Negative Matters: Multi-Granularity Hard-Negative Synthesis and Anchor-Token-Aware Pooling for Enhanced Text Embeddings

Anonymous ACL submission

Abstract

Text embedding models are essential for various natural language processing tasks, enabling the effective encoding of semantic information into dense vector representations. These models are typically optimized using triplets of (query, positive, negative) data pairs for contrastive learning, where the negative samples play a critical role in enhancing the model's ability to discern subtle semantic distinctions. In this work, we introduce a **M**ulti-**G**ranularity **H**ard-negative (MGH) synthesis framework that leverages large language models (LLMs) to generate diverse negative samples with varying levels of similarity with the query. This approach facilitates a coarse-to-fine curriculum learning strategy during supervised training, allowing the embedding model to progressively learn more nuanced semantic representations. Meanwhile, we propose an **A**nchor **T**oken **A**ware (ATA) pooling method that assigns higher weights to anchor tokens based on aggregation patterns observed in LLMs, improving text embedding accuracy without increasing model complexity. Comprehensive experiments on the MTEB benchmark demonstrate that our methods achieve state-of-the-art performance, surpassing existing synthesis strategies both with synthetic data and when combined with public retrieval datasets.

1 Introduction

Text embedding models are designed to encode the semantic meaning of a given sequence of natural language words, sentences, or larger text spans into dense vector representations. These vector representations capture not only the lexical content of the text but also its syntactic and semantic nuances, facilitating a wide range of downstream natural language processing (NLP) tasks such as sentiment analysis, text clustering, and content-based information retrieval.

Previous studies have explored the potential of leveraging large language models (LLMs) to generate synthetic data for text embedding training tasks (Bonifacio et al., 2022; Wang et al., 2024; Lee et al., 2024b). The substantial volume of synthetic data contributes to increased diversity, thereby improving the model's robustness across various encoding tasks. However, generating high-quality hard negative samples required by contrastive learning is a challenging task for synthetic models, as an effective hard negative must maintain the right balance in its distinction from the positive examples. If the hard negative is overly similar to the query, it may confuse the model with positive samples, whereas if there is a significant difference in the content, the model may struggle to extract useful information from hard negative samples.

In this light, we propose a data synthesis frame-

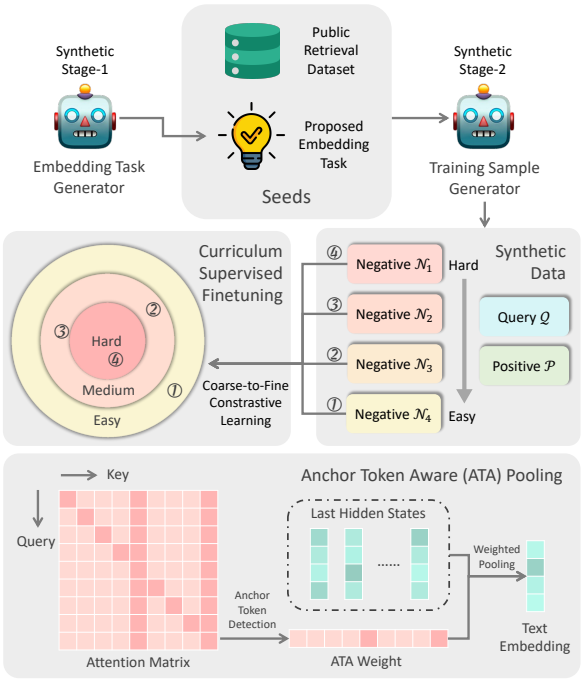

Figure 1: Illustration of our proposed multi-granularity hard-negative sample generation and coarse-to-fine learning paradigm. ①~④ indicate the training order of the synthetic negative samples.

work to fully leverage the LLMs' ability to identify the partial order of text similarity, facilitating the generation of multi-granularity synthetic data. By simultaneously generating hard negative samples at multiple similarity levels, the synthesized data not only improves overall quality but also allows for controlled difficulty of the negatives. In the subsequent supervised training, we use the generated difficulty-controllable data to implement coarse-to-fine curriculum learning. By progressively moving from simple to hard training samples, the embedding model learns increasingly complex representations, resulting in improved stability and effectiveness.

Moreover, as research increasingly focuses on effectively adapting large models into text embedding models, studies (Lee et al., 2024a; BehnamGhader et al., 2024) have explored converting causal attention to bidirectional attention to enhance embedding performance. In this context, two common methods to obtain embeddings from the hidden states of a sequence of tokens are *mean pooling*, which averages the final hidden states, and *last token pooling*, which uses the last hidden state of the <EOS> token as the sentence representation vector. However, mean pooling tends to dilute the critical information tokens when averaging across all tokens, which leads to a loss of significant features (Lee et al., 2024a). In contrast, the last token pooling method is sensitive to noisy information within the sentence, resulting in instability in the encoding (Springer et al., 2024). Recently, NV-Embed (Lee et al., 2024a) addressed the insufficient pooling issue by adding a cross-attention layer over the tokens' final hidden states in a dictionary learning method.

However, we suggest that a simple yet effective pooling method can be utilized without introducing additional parameters. Recent studies have revealed the aggregation pattern of large language models (Wang et al., 2023a; Huang et al., 2024), indicating that decoder-only models tend to aggregate textual information into anchor tokens at shallow layers and use these tokens to generate the next token in deeper layers.

In this paper, we observe that the aggregation pattern still holds in models transformed from causal to bidirectional attention. By simply assigning greater weight to anchor tokens, we achieve improved accuracy in text embedding tasks compared to conventional pooling methods.

Our contributions are summarized as follows:

1. **A MGH Data Synthesis Framework**. We propose a **M**ulti-**G**ranularity **H**ard-negative framework that effectively generates diverse negative samples of varying difficulty levels, fully leveraging the large model's capability to discern the partial order of text similarity. The framework allows for controlled progression in the difficulty of the generated negative examples, enabling subsequent text embedding models to learn a more accurate embedding representation through a coarse-to-fine manner.

2. **An ATA Pooling Method**. We propose an **A**nchor **T**oken **A**ware pooling method that effectively leverages the aggregation pattern of LLMs to acquire a more accurate sentence representation. The model trained with ATA pooling outperformed previous pooling methods and, when trained solely on publicly available retrieval data without MTEB training split, achieved the state-of-the-art model on the MTEB leaderboard.

2 Method

2.1 Multi-granularity Synthetic Data Generation

The overall framework of our proposed data synthesis method is illustrated in Figure 1, which consists of two primary stages. In accordance with the setup of Wang et al. (2024), we begin by querying LLMs to generate a list of potential tasks, categorized into two types: (1) asymmetric text matching tasks comprising four subtasks including short-long match, long-short match, long-long match, and short-short match; (2) symmetric task represented by semantic textual similarity (STS). The task brainstorming process in stage 1 generates a wide range of embedding tasks to enrich the diversity of synthetic data. Further details of the task types are provided in Appendix A.2.

