HMCL: Task-Optimal Text Representation Adaptation through Hierarchical Contrastive Learning

Zhenyi Wang, Yapeng Jia, Haiyan Ning, Peng Wang, Dan Wang, Yitao Cao
Ant Group
{wangzhenyi.wzy, jyp01659510, ninghaiyan,}
{mingye.wp, wd165820, yudan}@antgroup.com

Abstract

As general large language models continue to advance, their real-world adaptation through effective fine-tuning remains a significant challenge. We introduce Hierarchical Multilevel Contrastive Learning (HMCL), a new contrastive learning framework that improves task-specific text representation for general models. HMCL integrates 3-level semantic differentiation (positive, weak-positive, and negative) and unifies contrastive learning, pair classification, and ranking objectives into a cohesive optimization strategy. HMCL demonstrates exceptional results across multi-domain and multilingual benchmarks, including text similarity, retrieval, reranking and Retrieval-Augmented Generation (RAG) tasks. It outperforms top unsupervised methods and supervised fine-tuning approaches while maintaining broad compatibility with architectures ranging from BERT to Qwen, 330M to 7B. In real-world merchant consultation scenarios, HMCL shows a 0.70-6.24 point improvement over original fine-tuning methods in large-scale base models. This establishes HMCL as a versatile solution that bridges the gap between general-purpose models and specialized industrial applications.

1 Introduction

From the past to the present, text representation has played a fundamental role in natural language processing (Babic et al., 2020). Especially in the era of large language models, as the most crucial part of the RAG process, knowledge retrieval and reranking demand better text representations in specific fields (Gao et al., 2023), which determine the accuracy and completeness of the answers generated by Large Language Models (LLMs). Additionally, it is also central to unsupervised classification, evaluation, and clustering (Patil et al., 2023). In summary, the quality of text representation significantly affects the performance of all fields related to language understanding in artificial intelligence.

Nowadays, with the continuous growth of pre-trained model parameters and the expansion of public datasets, more general models are emerging to address the challenges in text representation. In recent years, successful general base models in this field have become a focal point of research, achieving top scores in public evaluation benchmarks (Chen et al., 2024; Wang et al., 2022; Li et al., 2023; BehnamGhader et al., 2024).

However, even with the success of general models, real-world text representation applications often require modifying these models to better match the specific standards, special expressions, and styles of each distinct task. This is because the definitions of what answers a query, what is considered similar, partially acceptable, or unacceptable can vary across different tasks. To illustrate this, we provide an analysis of seven prominent datasets across semantic textual similarity (STS), retrieval, and reranking tasks in Table 1 (Cer et al., 2017; Marelli et al., 2014; Wu et al., 2020; Li et al., 2025; Wadden et al., 2020; Yang et al., 2018; Muennighoff et al., 2023).

STS measures sentence similarity with 0-5 scores, but STS-B and SICK-R have starkly different standards: STS-B rates positive and negative forms of the same sentence as dissimilar (1.6), while SICK-R assigns a "very similar" score (4.2). Their tolerance for subject specificity (e.g., woman→girl, kids→boys) also differs. In reranking, positive pairs involve deeper associations rather than surface-level semantic similarity. For example, in MindSmallReranking, positives are based on user preferences, making connections of language and knowledge vague, while in StackOverFlowDupQuestions, positives must strictly follow question-answer relationships. In retrieval, dataset differences extend beyond language style and knowledge domains. SciFact requires only a detail in long passages to answer queries; FIQA2018 expects retrieved passages to serve as

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 13495–13518
November 4-9, 2025

STS tasks				
Evaluation Task	**Sentence A**	**Sentence B**	**Score**	**NLI label**
STS-B	a woman is dancing in the rain .	a girl dances in the rain .	3	Entailment
	some people have complete confidence in the people running these institutions .	other people have no confidence at all in the people running these institutions .	1.6	Contradiction
SICK-R	A group of kids is playing in a yard and an old man is standing in the background	A group of boys in a yard is playing and a man is standing in the background	4.5	Neutral
	There is no biker jumping in the air	A lone biker is jumping in the air	4.2	Contradiction
Reranking Tasks				
Evaluation Task	**Query**	**Passage**	**Label**	
MindSmall-Reranking	Donald Trump Jr. reflects on explosive 'View' chat: 'I don't think they like me much anymore'	Opinion: Colin Kaepernick is about to get what he deserves: a chance	Positive	
	Officer placed on leave after threatening teen skaters at gunpoint	Officer placed on leave after threatening teen skaters at gunpoint	Negative	
StackOver-Flow-DupQuestions	String isNullOrEmpty in Java?	Java equivalent of c# String.IsNullOrEmpty() and String.IsNullOrWhiteSpace()	Positive	
	String isNullOrEmpty in Java?	String [] to String	Negative	
Retrieval tasks				
Evaluation Task	**Query**	**Passage**	**Score**	
SciFact	0-dimensional biomaterials show inductive properties.	...Examples include magnetic nanoparticles and quantum dots for stem cell labeling and in vivo tracking...and engineered nanometer-scale scaffolds for stem cell differentiation and transplantation...	1 (Relevant)	
FIQA2018	Having a separate bank account for business/investing, but not a 'business account'?	If it makes your finances easier, why not?...I also have an account to handle transactions for my rental property, and one extra for PayPal use...	1 (Relevant)	
HotpotQA	What science fantasy young adult series, told in first person, has a set of companion books narrating the stories of enslaved worlds and alien species?	The Hork-Bajir Chronicles...The book is introduced by Tobias, who flies to the valley of the free Hork-Bajir, where Jara Hamee tells him the story of how the Yeerks enslaved the Hork-Bajir...	1 (Relevant)	

Table 1: Comparison of labeling standards across prominent public datasets for text embedding evaluation.

suggestions aiding in problem-solving through inference or extension; HotpotQA emphasizes holistic understanding, requiring precise recall of the exact noun description matching the query.

Similar phenomena are more widespread in real-world industrial applications. Therefore, powerful general-purpose models still need to be adapted to specific tasks, learning the language style and labeling standards of those tasks, to ensure optimal performance in specific scenarios.

