "*{difficulty}*" ∈ {high school, college, PhD}, "*{clarity}*" ∈ {clear, understandable with some effort, ambiguous}.

Brainstorm a list of text matching tasks where both the queries and the groundtruth documents are very short (one or two sentences, even a short phrase).

Here are a few examples:
- Given a scientific paper title, retrieve the title of papers that cite the given paper.
- Match a word with its definition.
- Provided a notable person's name, identify their occupation or achievement.

Your output must always be a python list of strings only, with about 20 elements, and each element corresponds to a distinct task in one sentence. Do not explain yourself or output anything else. Be creative!

You have been assigned a text matching task: {task}

Your mission is to write one example for this task in JSON format. The JSON object must contain the following keys:
- "input": a string, a random input specified by the task.
- "positive_document": a string, a relevant document for the "input" according to the task.

Please adhere to the following guidelines:
- The values of all fields should be in {language}.
- Both the "input" and "positive_document" should be very short (a sentence or a phrase), avoid substantial word overlaps, otherwise the task would be too easy.
- The "input" and "positive_document" should be independent of each other.

Your output must always be a JSON object only, do not explain yourself or output anything else. Be creative!

Table 10: Prompt template for the short-short matching subgroup. We do not generate negative documents as the matching task is already reasonably difficult.

Brainstorm a list of text matching tasks where the queries are long documents.

Here are a few examples:
- Given a document that supports a debatable argument, find another document that contains opposite arguments.
- Provided a lengthy business proposal, retrieve competitive business strategies in the same industry.

Your output must always be a python list of strings only, with about 20 elements, and each element corresponds to a distinct task in one sentence. Do not explain yourself or output anything else. Be creative!

You have been assigned a text matching task: {task}

Your mission is to write one example for this task in JSON format. The JSON object must contain the following keys:
- "input": a string, a random input specified by the task.
- "positive_document": a string, a relevant document for the "input" according to the task.

Please adhere to the following guidelines:
- The values of all fields should be in {language}.
- Both the "input" and "positive_document" should be long documents (at least 300 words), avoid substantial word overlaps, otherwise the task would be too easy.
- The "input" and "positive_document" should be independent of each other.

Your output must always be a JSON object only, do not explain yourself or output anything else. Be creative!

Table 11: Prompt template for the long-long matching subgroup. We do not generate negative documents for API latency reasons.

Write a {unit} triple with varying semantic similarity scores in JSON format. The semantic similarity score ranges from 1 to 5, with 1 denotes least similar and 5 denotes most similar.

Please adhere to the following guidelines:
- The keys in JSON are "S1", "S2", and "S3", the values are all strings in {language}, do not add any other keys.
- There should be some word overlaps between all three {unit}s.
- The similarity score between S1 and S2 should be {high_score}.
- The similarity score between S1 and S3 should be {low_score}.
- The {unit}s require {difficulty} level education to understand and should be diverse in terms of topic and length.

Your output must always be a JSON object only with three keys "S1", "S2" and "S3", do not explain yourself or output anything else. Be creative!

Table 12: Prompt template for monolingual STS. For placeholders, "*{high_score}*" ∈ {4, 4.5, 5}, "*{low_score}*" ∈ {2.5, 3, 3.5}, "*{unit}*" ∈ {sentence, phrase, passage}, "*{difficulty}*" ∈ {elementary school, high school, college}.

Write a {unit} triple with one {unit} in {src_lang} and two {unit}s in {tgt_lang} with varying translation qualities in JSON format.

The triple is denotes as ("S1", "S2", "S3"). The translation quality score ranges from 1 to 5, with higher scores are better.

Please adhere to the following guidelines:
- The values of "S1" is a string in {src_lang}, the value of "S2" and "S3" are strings in {tgt_lang}.
- There should be some word overlaps between "S2" and "S3".
- The translation quality score of "S2" with respect to "S1" should be {high_score}.
- The translation quality score of "S3" with respect to "S1" should be {low_score}.
- "S3" should be grammatical and fluent, but contain some keyword or number translation errors, or miss some information, or contain some redundant information.
- "S1" requires {difficulty} level education to understand and should be diverse in terms of topic and length.

Your output must always be a JSON object only with three keys "S1", "S2" and "S3", do not explain yourself or output anything else. Be creative!