In stage 2, using a diverse set of tasks as seeds, we query the LLM to generate `(query, positive, negative)` samples for subsequent contrastive supervised training, which are then used in contrastive supervised training with a standard infoNCE objective to minimize the distance between query and positive samples, while maximizing the separation from negative samples:

$$\mathcal{L} = -\log \frac{\phi(q, d^+)}{\phi(q, d^+) + \sum_{d^- \in N} (\phi(q, d^-))} \quad (1)$$

where ϕ denotes the cosine similarity function, q, d^+ and d^- represent the query, positive, and

negative samples respectively. Recognizing that the quality of negative samples significantly impacts the supervised training of text-embedding models, we aim to fully leverage the large model's ability to distinguish between different granularities of negative samples during the generation process. To be specific, we formulate the synthetic target as follows:

$$\mathcal{S}^S = \{(\mathcal{Q}^S, \mathcal{P}^S, \{\mathcal{N}_k^S\})\} \quad (2)$$

where $\mathcal{Q}^S$ represents an example query, $\mathcal{P}^S$ denotes its corresponding positive sample, and $\{\mathcal{N}_k^S\}$ refers to a set of hard negative samples with varying levels of granularity indexed by k, which is set to 4 in following experiments.

We use the template illustrated in Figure 2 to constrain the synthesizing format. Differing from Wang et al. (2024), our approach prompts LLM to simultaneously generate multiple hard negative samples $\{\mathcal{N}_k^S\}$ for a single query $\mathcal{Q}^S$. These negative samples are ranked based on their similarity to the query, arranged in descending order from highest to lowest. This approach allows large models to enforce similarity constraints when generating hard negative samples, effectively mitigating the uncontrolled variation in the hard negative sample similarity during the synthetis process across different query samples. The effectiveness of this approach is demonstrated in Section 5 similarity statistics and a detailed case study.

In addition to synthetic data, we also incorporate public retrieval datasets when training the text embedding model, which include $\{(\mathcal{Q}^R, \mathcal{P}^R)\}$ sample pairs. To integrate coarse-to-fine hard negative samples into the subsequent training process, we regenerate the negative samples of the retrieval dataset, denoted as $\{\mathcal{N}_k^R\}$, by querying LLM using the same multi-granularity approach. The refined retrieval dataset, $\mathcal{S}^R = \{(\mathcal{Q}^R, \mathcal{P}^R, \{\mathcal{N}_k^R\})\}$, is then combined with the synthetic dataset $\mathcal{S}^S$ to form the complete training dataset $\mathcal{S}$.

With the multi-granularity dataset $\mathcal{S}$, we adopt a curriculum learning strategy for supervised training of the text embedding model. By adjusting the difficulty level k of the hard negatives, the model progressively learns from coarse-grained to fine-grained distinctions, achieving more stable and effective embeddings. Specifically, during the supervised learning process, we gradually feed the negative samples into the model from $\{\mathcal{N}_4\}$ to $\{\mathcal{N}_1\}$, allocating equal proportions (25% each) to four difficulty levels. Further analysis in Section 4.1 demonstrates the effectiveness of the proposed scheduling method.

```
Omitted for space limitations
The output should be formatted as a JSON object with a
field indicating the relative similarity level. Use the
following format as a guide:
{
  "query": "QUERY_TEXT",
  "positive_example": "POSITIVE_EXAMPLE_TEXT",
  "hard_negative_examples": [
   {
    "similarity_level": "high",
    "text": "HIGH_SIMILARITY_NEGATIVE_EXAMPLE_TEXT"
   },{
     "similarity_level": "medium",
     "text": "MEDIUM_HIGH_SIMILARITY_NEGATIVE_EXAMPLE_TEXT"
   },{
     "similarity_level": "medium",
     "text": "MEDIUM_LOW_SIMILARITY_NEGATIVE_EXAMPLE_TEXT"
   },{
     "similarity_level": "low",
     "text": "LOW_SIMILARITY_NEGATIVE_EXAMPLE_TEXT"}
  ]
}
Omitted for space limitations
Please generate four negative examples for contrastive
learning based on the generated query and positive
example. These examples should be arranged in order of
decreasing similarity to the query, ranging from highly
similar to dissimilar. Ensure the similarity spans a
broad spectrum, and every negative example should be
different, without repeating words from previous
examples. Be creative!
Omitted for space limitations
```

Figure 2: The core template used for prompting LLMs to generate multi-granularity hard negatives. Due to space constraints, the full prompts are presented in Appendix A.1.

2.2 Anchor Token Aware Pooling

After obtaining the hidden states of multiple tokens from the model's final layer, an appropriate approach is required to derive the sentence representation vector v. In the widely adopted mean pooling method, the last hidden states of all tokens are averaged to form a sentence vector representation. This approach can result in the dilution of key information in the text, as non-critical tokens are averaged along with the more significant ones.

The proposed ATA pooling method aims to assign greater weight to anchor tokens (Wang et al., 2023a), which aggregate more semantic information compared to other tokens. This approach allows the model to adaptively allocate weights to the parts that are most pertinent to the task, resulting in a more effective pooling operation.

Motivated by Huang et al. (2024), we calculate the attention weight along the Key dimension of the attention matrix to identify anchor tokens with stronger representational capabilities. Compared to traditional mean pooling, this allows us to assign

higher weights to anchor tokens and more effectively filter out trivial tokens that do not contribute to the semantic information.

Specifically, let $\mathbf{A}_{\mathcal{L}}^{h}$ denote the attention matrix for the attention head h in the model's final layer, and let a_{ij}^{h} represent the corresponding value of $\mathbf{A}_{\mathcal{L}}^{h}[i][j]$, which indicates the attention score between `Query` i and `Key` j. We define the anchor weight of each `Query` as follows:

$$\mathbf{w}_i = \sum_{h=1}^{H} \sum_{j=1}^{S} \log(a_{ij}^{h} \cdot S + 1),\ i \in [1, \ldots, S] \quad (3)$$

where S represents the length of the token sequence, and H denotes the number of attention heads. We multiply a_{ij}^{h} by S because, while the sum of attention weights in the query dimension remains constant (i.e. $\sum_{i=1}^{S} a_{ij}^{h} = 1$), the expected value of each individual element decreases to $\frac{1}{S}$ as the sentence length increases. Multiplying by the sentence length helps to ensure stability across different sentence lengths by maintaining consistent scaling in subsequent computations.