We introduce HMCL, a novel framework for task-oriented fine-tuning of text embedding models. HMCL extends traditional positive-negative contrastive learning by incorporating multi-hierarchical comparisons, excelling in sentence similarity, reranking, retrieval and RAG tasks. It performs effectively across various base models, including SimCSE, E5, GTE, BGE, and large language models like Qwen-GTE. HMCL consistently outperforms other contrastive learning methods, whether unsupervised or supervised, such as E5's distilled methods (Wang et al., 2022), GTE's improved contrastive loss (Li et al., 2023), or BGE's ANN-style sampling strategy (Xiao et al., 2024; Xiong et al., 2020). Notably, HMCL achieves superior results with small-scale datasets and low computational costs, addressing the challenge of acquiring task-specific fine-tuning data while preserving the model's general capabilities. Relevant materials are available in the project repository: https://github.com/antgroup/hmcl-merchant-question.

2 Related Work

In the past, text embeddings served as low-dimensional vector representations for texts of varying lengths, primarily based on keywords and word understanding. Representative methods from this period include TF-IDF, LSA, word2vec, and the utilization of average or extreme word vec-

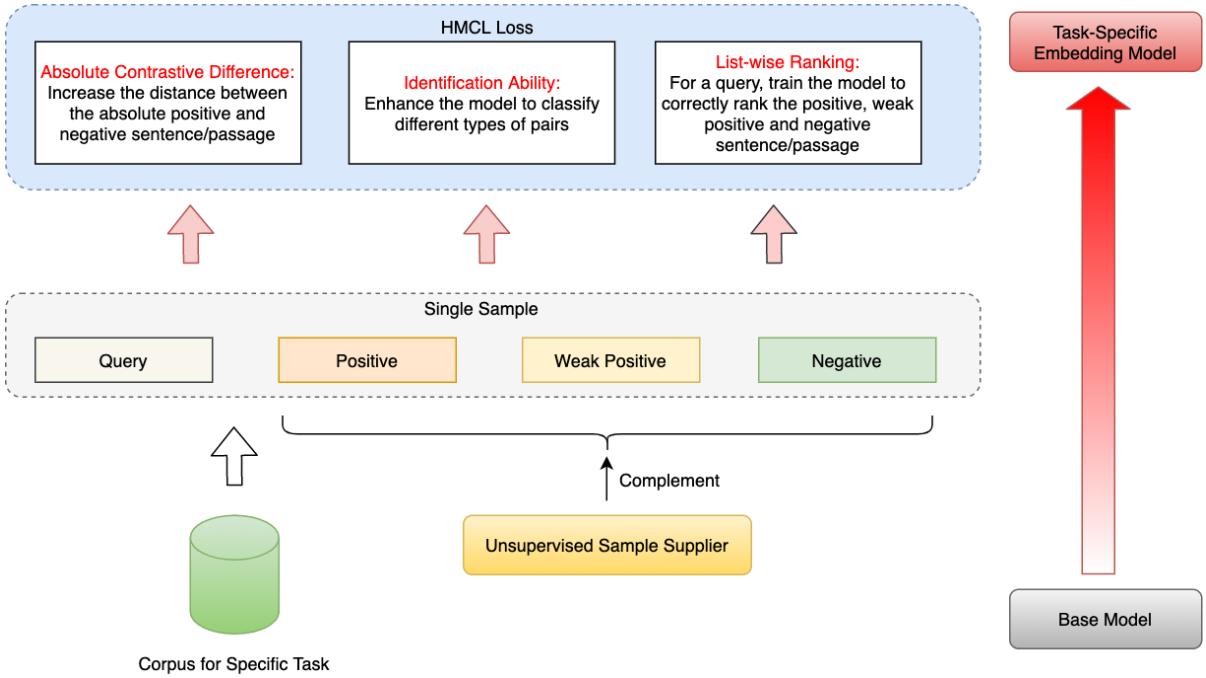

Figure 1: Illustration of HMCL training framework.

tors (Mikolov et al., 2013a; Forgues et al., 2014; Sparck Jones, 1988; Zhixiang et al., 2013; Mikolov et al., 2013b; Deerwester et al., 1990). However, long paragraphs, extensive word dictionaries, and complex contextual meanings in texts demanded more accurate and efficient ways to match relevant passages and sentences. With the advent of pre-trained language models, contrastive learning based on these models has emerged as the most effective method for acquiring semantic representations. This learning framework gained popularity due to SimCLR, which achieved remarkable results in image classification (Chen et al., 2020). By aligning similar pairs and distinguishing unrelated ones during training, this framework successfully generates distinct and meaningful vectors for texts. In the field of text representation, numerous works have proposed ways to enhance positive pairs (such as CERT, ConSERT) or negative pairs (such as SimCSE, ESimCSE) to improve the effectiveness of contrastive learning (Fang and Xie, 2020; Yan et al., 2021; Gao et al., 2021; Wu et al., 2022b).

With the development of comprehensive labeled datasets, many general models trained on very large-scale corpora have become key solutions for text representation. Examples include sentence-T5, E5, INSTRUCTOR, GTE, BGE, and even using LLMs as text encoders (Ni et al., 2021; Wang et al., 2022; Su et al., 2023; Li et al., 2023; Chen et al., 2024; Nie et al., 2024). To some extent, the competition among these general models has shifted to a race for larger data and more extensive training resources. From the era of SimCSE to the current era of LLMs, the number of training pairs has increased from millions to billions, leading to a significant improvement in performance on public benchmarks such as SentEval, MTEB and CMTEB (Conneau and Kiela, 2018; Muennighoff et al., 2023; Xiao et al., 2024).

When general base models achieve excellent performance in text embedding benchmarks, task-specific fine-tuning is still needed to help them achieve optimal performance in real-world industrial scenarios. Although these general models are typically trained using a multi-stage training framework and incorporate improved methods for the second or third training stages, there is still limited research focused on developing effective methods to fully leverage the potential of all kinds of base model to learn the standards of a specific task and perform exceptionally well in that actual scenario.

3 Approach

3.1 Training Framework

Our HMCL training framework, shown in Figure 1, transforms a base model into a task-specific embedding model. In HMCL, each query is associated with three types of comparisons: positive, weak-positive, and negative. The positive item represents

a completely correct match to the query, while the negative item is entirely irrelevant or incorrect. The weak-positive item is partially related to the query: it may not be the best choice to directly answer the query, but it contains useful information, such as related concepts, background context, or incomplete answers. It is more relevant than a negative item but less relevant than a positive one, addressing real-world scenarios where partially relevant or useful content is often selected for a query (see Appendix A.3 for examples of "weak-positive").