Table 13: Prompt template for bitext retrieval. For placeholders, "*{high_score}*" ∈ {4, 4.5, 5}, "*{low_score}*" ∈ {1.5, 2, 2.5}, "*{unit}*" ∈ {sentence, phrase, passage}, "*{difficulty}*" ∈ {elementary school, high school, college}.

Dataset	Instruction
ELI5	Provided a user question, retrieve the highest voted answers on Reddit ELI5 forum
HotpotQA	Given a multi-hop question, retrieve documents that can help answer the question
FEVER	Given a claim, retrieve documents that support or refute the claim
MIRACL / MrTyDi / NQ	Given a question, retrieve Wikipedia passages that answer the question
/ SQuAD / TriviaQA	Retrieve Wikipedia passages that answer the question
NLI	Given a premise, retrieve a hypothesis that is entailed by the premise
	Retrieve semantically similar text
MS-MARCO	Given a web search query, retrieve relevant passages that answer the query
	Given a web search query, retrieve relevant documents that answer the query
Quora Duplicates	Given a question, retrieve questions that are semantically equivalent to the given question
	Find questions that have the same meaning as the input question
DuReader / T2Ranking	Given a Chinese search query, retrieve web passages that answer the question

Table 14: Instructions for each training dataset.

Task Name	Instruction
AmazonCounterfactualClassif.	Classify a given Amazon customer review text as either counterfactual or not-counterfactual
AmazonPolarityClassification	Classify Amazon reviews into positive or negative sentiment
AmazonReviewsClassification	Classify the given Amazon review into its appropriate rating category
Banking77Classification	Given a online banking query, find the corresponding intents
EmotionClassification	Classify the emotion expressed in the given Twitter message into one of the six emotions: anger, fear, joy, love, sadness, and surprise
ImdbClassification	Classify the sentiment expressed in the given movie review text from the IMDB dataset
MassiveIntentClassification	Given a user utterance as query, find the user intents
MassiveScenarioClassification	Given a user utterance as query, find the user scenarios
MTOPDomainClassification	Classify the intent domain of the given utterance in task-oriented conversation
MTOPIntentClassification	Classify the intent of the given utterance in task-oriented conversation
ToxicConversationsClassif.	Classify the given comments as either toxic or not toxic
TweetSentimentClassification	Classify the sentiment of a given tweet as either positive, negative, or neutral
ArxivClusteringP2P	Identify the main and secondary category of Arxiv papers based on the titles and abstracts
ArxivClusteringS2S	Identify the main and secondary category of Arxiv papers based on the titles
BiorxivClusteringP2P	Identify the main category of Biorxiv papers based on the titles and abstracts
BiorxivClusteringS2S	Identify the main category of Biorxiv papers based on the titles
MedrxivClusteringP2P	Identify the main category of Medrxiv papers based on the titles and abstracts
MedrxivClusteringS2S	Identify the main category of Medrxiv papers based on the titles
RedditClustering	Identify the topic or theme of Reddit posts based on the titles
RedditClusteringP2P	Identify the topic or theme of Reddit posts based on the titles and posts
StackExchangeClustering	Identify the topic or theme of StackExchange posts based on the titles
StackExchangeClusteringP2P	Identify the topic or theme of StackExchange posts based on the given paragraphs
TwentyNewsgroupsClustering	Identify the topic or theme of the given news articles
SprintDuplicateQuestions	Retrieve duplicate questions from Sprint forum
TwitterSemEval2015	Retrieve tweets that are semantically similar to the given tweet
TwitterURLCorpus	Retrieve tweets that are semantically similar to the given tweet
AskUbuntuDupQuestions	Retrieve duplicate questions from AskUbuntu forum
MindSmallReranking	Retrieve relevant news articles based on user browsing history
SciDocsRR	Given a title of a scientific paper, retrieve the titles of other relevant papers
StackOverflowDupQuestions	Retrieve duplicate questions from StackOverflow forum
ArguAna	Given a claim, find documents that refute the claim
ClimateFEVER	Given a claim about climate change, retrieve documents that support or refute the claim
CQADupstackRetrieval	Given a question, retrieve detailed question descriptions from Stackexchange that are duplicates to the given question
DBPedia	Given a query, retrieve relevant entity descriptions from DBPedia
FEVER	Given a claim, retrieve documents that support or refute the claim
FiQA2018	Given a financial question, retrieve user replies that best answer the question
HotpotQA	Given a multi-hop question, retrieve documents that can help answer the question
MSMARCO	Given a web search query, retrieve relevant passages that answer the query
NFCorpus	Given a question, retrieve relevant documents that best answer the question
NQ	Given a question, retrieve Wikipedia passages that answer the question
QuoraRetrieval	Given a question, retrieve questions that are semantically equivalent to the given question
SCIDOCS	Given a scientific paper title, retrieve paper abstracts that are cited by the given paper
SciFact	Given a scientific claim, retrieve documents that support or refute the claim
Touche2020	Given a question, retrieve detailed and persuasive arguments that answer the question
TRECCOVID	Given a query on COVID-19, retrieve documents that answer the query
STS*	Retrieve semantically similar text.
BUCC/Tatoeba	Retrieve parallel sentences.
SummEval	Given a news summary, retrieve other semantically similar summaries