After obtaining the anchor weights, we normalize them by applying a linear weight adjustment along the `Query` dimension to compute the weight corresponding to each token, denoted as $\tilde{\mathbf{w}}_i$, such that the sum of all $\tilde{\mathbf{w}}_i$ equals 1:

$$\tilde{\mathbf{w}}_i = \frac{\mathbf{w}_i}{\sum_{j=1}^{S} \mathbf{w}_j},\ i \in [1, \ldots, S] \quad (4)$$

We then apply the normalized weights $\tilde{\mathbf{w}}_i$ to reweight the hidden states $\mathbf{H}_{\mathcal{D}} \in \mathbb{R}^{S \times \text{hid\_dim}}$ and obtain the final sentence embedding $\mathbf{v}$:

$$\mathbf{v} = \sum_{i=1}^{S} \tilde{\mathbf{w}}_i \times \mathbf{H}_{\mathcal{D}}[i] \quad (5)$$

3 Experiments

3.1 Data Synthetic Details

To ensure a fair comparison with Wang et al. (2024), we generated an equivalent volume of synthetic data, maintaining a consistent total token consumption of 180M. In order to minimize the costs associated with data generation, we utilized the APIs of GPT-4o and DeepSeek-V2. We observed that compared to GPT-4o, DeepSeek-V2 is relatively less creative when generating the potential tasks in stage 1 and tends to produce repetitive negative samples during stage 2. Consequently, we relied on GPT-4o to complete all stage 1 generation processes. In stage 2, we initially used GPT-4o to generate a sufficient amount of data, which was then input as seeds into DeepSeek-V2. Leveraging the data cache provided by the DeepSeek-V2 API, introducing additional seeds as input did not incur significant additional costs. The distribution of synthetic data across different task types is detailed in Appendix A.3.

The retrieval dataset used in supervised learning was curated by Springer et al. (2024), which consist of approximately 1.5 million samples, covering a variety of languages and retrieval scenarios. In line with LLM2Vec (BehnamGhader et al., 2024), we only used about one-third of the curated retrieval dataset. For more details on the dataset composition, please refer to Appendix B.3.

3.2 Experimental Details

Model Setup. To validate the effectiveness of our proposed fine-grained data synthesis framework and the ATA pooling method, we perform experiments following the open-sourced LLM2VEC (BehnamGhader et al., 2024) model, while replacing the supervised training stage with our method. Specifically, we adopt Mistral-7B-Instruct-v0.2 (Jiang et al., 2023) as the base model. We then transform the model's attention pattern from causal to bidirectional and integrate the LoRA weights trained on the masked next-token prediction task introduced in LLM2VEC, enabling the model to better adapt to bidirectional attention patterns. This setup serves as the starting point for our model.

Supervised Training. We adopt the standard InfoNCE loss, with both in-batch negatives and hard negatives utilized for training. To ensure a fair comparison, the prompt template follows Wang et al. (2024), as illustrated in Appendix B.3. Supervised training on the full dataset is conducted for 1600 steps, while training on public retrieval dataset is performed for 1000 steps, with a batch size of 64 and gradient accumulation of 8 in both cases. All training was performed on a single 80GB H100 GPU, taking approximately 32 hours to complete 1600 training steps. Further training hyperparameters are represented in Appendix B.1.

Evaluation. We assess our model on the widely used MTEB benchmark (Muennighoff et al., 2023), which encompasses a wide variety of text embedding tasks across different scenarios and domains. The benchmark includes 56 English embedding tasks organized into 7 categories: classification (12), clustering (11), pair classification (3), reranking (4), retrieval (15), semantic textual similarity

	FT. Data	Class.(12)	Clust.(11)	Pair.(3)	Rerank.(4)	Retr.(15)	STS(10)	Summ.(1)	Avg.(56)
	Sample Num	Acc.	V-Meas.	AP	MAP	nDCG@10	Spear.	Spear.	
Models trained with synthetic data only									
$\text{Mistral}_{\text{gpt-4o}}$ (Chen et al., 2024)	230K	77.7	47.7	83.9	58.7	46.7	80.9	30.7	62.2
$\text{Gecko}_{\text{o1b-768}}$ (Lee et al., 2024b)	6.6M	70.3	46.8	86.2	57.6	**53.2**	**83.1**	**32.2**	62.6
$\text{E5}_{\text{mistral-7b}}$ (Wang et al., 2024)	500K	78.2	**50.5**	86.0	59.0	46.9	81.2	31.9	63.1
SPEED (Chen et al., 2024)	920K	78.3	48.6	**86.3**	59.8	48.1	82.6	31.7	63.4
MGH(Ours)	310K	**78.6**	49.7	86.1	**60.1**	51.2	82.3	31.6	**64.5**
Models trained with synthetic data & public available retrieval data									
GTR_{xxl} (Ni et al., 2022)	662K	67.4	42.4	86.1	56.7	48.5	78.4	30.6	59.0
$\text{text-embedding-3}_{\text{large}}$ [1]	-	75.5	49.0	85.7	59.2	55.4	81.7	29.9	64.6
jina-embed (Sturua et al., 2024)	-	**82.6**	45.3	84.0	58.1	53.9	**85.8**	29.7	65.5
$\text{Gecko}_{\text{o1b-768}}$ (Lee et al., 2024b)	>6.6M	81.2	47.5	87.6	58.9	55.7	85.1	**32.6**	66.3
$\text{E5}_{\text{mistral-7b}}$ (Wang et al., 2024)	1.8M	78.5	**50.3**	**88.3**	60.2	56.9	84.6	31.4	66.6
SPEED (Chen et al., 2024)	2.2M	78.4	49.3	88.2	**60.8**	56.5	85.5	31.1	66.5
MGH(Ours)	820K	78.8	50.1	87.9	59.8	**57.5**	85.6	31.3	**67.0**

Table 1: Full MTEB benchmark performance comparison of different synthesis models, with training conducted on *synthetic data only* and *both synthetic and public retrieval dataset*. The highest performances are highlighted in bold, while the second-highest are indicated with underlines. Numbers in parentheses in column headers indicate the number of subtasks within each task category. *FT. Data Sample Num.* refers to the number of sample pairs used for training.

(10), and summarization (1).

3.3 Main Results

Table 1 presents a comparison of the performance of our proposed method against existing data synthesis approaches on the MTEB dataset. The results underscore the effectiveness of our data generation framework, demonstrating superior performance in both the synthetic-only and full-data settings. In particular, on the more challenging retrieval tasks, the full-data results achieved the best performance, underscoring the efficacy of our synthesis method.

To evaluate the effectiveness of the proposed ATA pooling method, we compare our model with previous approaches that leverage LLMs for text embedding. However, previous research has indicated that incorporating the MTEB dataset's training split during the supervised training process introduces a significant amount of MTEB-related data, thereby increasing the risk of over-fitting (Li et al., 2024). Therefore, we opted not to include the second training phase proposed by NV-Embed, which utilizes the MTEB training split for a continuous training.