In the original 2-level contrastive learning framework, the probability distribution P in the vector space is encouraged to follow:

$$P(a_+ \mid q) > P(a_- \mid q) \tag{1}$$

where P is given by:

$$P(a_j \mid q) = \frac{e^{s(q,a_j)/\tau}}{\sum_k e^{s(q,a_k)/\tau}} \tag{2}$$

Here q represents the query in a sample, a_+ denotes the positive item, a_- denotes the negative item, and a_{w+} denotes the weak-positive item. The function $s(q, a)$ represents the similarity measure between q and a.

Under the assumptions that $s(q, a)$ is normalized (e.g., cosine similarity), the temperature τ is fixed, and optimization converges sufficiently, the condition implies:

$$\frac{P(a_+ \mid q)}{P(a_- \mid q)} = e^{\frac{s(q,a_+)-s(q,a_-)}{\tau}} > 1 \tag{3}$$

and hence

$$s(q, a_+) - s(q, a_-) > 0 \tag{4}$$

This ensures a positive similarity gap, but it may be arbitrarily small. In practice, we expect an ideal text representation model to establish a clear decision boundary between positive and negative pairs, rather than merely producing a continuous similarity ranking. Therefore, we desire a minimal lower bound $r > 1$ on the probability ratio:

$$\frac{P(a_+ \mid q)}{P(a_- \mid q)} \geq r > 1 \tag{5}$$

However, 2-level contrastive learning does not enforce such a lower bound r; it only encourages $P(a_+|q) > P(a_-|q)$ without requiring a minimum separation.

In the HMCL framework, we introduce a weak-positive item a_{w+} whose relevance lies between a_+ and a_-. The desired ordering

$$P(a_+ \mid q) > P(a_{w+} \mid q) > P(a_- \mid q) \tag{6}$$

implicitly imposes stronger constraints on the probability ratios. Specifically, it requires:

$$\frac{P(a_+ \mid q)}{P(a_- \mid q)} > \frac{P(a_{w+} \mid q)}{P(a_- \mid q)} > 1 \tag{7}$$

Therefore, in the 3-level framework, the condition in Equation (5) is more likely to hold during training. This is because Equation (6) establishes an explicit separation between weak positives and negatives, introducing a data-dependent lower bound:

$$r' = \frac{P(a_{w+} \mid q)}{P(a_- \mid q)} > 1 \tag{8}$$

Consequently, we have

$$\frac{P(a_+ \mid q)}{P(a_- \mid q)} = \frac{P(a_+ \mid q)}{P(a_{w+} \mid q)} \cdot r' > r' > 1 \tag{9}$$

which yields an explicit margin in similarity space:

$$s(q, a_+) - s(q, a_-) \geq \tau \log r' > 0 \tag{10}$$

This ensures that strong positives are separated from negatives by a similarity margin of at least $\tau \log r'$, establishing a more reliable decision boundary in the embedding space. Compared to standard 2-level contrastive learning, where no intermediate supervision exists, HMCL leverages the weak-positive item as an anchor to simultaneously pull relevant content closer and push irrelevant content further away. As a result, HMCL encourages more robust and well-structured representations.

In HMCL, the best practice is to use supervised labels to select positive, weak-positive, and negative samples. For example, when similarity scores are available, high-, mid-, and low-scoring items can serve as positives, weak-positives, and negatives, respectively. In a retrieval list, annotators may directly label one fully correct answer, one partially relevant answer, and one entirely incorrect answer. However, complete 3-level tuples are not always obtainable for every query. To address this, we integrate an unsupervised sample supplier that randomly draws from multiple unsupervised modules to fill missing items during training (see Appendix A.2), ensuring the framework remains applicable in less supervised settings.

3.2 HMCL Loss

To accommodate this design, we have developed our HMCL loss, which comprises three components: the absolute contrastive loss L_c, the pair-type classification loss L_e, and the list-wise ranking loss L_l:

$$Loss = \mu_c \cdot L_c + \mu_l \cdot L_l + \mu_e \cdot L_e \tag{11}$$

In this formula, μ_c, μ_l, and μ_e are constants. After conducting ablation experiments, we select $\mu_e = 0.2$, $\mu_l = 1.0$, and $\mu_c = 2.0$ as the standard settings, which are used consistently across all experiments in this paper.

We use the list-wise ranking loss L_l prompts the model to assign similarity scores that reflect the correct ranking. We primarily refer to ListMLE, which is commonly employed in recommendation ranking systems (Xia et al., 2008). If $\widehat{S}$ represents the correct ranking sequence for m items involved in the comparison (where 1, 2, 3, ..., m denotes the correct order) and s represents the actual ranking scores, the probability of the correct sequence $P(\widehat{S})$ is given by:

$$P(\widehat{S}) = \prod_{k=1}^{m} \frac{e^{s_k}}{\sum_{l=k}^{m} e^{s_l}} \tag{12}$$

In our framework, we aim for the model to correctly rank 3-level answers for each sample. Specifically, for every sample, the correct order is such that the positive answer is ranked first, the weak-positive answer is ranked second, and all negative answers are considered tied for third place. Given that the ranking score is determined by the similarity score between the query and the answer, $s(q, a)$, the loss function to optimize the probability of the correct sequence for a batch of samples (with a batch-size of n) can be formulated as follows:

$$L_l = -\frac{1}{n}\sum_{i=1}^{n} log(\frac{e^{s(q_i,a_{+,i})/\tau}}{e^{s(q_i,a_{+,i})/\tau} + Z_1} \cdot \frac{e^{s(q_i,a_{w+,i})/\tau}}{Z_1}) \tag{13}$$

Z_1 represents the sum of the scores from the weak-positive pair and all types of negative pairs:

$$Z_1 = e^{s(q_i,a_{-,i})/\tau} + \sum_{j\neq i}^{n} e^{s(q_i,q_j)/\tau} + \sum_{j\neq i}^{n} e^{s(q_i,a_{+,j})/\tau} + \sum_{j}^{n} e^{s(q_i,a_{w+,j})/\tau} + \sum_{j\neq i}^{n} e^{s(a_{+,i},a_{+,j})/\tau} \tag{14}$$