Table 15: Instructions used for evaluation on the MTEB benchmark. "STS*" indicates we use the same instructions for all the STS tasks.

Task group: short-long matching	
Task definition: The query is a name of a computer software or app, find documents that offer tutorials, guides or user experiences about the software or app.	
Generated data: {	

"user_query": "How to use Microsoft Power BI for data analysis",

"positive_document": "Microsoft Power BI is a sophisticated tool that requires time and practice to master. In this tutorial, we'll show you how to navigate Power BI and perform basic data analysis. First, launch the application and log in with a Microsoft account. Next, create a new project using the Wizard on the left-hand toolbar. Import the dataset you wish to work with by clicking on 'Get Data'. Power BI can handle a variety of file formats, including Excel and CSV. Load your data, then use a sequence of drag-and-drop operations to generate charts and (if applicable) maps. If you are confused at any point, consider using the 'Ask a Question' feature for clarification.",

"hard_negative_document": "Excel is an incredibly powerful tool for managing and analyzing large amounts of data. Our tutorial series focuses on how you can achieve maximum efficiency with Excel. In this guide, we'll walk you through transferring data from your computer to an Excel spreadsheet as well as colour-coding cells to match various criteria. We'll also explore VLOOKUP, conditional formatting, and pivot tables, essentially transforming you into a spreadsheet wizard. Once you've mastered these techniques, consider joining us for our Advanced Data Analysis training to branch out even further."
}

Task group: long-short matching

Task definition: Identifying severity level of customer complaints in support tickets

Generated data: {

"input_text": "I am writing to express my intense dissatisfaction with one of your products, a TV that has stopped functioning only a month after purchase. This situation yields less satisfaction to me and speaks voluminously about your quality control procedures in assembly lines. I hope this troubling issue etches into your improvement list for invoking earnest attention.",

"label": "High Severity",

"misleading_label": "Low Severity"
}

Task group: short-short matching

Task definition: Provided a movie quote, find the movie title in which it is said.

Generated data: {

"input": "I'm going to make him an offer he can't refuse.",

"positive_document": "The Godfather"
}

Task group: long-long matching

Task definition: Provided a legal brief, retrieve other legal documents presenting similar legal arguments.

Generated data: {

"input": "In the united states district court for the eastern district of pennsylvania. Plaintiff John Doe brings this action against Defendants ABC Corporation and XYZ Inc., alleging negligence which resulted in his injuries. The Plaintiff alleges ABC Corporation negligently designed, manufactured, and marketed a specific product. The Plaintiff also asserts XYZ Inc. negligently failed to inspect the same product, leading to harmful exposure. Plaintiff respectfully requests this honorable court to hold the defendants liable and award damages. All parties are diverse for purposes of 28 U.S.C. § 1332.",

"positive_document": "Before the Superior Court of California, County of Los Angeles. In the matter of the lawsuit initiated by the Plaintiff Jane Smith against the Defendants PQR Industries and LMN Enterprises, charging they negligently designed, produced, and advertised a product that directly caused harm to her. Furthermore, the Plaintiff has claimed that LMN Enterprises negligently fell short in inspecting the product, leading to unsafe usage. The plaintiff hereby requests this esteemed court to declare the defendants guilty and demands compensation for the damages suffered. According to 28 U.S.C. § 1332, complete diversity exists between the parties to this lawsuit."
}

Task group: bitext matching

Task definition: Retrieve parallel sentences.