Table 2 compares the performance of our approach with existing models trained exclusively on publicly available retrieval data. Our model achieves state-of-the-art performance under the MTEB training split free setting, demonstrating the effectiveness of the ATA pooling method.

[1] https://platform.openai.com/docs/guides/embeddings

Model	MTEB Score
SGPT (Muennighoff, 2022)	58.93
UDEVER-bloom-7b (Zhang et al., 2023a)	60.63
ECHO (Springer et al., 2024)	64.68
LLM2Vec (BehnamGhader et al., 2024)	64.80
NV-Embed (Lee et al., 2024a)	64.18
bge-large-en-v1.5 (Xiao et al., 2024)	64.23
bge-en-icl w/o icl (Li et al., 2024)	64.83
bge-en-icl w/ icl (Li et al., 2024)	66.08
MGH(Ours) w/o icl	65.87
MGH(Ours) w/ icl	**66.43**

Table 2: Performance comparison of MTEB scores for different models trained on publicly available retrieval corpora, without introducing MTEB training split during training.

4 Ablation Study

4.1 Data Synthesis and Training Strategies

In this section, we evaluate the effectiveness of the proposed MGH synthesis framework through an ablation study on a subset of the MTEB benchmark, as used in Springer et al. (2024)[2]. We evaluate the impact of data training order on supervised learning through the following experimental settings, analyzing how the results evolve throughout the training process over 1600 steps:

1. **Curriculum Learning:** Progressing from easier to more challenging hard negatives, as adopted in our main approach.

[2] Conducting a full evaluation on the MTEB dataset is computationally expensive, requiring over 200 hours on a single H100 GPU. Therefore, a subset of the dataset was selected for this study. Details of the subset composition can be found in Appendix C.2.

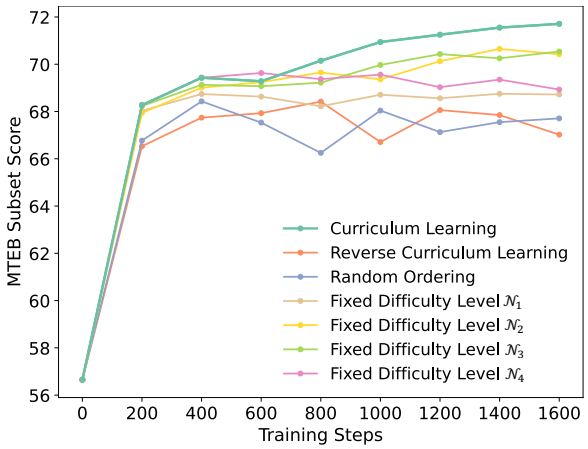

Figure 3: Trend of MTEB subset scores during supervised training across four experimental settings.

2. **Reverse Curriculum Learning:** Progressing hard negative samples from harder to easier.
3. **Random Ordering:** Randomly inputting hard negative examples of varying difficulty.
4. **Fixed Difficulty Level:** Consistently using hard negatives of a fixed difficulty level.

The results in Figure 3 demonstrate that the curriculum learning strategy adopted by MGH not only achieves the best performance but also maintains greater stability during training. In contrast, the random ordering strategy predictably exhibits fluctuations in the results, while the reverse curriculum learning method also fails to yield satisfactory training outcomes, as the model struggles to adapt to the reversed difficulty progression.

Additionally, none of the four fixed difficulty settings outperformed the curriculum learning approach. Among them, maintaining a difficulty level of 2 or 3 yielded relatively better results, while using excessively low or high difficulty levels failed to converge to optimal outcomes. This suggests that overly simple negative examples may prevent the model from learning useful knowledge, and excessively difficult synthetic negatives may hinder the model's ability to distinguish them from positive examples at early training stages.

4.2 Pooling Methods

This section presents a detailed comparison of four pooling methods employed in embedding tasks, namely mean pooling, last token pooling, NV-Embed pooling (Lee et al., 2024a), and the proposed ATA pooling. The ablation experiments leveraged publicly available retrieval datasets only, adhering to the hyperparameter settings outlined in Section 3.2. The evaluation is conducted using the full MTEB benchmark, with Table 3 summarizing their respective performances.

Pooling Method	MTEB Score
mean pooling	65.41
last pooling	64.97
NV-Embed pooling	65.80
ATA pooling (Ours)	**65.87**

Table 3: Performance comparison of MTEB scores for different pooling methods trained on publicly available retrieval corpora.

The results suggest that conventional mean pooling and last token pooling yield subpar performance in text embedding tasks using bidirectional models. On the other hand, both the NV-Embed pooling and ATA pooling methods demonstrate favorable results, validating the necessity of adaptive reweighting for the last hidden states.

5 Further Analysis

5.1 Difficulty of Synthesized Hard Negatives

To assess the granularity of generated negative samples, we use the model trained exclusively on publicly available retrieval corpora, as described in Section 4.2, to evaluate the similarity between the four levels of synthesized $\mathcal{N}_k$ and the query $\mathcal{Q}$. Since this model was not exposed to the synthesized negative samples during training, it serves as an unbiased evaluator to assess the effectiveness of our MGH framework in generating negative samples with different difficulty levels.

Negative Granularity	Cosine Similarity
$\mathcal{N}_1$ (Hardest)	0.881
$\mathcal{N}_2$ (Medium)	0.857
$\mathcal{N}_3$ (Medium)	0.845
$\mathcal{N}_4$ (Easiest)	0.793

Table 4: Average cosine similarity between queries and negative samples at different difficulty levels.

The experimental results in Table 4 demonstrate that average cosine similarity between the negative samples and the query increases progressively from $\mathcal{N}_4$ to $\mathcal{N}_1$. This aligns with our initial hypothesis that $\mathcal{N}_1$ (hardest) negative samples should be more similar to the query than $\mathcal{N}_4$ (easiest), indicating that our MGH framework effectively generates increasingly challenging negative samples at higher difficulty levels.