Besides, we also use the absolute contrastive loss L_c which is adapted from the InfoNCE loss (van den Oord et al., 2018):

$$L_c = -\frac{1}{n}\sum_{i=1}^{n} log\frac{e^{(s(q_i,a_{+,i}))/\tau}}{Z_2} \tag{15}$$

here,

$$Z_2 = e^{s(q_i,a_{-,i})/\tau} + \sum_{j\neq i}^{n} e^{s(q_i,q_j)/\tau} + \sum_{j}^{n} e^{s(q_i,a_{+,j})/\tau} \tag{16}$$

Simultaneously, we employ a cross-entropy loss to enhance the similarity scores generated by the model, thereby effectively distinguishing the three level pairs.

$$L_e = -(\frac{1}{N(pair_+)} \sum_{p=1}^{N(pair_+)} log\frac{e^{z_{p,y_p}}}{\sum_{k=1}^{3} e^{z_{p,k}}} + \frac{1}{N(pair_{w+})} \sum_{p=1}^{N(pair_{w+})} log\frac{e^{z_{p,y_p}}}{\sum_{k=1}^{3} e^{z_{p,k}}} + \frac{\mu_{e,d}}{N(pair_-)} \sum_{p=1}^{N(pair_-)} log\frac{e^{z_{p,y_p}}}{\sum_{k=1}^{3} e^{z_{p,k}}})/n \tag{17}$$

Here, $N(pair)$ denotes the total number of positive, weak-positive, and negative pairs. The variable y_p represents the true label for the three different types of pair levels and $z_{p,k}$ means the logit on the category k of the sample pair p. The constant $\mu_{e,d}$ is set to 0.1 in the standard configuration. In all components of the HMCL loss, the temperature hyperparameter τ (set to 0.05) controls the strength of this effect.

This loss function, by simultaneously optimizing the contrast between positive and negative samples, pair-level classification, and ranking tasks, enables the model to thoroughly learn the hierarchical differences among the three different levels of pairs. Additionally, possible extensions more than the standard mode of our HMCL, such as negative sample expansion, are discussed in Appendix A.6.

Training Method	Avg. STS & SICK	Best on
10 thousand training sentences		
SimCSE	69.71	2
DiffCSE	69.87	1
ESimCSE	68.81	0
InfoCSE	69.65	0
HMCL	**71.51**	4
30 thousand training sentences		
SimCSE	69.95	1
DiffCSE	71.33	0
ESimCSE	71.03	0
InfoCSE	71.73	2
HMCL	**72.21**	4
50 thousand training sentences		
SimCSE	68.28	0
DiffCSE	71.63	0
ESimCSE	72.35	2
InfoCSE	71.86	1
HMCL	**72.87**	4

Table 2: Comparison of the effectiveness of unsupervised training methods for text representation based on BERT_{base} on small-scale training corpora. Results are reported as Spearman's correlations on the benchmarks, calculated using the "all" setting. Training data are constructed by randomly sampling sentences from the NLI corpus.

Training Method	STS-B Test	SICK-R Test
SimCSE	81.71	73.17
SimCSE-ALL	80.48	71.93
HMCL	**82.72**	**75.24**

Table 3: Comparison of the effectiveness of supervised training methods for text representation based on BERT_{base}. Results are reported as Spearman's correlations on the test dataset of benchmarks.

Model	STS-B Test	SICK-R Test	Average
SimCSE-Large	86.68	81.51	84.09
SimCSE-Large-OM	86.71	81.66	84.19
SimCSE-Large-HMCL	**87.47**	**81.97**	**84.72**
GTE-Large	86.07	79.05	82.56
GTE-Large-OM	86.49	79.34	82.91
GTE-Large-HMCL	**87.73**	**80.48**	**84.10**
BGE-Large	85.10	80.59	82.85
BGE-Large-OM	87.09	81.17	84.13
BGE-Large-HMCL	**88.26**	**81.34**	**84.80**
E5-Large	86.26	79.71	82.98
E5-Large-OM	87.33	79.33	83.33
E5-Large-HMCL	**88.40**	**80.62**	**84.51**

Table 4: Comparison of the effectiveness of task-oriented fine-tuning methods for text representation, based on different general base models, in textual similarity tasks. Results are reported as Spearman's correlations on the test dataset of benchmarks.

4 Experiments

4.1 Training embedding model with small dataset

We first evaluate our HMCL framework by applying it to a generic pre-trained model through few-shot training. We randomly sample 10k, 30k, and 50k sentences from the SNLI dataset (Bowman et al., 2015) to form the training sets. The HMCL framework's unsupervised sample supplier is used to augment the samples and train the BERT_{base} model for 5 epochs. As shown in Table 2 and Table A7, our method consistently achieves the highest average scores in few-shot training settings, outperforming classical unsupervised approaches and showing strong potential for rapidly constructing text representation spaces in scenarios with limited data.

Similarly, we conduct supervised training starting from BERT_{base}, using the train datasets for each task and measuring performance on the targeted test datasets. Compared to the SimCSE framework (Table 3), our method shows significant improvement. This is because the comprehensive HMCL loss enhance the semantic discriminability of text representation vectors in training, facilitating the construction of a well-distributed representation space (Figure 2).

Additionally, our HMCL uses a 3-level pair contrast, while other frameworks rely on a 2-level contrast (positive/negative). To ensure a fair comparison, we process SimCSE's training data in two ways: one excludes weak-positive sentences (SimCSE), and the other treats weak-positives as hard-positives (SimCSE-ALL). However, SimCSE-ALL performs significantly worse than SimCSE, indicating that weak-positives cannot be directly utilized without a well-designed framework. As a result, we exclude weak-positive samples when reproducing other 2-level contrast methods in subsequent sections.

4.2 Task-oriented fine-tune for different general base models

In the subsequent experiments, we utilize several classic text representation general models that have been pretrained on large datasets as our base models, including SimCSE-Large, E5-Large, BGE-Large, and GTE-Large. We fine-tune these base models using both their recommended original fine-

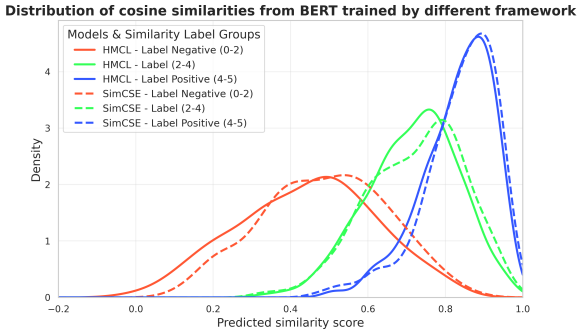

Figure 2: Comparison of cosine similarity score distributions in the STS-B Test between BERT_{base} models trained with the HMCL framework and with the 2-level contrastive framework SimCSE under supervised conditions.

tuning methods and parameters, as well as our proposed HMCL fine-tuning method. The training data for each task is the same across all models and methods. We then compare the performance of the original base models, the models fine-tuned with the original methods (denoted as -OM: original method), and the models fine-tuned with the HMCL framework (denoted as -HMCL) on the respective tasks.