Generated data: {

"S1": "Men bugun uzumli tort yasashni rejalashtirdim.",

"S2": "I have planned to make a grape cake today.",

"S3": "I am going to bake a strawberry pie today."
}

Task group: monolingual STS

Task definition: Retrieve semantically similar text.

Generated data: {

"S1": "Tom loves to ride his blue bicycle in the park every morning.",

"S2": "Every morning, Tom enjoys riding his blue bike in the park.",

"S3": "Tom takes his blue pen to school every day."
}

Table 16: Random samples for each subgroup of the synthetic data.

Dataset	w/ synthetic only	w/ synthetic + msmarco	w/o synthetic data	full data
BIOSSES	84.2	81.0	85.4	85.5
SICK-R	78.6	78.5	81.7	82.6
STS12	75.8	74.7	77.9	79.7
STS13	84.3	85.3	88.0	88.4
STS14	80.9	81.2	83.7	84.5
STS15	86.2	86.8	89.5	90.4
STS16	85.0	85.3	86.5	87.7
STS17	87.3	87.7	91.0	91.8
STS22	66.0	67.1	66.2	67.0
STSBenchmark	83.5	84.0	87.8	88.6
SummEval	31.9	32.7	31.9	31.4
SprintDuplicateQuestions	93.5	95.8	96.0	95.7
TwitterSemEval2015	78.0	78.5	81.7	81.6
TwitterURLCorpus	86.5	86.9	87.7	87.8
AmazonCounterfactualClass.	79.6	79.9	77.2	78.7
AmazonPolarityClassification	95.8	95.9	93.9	95.9
AmazonReviewsClassification	56.9	55.5	48.2	55.8
Banking77Classification	86.2	87.0	88.8	88.2
EmotionClassification	49.2	47.6	51.0	49.8
ImdbClassification	94.8	94.9	89.0	94.8
MassiveIntentClassification	79.8	79.9	79.6	80.6
MassiveScenarioClassification	81.7	82.4	82.3	82.4
MTOPDomainClassification	95.6	95.9	95.7	96.1
MTOPIntentClassification	84.9	85.9	83.4	86.1
ToxicConversationsClassification	70.2	70.8	70.9	69.6
TweetSentimentExtractionClass.	63.5	63.4	61.6	63.7
AskUbuntuDupQuestions	64.3	65.3	67.4	67.0
MindSmallReranking	33.1	32.8	32.5	32.6
SciDocsRR	86.0	86.0	85.7	86.3
StackOverflowDupQuestions	52.5	53.7	55.9	54.9
ArxivClusteringP2P	51.4	51.2	47.8	50.5
ArxivClusteringS2S	46.5	44.9	44.6	45.5
BiorxivClusteringP2P	44.5	43.3	36.9	43.5
BiorxivClusteringS2S	40.9	40.1	37.0	40.2
MedrxivClusteringP2P	40.5	39.9	32.6	38.2
MedrxivClusteringS2S	38.0	37.9	32.8	37.5
RedditClustering	56.3	55.9	63.1	57.7
RedditClusteringP2P	66.3	64.8	66.4	66.5
StackExchangeClustering	72.9	72.7	74.5	73.1
StackExchangeClusteringP2P	46.1	45.6	34.3	45.9
TwentyNewsgroupsClustering	52.2	52.5	55.6	54.3
ArguAna	52.2	42.7	62.5	61.9
ClimateFEVER	21.1	28.8	25.2	38.4
CQADupstackAndroidRetrieval	40.8	36.0	44.5	43.0
DBPedia	42.0	43.7	47.7	48.9
FEVER	72.5	83.5	73.1	87.8
FiQA2018	38.1	48.4	54.5	56.6
HotpotQA	48.1	64.0	75.6	75.7
MSMARCO	25.7	45.0	42.9	43.1
NFCorpus	35.5	40.0	35.3	38.6
NQ	53.3	63.5	57.3	63.5
QuoraRetrieval	75.0	79.5	89.5	89.6
SCIDOCS	20.6	15.8	19.0	16.3
SciFact	71.5	71.9	74.7	76.4
Touche2020	25.4	32.5	19.1	26.4
TRECCOVID	82.3	87.3	70.8	87.2
Average	63.1	64.5	64.6	**66.6**

Table 17: Results for each dataset in the MTEB benchmark. The evaluation metrics and detailed baseline results are available in the original paper (Muennighoff et al., 2023).