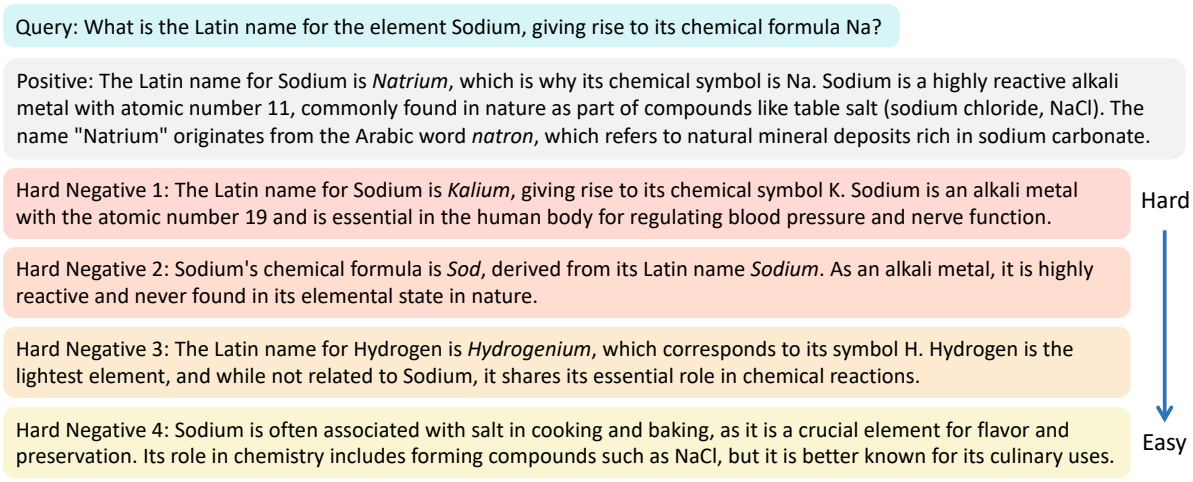

Figure 4: An example of a multi-granularity hard negative synthesis, randomly selected for illustration.

5.2 Case Study

How does MGH enhance hard negative sample quality? We illustrate this through the example presented in Figure 4, which demonstrates how MGH effectively improves the quality of hard negative samples by leveraging multi-granularity similarity constraints.

In this example, the data synthetic model is tasked with generate negative samples for the Latin name of the element *Sodium*. The GPT-4o model used in this case selects *Potassium*, which shares similar chemical properties with *Sodium*, as the most challenging hard negative example. Subsequently, the model generates a fake Latin name for *Sodium* as a moderately confusable negative sample, followed by answering element *Hydrogen* as a more distinguishable example. While the first three negative samples involve answering the Latin names of chemical elements, the last simplest negative sample generated by the model focuses on *Sodium* but lacks any reference to its Latin name.

As demonstrated in the example above, the MGH approach effectively distills world knowledge from LLMs, enabling the generation of multiple negative samples with varying granularities. As the examples progress from challenging to simple, the synthetic model's outputs range from showing subtle differences in detail to being more easily distinguishable. In this process, the subsequent negative samples are adjusted based on the previously generated prefix, enabling a dynamic progression of negative sample difficulty, further enhancing the quality of negative sample generation.

How does ATA reweight using aggregation pattern? As shown in the example from Figure 5, the aggregation pattern still remains when the base model is transformed from causal to bidirectional attention. The figure illustrates three prominent anchor tokens: the initial token, the punctuation between the two sentences, and the `[INST]` template appended to the end of the sentence. Accordingly, these anchor tokens receive higher weight values in the ATA weight calculation, contributing more significantly to the subsequent computation of text embeddings.

Through observations of numerous examples, we found that most samples allocate a greater proportion of the ATA weight to the three anchor patterns mentioned above, with particular emphasis on the `[INST]` token at the sentence's end. Therefore, the ATA pooling method captures the important last token while also dynamically identifying key positions within the preceding text. This approach not only mitigates the stability issues associated with relying solely on the last token but also assigns greater weight to tokens that are essential for capturing the entire semantic meaning of the input sequence, thereby facilitating more effective embedding learning.

5.3 Cost of Synthetic Data

Although our model entails additional tokens to generate multiple hard negatives per synthetic sample, the cost is offset by maintaining the same token consumption (180M) as Wang et al. (2024), which results in fewer synthetic samples being generated. Under this fair comparison, the model trained with our MGH strategy outperforms previous state-of-the-art results, demonstrating that our method is more effective under the same token consumption.

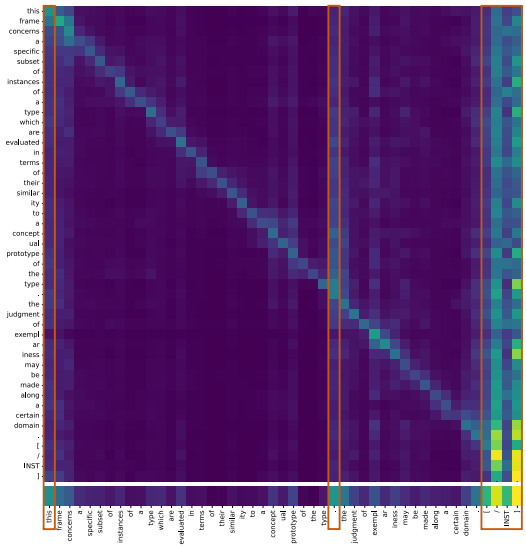

Figure 5: The upper part illustrates the summed results of the model's final layer attention weights across 32 attention heads[3], while the lower part shows the corresponding ATA weights for each token. Example is randomly selected in STS13 evaluation split.

6 Related Work

LLM Based Text Embedding Models In recent years, as decoder-only models have scaled up in terms of parameters and training data, researchers have explored the possibility of transforming next-token prediction models into effective text embedding models through continued training. Neelakantan et al. (2022) was the first to apply the GPT-3 model to text embedding tasks, leveraging the <EOS> token as the representation vector. Subsequent work by Ma et al. (2024) employed a similar last-token pooling method, fine-tuning the LLaMA-2 model.

However, the autoregressive training objective imposes an inherent limitation on the model's performance, as the causal attention mask prevents earlier tokens from accessing subsequent tokens. SGPT (Muennighoff, 2022) addressed this limitation by linearly assigning more weight to tokens at later positions, a strategy subsequently adopted by $E5_{\text{mistral-7b}}$ (Wang et al., 2024). LLM2Vec (BehnamGhader et al., 2024) transformed the model from causal to bidirectional attention by employing a masked next-token prediction approach, followed by mean pooling for supervised learning. Recent work by NV-Embed (Lee et al., 2024a) introduced an additional cross-attention layer for hidden state pooling, simultaneously removing the causal mask. Additionally, Echo embeddings (Springer et al., 2024) repeated a text twice and used the second instance to compute the representation vector. In this work, we attempt to address the insufficient pooling problem in LLM based embedding models by an adaptive weighting strategy using the model's aggregation pattern.

Data Synthesis for Embedding Models High-quality data is crucial for training effective text embedding models. Previous studies (Nogueira et al., 2019; Wang et al., 2023b) have explored expansion-based approaches to augment document and query data. With the advancements in large language model (LLM) capabilities, recent research has focused on leveraging LLMs to generate large amounts of high-quality supervised training data (Wang et al., 2024; Jeronymo et al., 2023; Sturua et al., 2024). In domain-specific retrieval, studies (Dai et al., 2022; Khramtsova et al., 2024) have shown that LLM-generated query-document pairs significantly improve embedding quality in domain-specific retrieval tasks. Additionally, Gecko (Lee et al., 2024b) curates web data to enable LLMs to produce high-quality synthetic samples. Our work focuses on how to enable large models to generate multi-granularity synthetic negative examples, and achieves more efficient and stable text embedding model training by controlling the training difficulty.