Table 4 presents the fine-tuning results of the four models on the textual similarity task. It is evident that, regardless of the base model or the fine-tuning method, task-targeted fine-tuning significantly improves the model's performance on the test evaluation set. Notably, the HMCL framework outperforms other fine-tuning methods, with a general improvement of 0.62 to 1.95 points over the original base models and 0.53 to 1.19 points over the models fine-tuned with the original methods. This demonstrates the superior effectiveness of our approach in similarity comparison tasks, applicable to a wide range of base models.

Furthermore, we conduct preliminary validation of our HMCL framework on common text representation tasks, such as reranking and retrieval, as shown in Table 5. For this experiment, we select 6 tasks from the MTEB benchmark with training sets. We also include 2 CMTEB biomedical tasks and using data from the MedicalRetrieval dataset to contrust their training data. For each task, all methods use the same data, allowing us to compare the performance of HMCL and other fine-tuning methods on 4 reranking and 4 retrieval tasks.

Our results show that the HMCL framework consistently achieve the highest average scores across all base models and outperform other methods on a greater number of tasks. The average score improvements over the original fine-tuning methods ranged from 0.48 to 4.91 points, indicating that our approach is also effective for targeted fine-tuning in reranking and retrieval tasks.

To verify the overall stability of the model's performance after targeted fine-tuning using the HMCL framework, we also evaluated the model on the entire MTEB benchmark of 41 tasks after fine-tuning it on small training sets from 3 tasks (FIQA2018, STS-B, and SICK-R). We found that the overall performance of the fine-tuned model remained stable. While there was an improvement in the STS task metrics, the performance on other types of tasks did not show significant degradation, with the overall average score remaining largely unchanged (Table A8). This demonstrates that the HMCL fine-tuning with small dataset enhances the performance on specific tasks while still preserving the model's strong general capabilities across other types of tasks.

While the experiments conducted on the widely-used MTEB benchmark demonstrate the generalizability of our approach, they do not fully validate its effectiveness in real-world applications. This is because publicly available benchmarks, characterized by high-quality corpora and annotations, are extensively used for training various general models, particularly large-scale ones (Ni et al., 2024). In selecting our experimental tasks, we exclude those that are widely used for general model training, such as FEVER and MSMARCO (Wang et al., 2022; Li et al., 2023). However, as reported in the technical report of BGE-M3 (Chen et al., 2024), plenty of the MTEB and CMTEB evaluation tasks appear in the training data. The prevalent use of public corpus for general model training may explain the limited benefits observed in task-specific fine-tuning (regardless of the fine-tuning method employed) for some base models in Table 5.

Therefore, to showcase the effectiveness of the HMCL framework on larger general models, we construct two new evaluation datasets, MerchantQuestion-S2S and MerchantDocs-S2P, along with their corresponding training data, from a real-world industrial system for merchant consultation and Q&A. MerchantQuestion-S2S is designed for a textual similarity task (scoring related sentences based on a query), while MerchantDocs-S2P is a reranking task (ordering relevant documents based on a query). The results in Table 6 indicate that task-specific fine-tuning on these

Model	CMedQAV-1	CMedQAV-2	MSR	SOFDQ	FIQA2018	HotpotQA	NFCorpus	SciFact	Average	Best on
SimCSE-Large	13.84	14.31	30.35	38.56	18.72	28.65	17.53	34.73	24.59	0
SimCSE-Large-OM	9.89	19.95	27.82	48.95	37.01	42.18	25.84	58.98	33.83	0
SimCSE-Large-HMCL	**27.00**	**30.33**	**31.44**	**49.68**	**37.14**	**43.42**	**26.42**	**64.46**	**38.74**	8
E5	67.64	66.78	31.42	49.72	43.80	**71.23**	34.00	70.41	54.38	1
E5-Large-OM	68.30	66.32	**33.28**	54.40	43.30	63.20	34.11	73.13	54.51	1
E5-Large-HMCL	**74.69**	**76.14**	32.78	**54.67**	**45.91**	65.20	**34.23**	**74.52**	**57.27**	6
GTE-Large	81.98	82.69	32.63	53.63	44.50	**67.16**	**38.17**	74.27	59.38	2
GTE-Large-OM	85.29	85.84	31.79	55.48	44.10	60.41	36.18	**76.73**	59.48	1
GTE-Large-HMCL	**86.86**	**87.24**	**32.05**	**55.68**	**44.93**	62.87	37.90	75.10	**60.33**	5
BGE-Large	81.88	**84.29**	30.99	55.24	44.99	**74.64**	34.57	72.37	59.87	2
BGE-Large-OM	**82.03**	82.45	**32.76**	53.04	45.07	68.15	34.91	**77.04**	59.43	3
BGE-Large-HMCL	81.41	83.69	31.38	**56.21**	**45.50**	67.77	**38.23**	75.12	**59.91**	3

Table 5: Comparison of the effectiveness of task-oriented fine-tuning methods for text representation, based on different general base models, in MTEB reranking and retrieval tasks. Results are reported as nDCG@10 for retrieval tasks and MAP for reranking tasks. Note: MSR here means MindSmallReranking, SOFDQ means StakOverFlowDupQuestions.

Model	MerchantQuestions-S2S	MerchantDocs-S2P
BGE-Large	69.87	67.41
BGE-Large-OM	73.26	72.70
BGE-Large-HMCL	**74.23**	**75.81**
BGE-M3	71.06	69.07
BGE-M3-OM	71.09	69.94
BGE-M3-HMCL	**73.69**	**76.18**
Qwen-GTE-1.5B	67.65	69.31
Qwen-GTE-1.5B-OM	68.10	72.66
Qwen-GTE-1.5B-HMCL	**70.53**	**73.36**
Qwen-GTE-7B	64.79	69.66
Qwen-GTE-7B-OM	69.94	73.71
Qwen-GTE-7B-HMCL	**71.68**	**75.16**

Table 6: Comparison of the effects of fine-tuning large-scale text representation base models using HMCL and the original method on a real-world dataset for merchant business consulting. Results are reported as Spearman's correlation for S2S task and MAP for S2P task.

	Finance	Movie	Music	Open	Sports	Total
BGE-M3	23.60	42.71	51.40	58.02	**35.38**	40.95
BGE-M3-SimCSE	24.46	44.17	52.51	57.84	33.79	41.29
BGE-M3-E5	23.60	45.15	49.44	56.72	33.39	40.80
BGE-M3-GTE	**25.70**	**45.63**	52.23	58.02	32.81	41.74
BGE-M3-BGE	24.61	45.15	52.23	**59.51**	34.58	42.00
BGE-M3-HMCL	24.92	45.47	**52.79**	58.58	34.98	**42.12**

Table 7: Comparison of RAG performance using different fine-tuned BGE-M3 models (with fine-tuning frameworks: -SimCSE, -E5, -GTE, -BGE, -HMCL) evaluated on the CRAG dataset. Scoring: 1 for fully correct, 0.5 for partial, and 0 for incorrect answers. Average scores (multiplied by 100) are compared.

new, data-leak-free tasks yields significant improvements, with performance gains of up to 8.40 points. This underscores the necessity of such fine-tuning steps in practical industrial applications, even when employing powerful base models.

Compared to the original fine-tuning method, our HMCL consistently outperforms, achieving improvements of 0.98 to 2.59 points on the S2S task and 0.70 to 6.24 points on the S2P task. These results convincingly demonstrate the broad applicability and superiority of our HMCL framework in fine-tuning base models.

Additionally, we compared HMCL with other fine-tuning frameworks on RAG tasks, as shown in Table 7. Using the same generative model and BGE-M3 models fine-tuned with various frameworks as retrievers, we retrieved the top-3 passages for each query from CRAG's web snapshots. The consistency of the generated answers with the reference answers was then evaluated. The results demonstrate that the HMCL-finetuned model achieves the greatest overall improvement for RAG, enhancing generation correctness by 1.17 points compared to the original BGE-M3 and outperforming all other fine-tuning frameworks.

4.3 Ablation studies

Compared with conventional contrastive learning, HMCL extends the binary positive-negative scheme to a 3-level hierarchy, enriching the learn-

Similar Pair Level	Avg. STS
Positive/Negative	71.78
Positive/Weak-Positive/Negative	**73.41**
Positive/Weak-Positive/Weak-Negative/Negative	71.48

Table 8: Training performance in the HMCL framework using 2-level pair samples, 3-level pair samples, and 4-level pair sample. Results are reported as Spearman's correlations on the benchmarks, calculated using the "all" setting. Note: The weak-negative here means the sample that is less relevant than the positive but more relevant than the strong-negative. For specific explanations and examples, see Appendix A.3.

Loss Construction	STS-B Test
Standard Mode ($\mu_e = 0.2, \mu_{e,d} = 0.1, \mu_l = 1.0, \mu_c = 2.0$)	**82.72**
Changing μ_l	
$\mu_l = 0.0$	80.97
$\mu_l = 0.5$	82.59
$\mu_l = 2.0$	81.92
Changing μ_c	
$\mu_c = 0.0$	82.45
$\mu_c = 1.0$	82.49
$\mu_c = 3.0$	82.60
Changing μ_e	
$\mu_e = 0.0$	82.29
$\mu_e = 1.0$	82.19
Changing $\mu_{e,d}$	
$\mu_{e,d} = 0.0$	82.28
$\mu_{e,d} = 0.5$	82.61

Table 9: Ablation study on the optimal values of contribution coefficients that control the proportion of three tasks in the HCML framework. Results are reported as Spearman's correlation on the test dataset.

ing signal. To examine whether additional levels improve performance, we conduct an ablation study (Table 8) varying the number of levels while keeping the HMCL loss components unchanged, including absolute contrastive loss, ranking loss, and classification loss. Results show that the 3-level setting outperforms the 2-level baseline, but performance drops with 4 levels (73.41 vs. 71.48). This indicates that overly fine-grained ranking is impractical: beyond 3 levels it becomes harder to maintain meaningful gradient differences and obtain high-quality samples, while relevance boundaries blur, reducing positive-negative separation.

As noted, the HMCL loss combines three components weighted by contribution coefficients. Ablation results in Table 9 show that removing any component degrades performance, especially the ranking loss. Overemphasizing any single term also harms results. Based on these findings, we recommend specific coefficient values for the standard mode, which are adopted in all other experiments in this paper.

5 Conclusion

We propose HMCL, a novel fine-tuning framework that enhances task-specific text representations. By integrating 3-level semantic differentiation and unifying multiple learning objectives, HMCL achieves outstanding performance across diverse benchmarks and various model architectures. It demonstrates significant gains in real-world applications, such as a merchant consultation, bridging the gap between general-purpose models and specialized tasks.

Limitations

HMCL is a task-oriented fine-tuning framework that achieves strong results across diverse tasks and base models, but several limitations remain.

First, the presence of influential public benchmarks in the pretraining data of recent high-performing base models risks data leakage, making it difficult to assess the real gain from task-specific fine-tuning. Moreover, many text representation tasks lack dedicated training sets, limiting evaluation scope. To address this, we created two new benchmarks for fairer comparison, but their construction was costly, labor-intensive, and domain-limited, restricting further expansion.

Second, HMCL relies on a fixed 3-level contrastive hierarchy. Experiments with 4-level structures consistently underperformed—sometimes even trailing the 2-level baseline—due to the difficulty of defining clear relevance tiers and obtaining reliable supervision beyond three levels. Thus, three levels appear to be a practical upper bound in our setting rather than a tunable parameter.

Finally, regarding bias and fairness, our unsupervised data augmentation method recombines existing samples without adding new subjective opinions, thereby avoiding new bias introduction. However, if future applications use human- or LLM-generated weak positives, cultural or semantic biases may emerge or be amplified. Possible mitigations include neutral label descriptors, bias-aware annotation guidelines, diverse annotator pools, subpopulation audits (e.g., FPR/TPR disparities), and data rebalancing when necessary.

Acknowledgments

This research was supported by Ant Group. We extend our sincere gratitude to the anonymous reviewers for their thoughtful suggestions and comments, which have greatly improved the quality of this paper.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263, Denver, Colorado. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 81–91, Dublin, Ireland. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA. Association for Computational Linguistics.