7 Conclusion

In this work, we evaluate the importance of hard negative granularity when training text embedding models using contrastive learning. Through the proposed MGH synthesis framework, we generate diverse negative samples at varying levels of similarity, enabling the embedding model to learn more nuanced semantic representations by coarse-to-fine curriculum learning approach. Experimental results demonstrate that our methods achieve state-of-the-art performance on the MTEB benchmark, outperforming existing synthesis strategies both with synthetic-only and combined datasets. Additionally, our proposed ATA pooling method effectively leverages the aggregation patterns inherent in large language models, improving sentence pooling efficacy without introducing extra parameters. Ablation studies confirm the effectiveness of our MGH framework and ATA pooling method in enhancing text embedding model performance and training stability.

[3]For space limitations, the individual attention maps from all 32 attention heads are provided in Appendix D.2.

8 Limitations

Despite the effectiveness of our method, there are several limitations that should be acknowledged: (1) Due to the costs associated with API usage, we limited the synthetic data generation to the same token volume as used in previous studies (Wang et al., 2024). This constraint prevented us from exploring whether a larger synthetic dataset could further improve the performance of the text embedding model. We leave this exploration to future work, where more extensive synthetic data could be generated to assess the scalability and potential performance gains. (2) To facilitate comparisons with prior work (BehnamGhader et al., 2024; Springer et al., 2024), we used Mistral-7b-v0.2-Instruct as our base text embedding model. Given the continuous advancements in 7b-level models, we plan to investigate more powerful models as our base embedding model in our future work.

References

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. Llm2vec: Large language models are secretly powerful text encoders. *arXiv preprint arXiv:2404.05961*.

Luiz Henrique Bonifacio, Hugo Queiroz Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Data augmentation for information retrieval using large language models. *ArXiv*, abs/2202.05144.

Haonan Chen, Liang Wang, Nan Yang, Yutao Zhu, Ziliang Zhao, Furu Wei, and Zhicheng Dou. 2024. Little giants: Synthesizing high-quality embedding data at scale. *Preprint*, arXiv:2410.18634.

Zhuyun Dai, Vincent Y. Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B. Hall, and Ming-Wei Chang. 2022. Promptagator: Few-shot dense retrieval from 8 examples. *Preprint*, arXiv:2209.11755.

DataCanary, hilfialkaff, Lili Jiang, Meg Risdal, Nikhil Dandekar, and tomtung. 2017. Quora question pairs. `https://kaggle.com/competitions/quora-question-pairs`. Kaggle.

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.

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.

Qidong Huang, Xiaoyi Dong, Pan Zhang, Bin Wang, Conghui He, Jiaqi Wang, Dahua Lin, Weiming Zhang, and Nenghai Yu. 2024. Opera: Alleviating hallucination in multi-modal large language models via over-trust penalty and retrospection-allocation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13418–13427.

Vitor Jeronymo, Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, Roberto Lotufo, Jakub Zavrel, and Rodrigo Nogueira. 2023. Inpars-v2: Large language models as efficient dataset generators for information retrieval. *Preprint*, arXiv:2301.01820.

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 arXiv: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.

Ekaterina Khramtsova, Shengyao Zhuang, Mahsa Baktashmotlagh, and Guido Zuccon. 2024. Leveraging llms for unsupervised dense retriever ranking. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1307–1317.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024a. Nv-embed: Improved techniques for training llms as generalist embedding models. *arXiv preprint arXiv:2405.17428*.

Jinhyuk Lee, Zhuyun Dai, Xiaoqi Ren, Blair Chen, Daniel Cer, Jeremy R Cole, Kai Hui, Michael Boratko, Rajvi Kapadia, Wen Ding, et al. 2024b. Gecko: Versatile text embeddings distilled from large language models. *arXiv preprint arXiv:2403.20327*.

Chaofan Li, MingHao Qin, Shitao Xiao, Jianlyu Chen, Kun Luo, Yingxia Shao, Defu Lian, and Zheng Liu. 2024. Making text embedders few-shot learners. *arXiv preprint arXiv:2409.15700*.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2421–2425.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *Preprint*, arXiv: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.

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

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. 2022. 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 arXiv:1904.08375*.

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.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. 2024. Repetition improves language model embeddings. *arXiv preprint arXiv:2402.15449*.

Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Nan Wang, and Han Xiao. 2024. jina-embeddings-v3: Multilingual embeddings with task lora. *Preprint*, arXiv:2409.10173.

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.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023a. Label words are anchors: An information flow perspective for understanding in-context learning. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, Bangkok, Thailand. Association for Computational Linguistics.

Liang Wang, Nan Yang, and Furu Wei. 2023b. 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, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 641–649, New York, NY, USA. Association for Computing Machinery.

Xiaohui Xie, Qian Dong, Bingning Wang, Feiyang Lv, Ting Yao, Weinan Gan, Zhijing Wu, Xiangsheng Li, Haitao Li, Yiqun Liu, and Jin Ma. 2023. T2ranking: A large-scale chinese benchmark for passage ranking. *Preprint*, arXiv: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. *Preprint*, arXiv: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 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.

A Experimental Details for Data Synthesis

A.1 Prompts for Data Synthesis

To enable large models to generate multiple hard negatives with varying granularities, we extend and refine the prompt template proposed by Wang et al. (2024). Specifically, we modify the original template to guide the model in producing negative samples with different levels of similarity to the query, thereby enhancing the diversity and difficulty of the generated data. Table 6 illustrates the complete prompt template used to generate short-long match tasks as an example.

A.2 Details of Task Categories

During the task brainstorming process in synthesis stage 1, we followed Wang et al. (2024) to systematically classify the potential tasks into two distinct categories: asymmetric tasks and symmetric tasks. The fundamental distinction lies in the semantic relationship between queries and their corresponding positive documents. In asymmetric tasks, the query and its relevant document exhibit semantic relevance but are not paraphrases of one another. Conversely, symmetric tasks are characterized by query-document pairs that preserve semantic equivalence through different linguistic formulations.

A.3 Statistics on Synthesized Data

Figure 6 illustrates the distribution of synthetic data across five different task types. In line with Wang et al. (2024), we adopted the same task ratio allocation, where Short-Long, Long-Short, and STS comprise the majority of the data, while Long-Long and Short-Short account for relatively smaller proportions.

B Experimental Details for Supervised Training

B.1 Hyperparameters

We present the hyperparameters involved in the supervised training in Table 5. The *max sequence length* specifies that any text sequence exceeding this number of tokens is truncated in our text embedding model.