Eneko Agirre, Aitor Gonzalez-Agirre, Iñigo Lopez-Gazpio, Montse Maritxalar, German Rigau, and Larraitz Uria. 2016. SemEval-2016 task 2: Interpretable semantic textual similarity. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 512–524, San Diego, California. Association for Computational Linguistics.

Karlo Babic, Sanda Martincic-Ipsic, and Ana Mestrovic. 2020. Survey of neural text representation models. *Inf.*, 11(11):511.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. LLM2vec: Large language models are secretly powerful text encoders. In *First Conference on Language Modeling*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference.

Daniel Matthew Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *International Workshop on Semantic Evaluation*.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljacic, Shang-Wen Li, Scott Yih, Yoon Kim, and James Glass. 2022. DiffCSE: Difference-based contrastive learning for sentence embeddings. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4207–4218, Seattle, United States. Association for Computational Linguistics.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R.A. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science 41*, pages 391–407.

Hongchao Fang and Pengtao Xie. 2020. CERT: contrastive self-supervised learning for language understanding. *CoRR*, abs/2005.12766.

Gabriel Forgues, Joelle Pineau, Jean-Marie Larchevêque, and Réal Tremblay. 2014. Bootstrapping dialog systems with word embeddings. In *Nips, modern machine learning and natural language processing workshop*, volume 2.

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.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *ArXiv*, abs/2312.10997.

Xiangyang Li, Kuicai Dong, Yi Quan Lee, Wei Xia, Hao Zhang, Xinyi Dai, Yasheng Wang, and Ruiming Tang. 2025. Coir: A comprehensive benchmark for code information retrieval models. *Preprint*, arXiv:2407.02883.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *Preprint*, arXiv:2308.03281.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013b. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations*.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2023. Mteb: Massive text embedding benchmark. *Preprint*, arXiv:2210.07316.

Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. 2018. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *Preprint*, arXiv:1707.09564.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B. Hall, Daniel Cer, and Yinfei Yang. 2021. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *Preprint*, arXiv:2108.08877.

Shiwen Ni, Xiangtao Kong, Chengming Li, Xiping Hu, Ruifeng Xu, Jia Zhu, and Min Yang. 2024. Training on the benchmark is not all you need. *Preprint*, arXiv:2409.01790.

Zhijie Nie, Zhangchi Feng, Mingxin Li, Cunwang Zhang, Yanzhao Zhang, Dingkun Long, and Richong Zhang. 2024. When text embedding meets large language model: A comprehensive survey. *Preprint*, arXiv:2412.09165.

Rajvardhan Patil, Sorio Boit, Venkat Gudivada, and Jagadeesh Nandigam. 2023. A survey of text representation and embedding techniques in nlp. *IEEE Access*, 11:36120–36146.

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 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Karen Sparck Jones. 1988. *A statistical interpretation of term specificity and its application in retrieval*, page 132–142. Taylor Graham Publishing, GBR.

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. *Preprint*, arXiv:2212.09741.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. *Preprint*, arXiv:2004.14974.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *ArXiv*, abs/2212.03533.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, editor = "Jurafsky Dan Zhou, Ming", Joyce Chai, Natalie Schluter, and Joel Tetreault. 2020. Mind: A large-scale dataset for news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606, Online. Association for Computational Linguistics.

Xing Wu, Chaochen Gao, Zijia Lin, Jizhong Han, Zhongyuan Wang, and Songlin Hu. 2022a. InfoCSE: Information-aggregated contrastive learning of sentence embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3060–3070, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xing Wu, Chaochen Gao, Liangjun Zang, Jizhong Han, Zhongyuan Wang, and Songlin Hu. 2022b. ESimCSE: Enhanced sample building method for contrastive learning of unsupervised sentence embedding. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3898–3907, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise approach to learning to rank: theory and algorithm. In *International Conference on Machine Learning*.

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.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *Preprint*, arXiv:2007.00808.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. ConSERT: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long*

Papers), pages 5065–5075, Online. Association for Computational Linguistics.

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Daniel Gui, Ziran Will Jiang, Ziyu Jiang, Lingkun Kong, Brian Moran, Jiaqi Wang, Yifan Ethan Xu, An Yan, Chenyu Yang, Eting Yuan, Hanwen Zha, Nan Tang, Lei Chen, Nicolas Scheffer, Yue Liu, Nirav Shah, Rakesh Wanga, Anuj Kumar, Wen tau Yih, and Xin Luna Dong. 2024. Crag – comprehensive rag benchmark. *arXiv preprint arXiv:2406.04744*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *Preprint*, arXiv:1809.09600.

Zhixiang, Xu, Minmin Chen, Kilian Q. Weinberger, and Fei Sha. 2013. An alternative text representation to tf-idf and bag-of-words. *Preprint*, arXiv:1301.6770.

A Appendix

A.1 Experimental settings

For the experiments in this paper, the training configurations of HMCL are shown in Table A1. The reported experiments consist of two categories: the first involves directly training pre-trained models (e.g., BERT) that are not specifically adapted to text representation tasks into text representation models using unsupervised or supervised methods (Tables 2-3, 7-8); the second involves task-specific fine-tuning for adaptation tasks based on pre-trained foundational text representation models (e.g., SimCSE-Large, BGE, GTE) (Tables 4-6). The learning rates and batch sizes differ between these two types of tasks, as detailed in the tables. All experiments are conducted on A100 GPUs, with the number of GPUs used depending on the parameter size of the base model and the text sequence length, ranging from 1 to 6 GPUs (e.g., 1 A100 for 110M parameter-level models, 2-3 for 300M parameter models, and 4-6 for models with >1B parameters). Information of HMCL parameters have mentioned in Approach and except for the ablation study, the standard mode of parameters are used in all the experiments.

For the experiments in Table 2, we compare the performance differences between recent prominent unsupervised text representation training methods and our proposed approach based on BERT$_{base}$. To ensure a fair comparison, all methods are trained strictly using their corresponding official open-source code and default parameter configurations from the provided shell scripts. Relevant implementation details are listed in Table A2. For the unsupervised experiments training from BERT$_{base}$, we train with all methods for 5 epochs and report the best results based on the evaluation conducted at the end of each epoch fairly in a continuous single run. For the supervised training in Table 3, we train the models for 10 epochs.