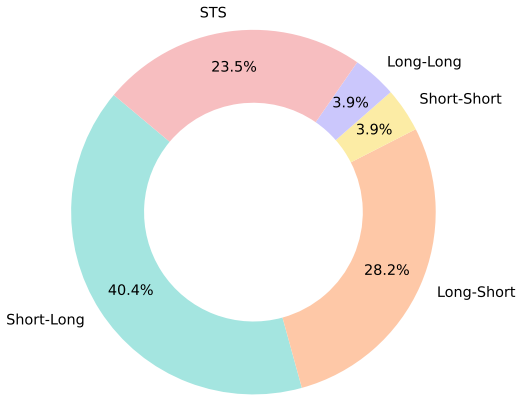

Figure 6: Distribution of the task categories in the synthetic data

Hyperparameter	Value
Batch Size	64
Gradient Accumulation Steps	8
Learning Rate	2e-5
Max Sequence Length	512
LoRA rank	16
LoRA α	32
Optimizer	Adam
Training Steps - Synthetic Only	600
Training Steps - Public Only	1000
Training Steps - Synthetic & Public	1600
Warmup Steps - Synthetic Only	200
Warmup Steps - Public Only	300
Warmup Steps - Synthetic & Public	300

Table 5: Hyperparameters used in the experiments

B.2 Training Time

When training for 1600 steps on a single H100 GPU, the time required with and without ATA pooling was 32.45 hours and 32.40 hours, respectively, demonstrating that out proposed ATA pooling strategy doesn't adds additional computational overhead compared to the LLM backbone. The slight variations in training and inference time are likely due to normal fluctuations in GPU workload and system scheduling.

B.3 Public Retrieval Datasets

We follow the training data setup from previous work (Springer et al., 2024; BehnamGhader et al., 2024), adopting the dataset configuration used by Wang et al. (2024), which includes the following datasets: ELI5 (Fan et al., 2019) (sample ratio 0.1) , HotpotQA (Yang et al., 2018), FEVER (Thorne et al., 2018), MIRACL (Zhang et al., 2023b), MS-MARCO (Bajaj et al., 2016) passage ranking (sam-

ple ratio 0.5) and document ranking (sample ratio 0.2), 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), Mr-TyDi (Zhang et al., 2021), DuReader (Qiu et al., 2022), and T2Ranking (Xie et al., 2023) (sample ratio 0.5). The full supervised training data has approximately 1.5M training examples. The instructions applied to each dataset are in line with BehnamGhader et al. (2024), which are listed in Table 7.

B.4 Similarity Function

The cosine similarity was adopted as the similarity metric in Equation 1, which can be mathematically expressed as follows:

$$\phi = \frac{\bar{u} \cdot \bar{v}}{||\bar{u}|| \times ||\bar{v}||}$$

C Experimental Details for Evaluation

C.1 Prompts for MTEB Evaluation

For a fair comparison with previous work (Wang et al., 2024; BehnamGhader et al., 2024; Lee et al., 2024a) evaluated on MTEB, we adopted the same set of prompt instructions used in their evaluations when assessing our model's performance. The instructions applied to each evaluation dataset are listed in Table 8.

C.2 Subset Used for Ablation Study

To speed up evaluation in the ablation study, we follow Springer et al. (2024) by selecting a representative subset of the MTEB evaluation benchmark, which includes the following datasets: FiQA2018, SCIDOCS, SciFact, NFCorpus, TwitterSemEval2015, TwitterURLCorpus, ImdbClassification, AmazonReviewsClassification, TweetSentimentExtractionClassification, MTOPDomainClassification, TwentyNewsgroupsClustering, BiorxivClusteringS2S, MedrxivClusteringS2S, StackOverflowDupQuestions, AskUbuntuDupQuestions, SciDocsRR, BIOSSES, STS12, STS13, STS14, STS15, STS16, STS17, STS22, STSBenchmark, and SICK-R.

D Additional Results

D.1 Full MTEB Results

In this section, we present the complete results for all 56 MTEB datasets across the three experimental settings of our main experiment: public retrieval data only, synthetic data only, and full data. The corresponding results are shown in Table 9.

D.2 Full Attention Matrices

As shown in Figure 7, after transforming the attention mask of the base model (i.e. Mistral-7B-Instruct-v0.2) from causal to bidirectional, the attention heads continue to exhibit distinct patterns, with some heads focus on tokens in their original positions, while others show higher attention scores across all query dimensions at the current position (e.g. Head 1, Head 6, Head 21 and Head 22). The latter pattern reflects the characteristics of anchor tokens, allowing for more effective aggregation of information from the entire sentence. Consequently, in our ATA pooling method, these attention heads are assigned with greater pooling weight.

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 the following 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 list of strings, hard negative documents that only appears relevant to the query.

The output should be formatted as a JSON object with a field indicating the relative similarity level of hard negative examples. Use the following format as a guide:

```
{
    "user_query": "QUERY_TEXT",
    "positive_document": "POSITIVE_EXAMPLE_TEXT",
    "hard_negative_document": [
        {
            "similarity_level": "high",
            "text": "HIGH_SIMILARITY_NEGATIVE_EXAMPLE_TEXT"
        },{
            "similarity_level": "medium",
            "text": "MEDIUM_HIGH_SIMILARITY_NEGATIVE_EXAMPLE_TEXT"
        },{
        "similarity_level": "medium",
        "text": "MEDIUM_LOW_SIMILARITY_NEGATIVE_EXAMPLE_TEXT"
        },{
            "similarity_level": "low",
            "text": "LOW_SIMILARITY_NEGATIVE_EXAMPLE_TEXT"
        }
    ]
}
```

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". Please generate four hard negative documents for contrastive learning based on the generated query and positive example. These examples should be arranged in order of decreasing similarity to the query, ranging from highly similar to dissimilar. Ensure the similarity spans a broad spectrum, and every negative example should be different, without repeating words from previous examples.
- 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 6: Prompt template for the short-long matching task. 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}.

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

Table 7: The prompt instructions used for public retrieval datasets, following BehnamGhader et al. (2024)

Task Name	Instruction
AmazonCounterfactualClassification	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.
SummEval	Given a news summary, retrieve other semantically similar summaries

Table 8: The prompt instructions used for MTEB benchmark evaluation, following Wang et al. (2024). The "STS*" instruction applies to all STS tasks.

Dataset	Public Retrieval Data Only	Synthetic Data Only	Full Dataset
AmazonCounterfactualClassification	80.1	78.7	79.2
AmazonPolarityClassification	94.0	94.4	95.9
AmazonReviewsClassification	51.8	54.1	55.8
ArguAna	60.2	51.4	61.3
ArxivClusteringP2P	48.1	50.7	50.3
ArxivClusteringS2S	46.0	47.2	46.9
AskUbuntuDupQuestions	64.2	66.3	66.1
BIOSSES	85.6	85.1	87.5
Banking77Classification	88.5	88.3	89.2
BiorxivClusteringP2P	37.7	43.8	42.7
BiorxivClusteringS2S	36.9	41.4	41.2
CQADupstackRetrieval	48.8	44.3	47.1
ClimateFEVER	35.4	26.0	37.8
DBPedia	51.5	45.8	52.3
EmotionClassification	51.2	53.4	51.9
FEVER	91.2	79.1	89.4
FiQA2018	54.1	45.8	55.8
HotpotQA	77.6	57.9	75.9
ImdbClassification	90.3	93.4	94.2
MSMARCO	43.4	29.3	42.4
MTOPDomainClassification	96.3	95.7	96.6
MTOPIntentClassification	86.5	87.4	87.0
MassiveIntentClassification	80.1	80.6	80.3
MassiveScenarioClassification	82.1	81.8	82.4
MedrxivClusteringP2P	32.2	34.8	33.6
MedrxivClusteringS2S	32.5	35.4	34.8
MindSmallReranking	32.5	33.8	33.3
NFCorpus	39.4	37.9	38.5
NQ	65.9	57.7	66.9
QuoraRetrieval	89.5	86.0	89.1
RedditClustering	63.9	61.7	64.8
RedditClusteringP2P	66.8	64.1	67.3
SCIDOCS	22.0	23.7	22.7
SICK-R	83.5	80.4	83.8
STS12	76.6	75.4	79.8
STS13	86.8	86.6	88.3
STS14	83.1	82.4	85.6
STS15	88.5	88.6	91.3
STS16	85.9	86.6	88.1
STS17	91.7	87.0	91.9
STS22	67.9	66.5	69.7
STSBenchmark	87.9	84.4	89.7
SciDocsRR	84.4	85.7	84.7
SciFact	78.6	74.1	76.4
SprintDuplicateQuestions	95.3	94.7	95.3
StackExchangeClustering	72.9	71.3	72.6
StackExchangeClusteringP2P	37.1	43.5	42.9
StackOverflowDupQuestions	54.1	54.7	55.0
SummEval	31.1	31.6	31.3
TRECCOVID	81.4	81.3	82.2
Touche2020	23.3	26.6	24.6
ToxicConversationsClassification	66.9	69.8	68.8
TweetSentimentExtractionClassification	63.7	65.3	64.8
TwentyNewsgroupsClustering	53.4	53.2	53.8
TwitterSemEval2015	81.1	77.5	81.5
TwitterURLCorpus	87.1	86.3	87.0
Average	65.9	64.5	67.0

Table 9: Complete MTEB evaluation results for each dataset. Detailed evaluation metrics and dataset information are available in Muennighoff et al. (2023).

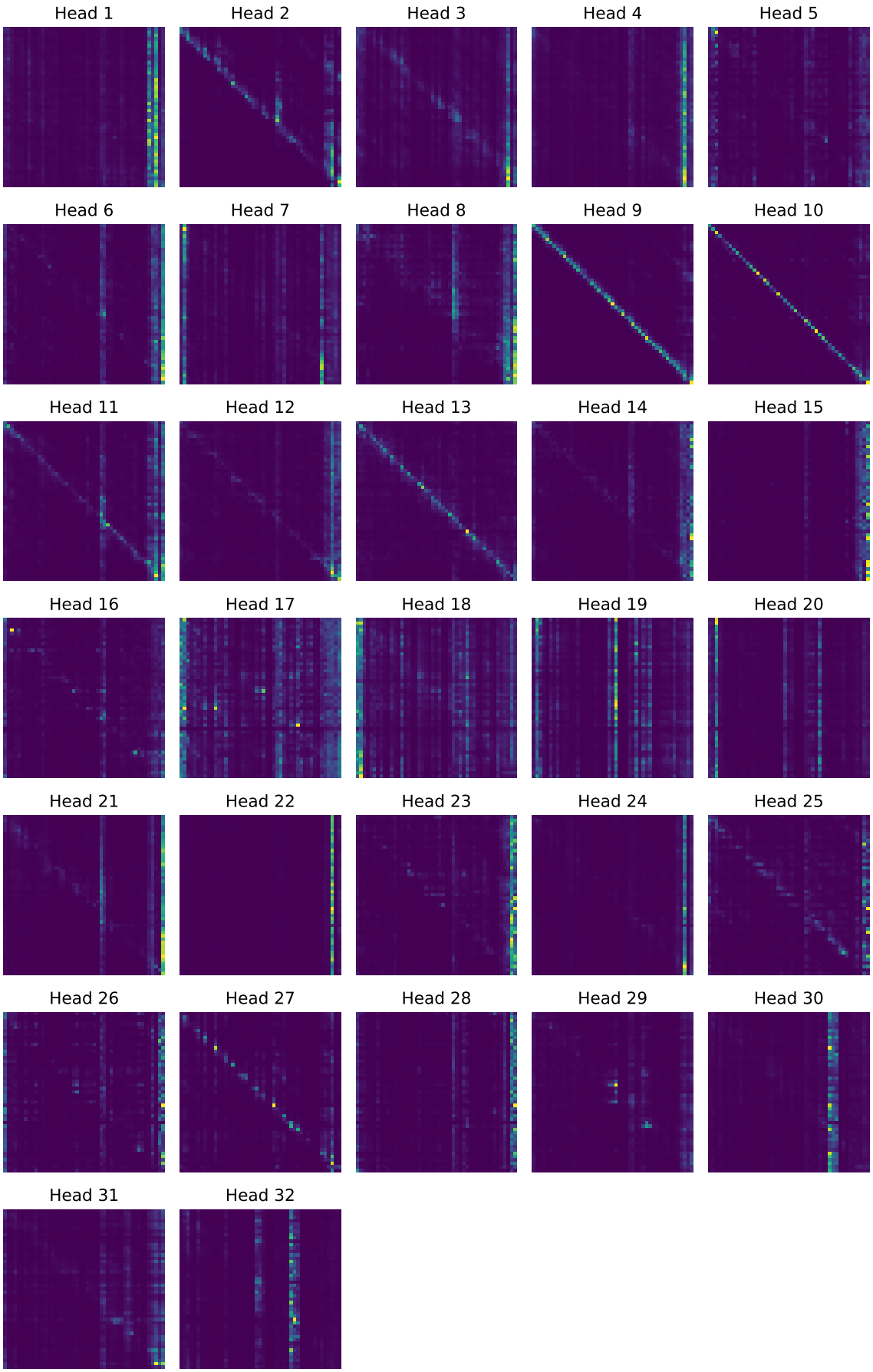

Figure 7: The individual attention matrices from all 32 attention heads in the last layer of the bidirectional model, obtained from a randomly selected example in the STS13 dataset.