Table A3 provides detailed statistics for all unsupervised evaluation tasks from STS-12 to SICK-R. Since these evaluation datasets have many splits, we calculated the final results using a Spearman's correlation based on the unified ranking of all samples within each task (Reimers and Gurevych, 2019) (rather than averaging the results across different splits).

HMCL Training Settings	
Batch size (training from BERT)	96
Batch size (supervise)	32
Learning rate (training from BERT)	3.00E-05
Learning rate (fine-tune)	1.00E-05
GPU	NVIDIA A100 Tensor Core GPU

Table A1: Training settings of HMCL in experiments.

Unsupervised training method	**Learning rate**	**Pooler type**	**Official training code**
SimCSE	3.00E-05	cls	https://github.com/princeton-nlp/SimCSE
ESimCSE	3.00E-05	cls	https://github.com/caskcsg/sentemb/tree/main/ESimCSE
InfoCSE	7.00E-06	cls	https://github.com/caskcsg/sentemb/tree/main/InfoCSE
DiffCSE	7.00E-06	cls	https://github.com/voidism/DiffCSE

Table A2: Training Parameters and Source Code of Common Unsupervised Training Methods for Text Representation (Gao et al., 2021; Wu et al., 2022b,a; Chuang et al., 2022)

Task name	**Number of splits**	**Number of total evaluation pairs**
STS-12	5	3108
STS-13	3	1500
STS-14	6	3750
STS-15	5	3000
STS-16	5	1189
SICK-R	5	9927
STS-B	3	8356

Table A3: Statistics for the unsupervised evaluation tasks (Agirre et al., 2012, 2013, 2014, 2015, 2016; Cer et al., 2017; Marelli et al., 2014)

Base model name	Model in HuggingFace	Parameter counts	Proposed learning rate in original method	Pooler type	Prompt	Official training code
SimCSE-Large	sup-simcse-roberta-large	355M	1.00E-05	cls	/	https://github.com/princeton-nlp/SimCSE
GTE-Large	gte-large (for English task) gte-large-zh (for Chinese task)	335M 326M	5.00E-05	cls	/	/
E5-Large	multilingual-e5-large	330M	1.00E-05	last average pooling	Official prompt	https://github.com/microsoft/unilm/tree/master/e5
BGE-Large	bge-large-en (for English task) bge-large-zh (for Chinese task)	335M 326M	1.00E-05	cls	Official prompt	https://github.com/FlagOpen/FlagEmbedding
BGE-M3	bge-m3	560M	1.00E-05	cls	/	https://github.com/FlagOpen/FlagEmbedding/tree/master/FlagEmbedding/BGE_M3
Qwen-GTE-1.5B	gte-Qwen2-1.5B-instruct	1.78B	/	last-token pooling	Official prompt	/
Qwen-GTE-7B	gte-Qwen2-7B-instruct	7.61B	/	last-token pooling	Official prompt	/

Table A4: Model specifications, fine-tuning configurations, and official codebases of base models (Gao et al., 2021; Wang et al., 2022; Li et al., 2023; Xiao et al., 2024; Chen et al., 2024)

Task name	Language	Test split rows	Fine-tuning data rows	Source of fine-tuning data
STS-B Test	English	1379	13427	Train and Dev dataset of STS-B
SICK-R Test	English	4927	4802	Train dataset of SICK-R
CMedQAV-1	Chinese	1000	1000	MedicalRetrieval dataset
CMedQAV-2	Chinese	1000	1000	MedicalRetrieval dataset
FiQA2018	English	1706	14166	Train dataset of FiQA2018
HotpotQA	English	14810	50000	Train dataset of HotpotQA
MindSmallReranking	English	141876	50000	Train dataset of MindSmallReranking
NFCorpus	English	12334	50000	Train dataset of NFCorpus
SciFact	English	339	919	Train dataset of SciFact
StackOverflowDupQuestions	English	2992	19847	Train dataset of StackOverflowDupQuestions
CRAG	English	1332	1366	Dev dataset of CRAG-Task1

Table A5: Supervised fine-tuning task training dataset information (Cer et al., 2017; Marelli et al., 2014; Conneau and Kiela, 2018; Muennighoff et al., 2023; Xiao et al., 2024; Yang et al., 2024)

In the fine-tuning experiments of public datasets, we conduct experiments using various base models and obtain the best results within 3 epochs (However, many public corpus may have been used in the pre-training of some base models, the best performance is usually achieved in the first epoch of fine-tuning). Details such as the correspondence between model names in the article and Hugging Face models, training parameters, and official code references appear in Table A4. For base models with language-specific versions, we employ dedicated language models for English and Chinese tasks respectively (e.g., gte-large-zh for Chinese and gte-large-en for English). We use the HMCL's learning rate of 1e-5 as the default learning rate because it is the same as the learning rate officially recommended by most models (For GTE, the official recommended learning rate for the pre-training phase is provided, but the parameters for the fine-tuning phase are missing. Beside, this recommended learning rate is much higher than others and does not yield good results). We use the training method from the fine-tuning stage (usually the final training phase) as the baseline for comparison with HMCL. For models without official code available, we faithfully reproduce methods based on original paper descriptions. When using the corresponding models, the pooler type and prompt settings of the original base model, the original method fine-tuning, and HMCL fine-tuning are kept entirely consistent, ensuring absolute fairness.

The detailed training data for each fine-tuning task using public evaluation datasets is provided in Table A5. For each task, the training dataset is composed of randomly sampled and processed data from the task's train dataset or similar domain

Task Name	Train dataset	Test dataset	Example (translated in English) Query	Positive	Negative	Weak-positive
MerchantQuestions-S2S	1186 pairs	591 pairs	What is the calculation methodology for total user assets?	How is the total user assets calculated?	How is "merchant's total visiting user count" defined?	What does "total user assets" mean?
MerchantDocs-S2P	5053 rows	7581 rows	Will unused new customer red packets in live streaming rooms be automatically refunded?	# ## Pre-Usage Instructions Live stream red packets are configured by merchants through the streaming backend (including amount, quantity, target audience, etc.) and distributed to viewers as cash incentives. These red packets help enhance fan engagement, viewing duration, and interaction within live streams. ## Red Packet Specifications 1. Red packets become claimable 5 minutes after being pushed to the live stream room (countdown starts from packet generation);\2. Minimum total amount per distribution: CNY 10. Minimum individual packet: CNY 0.1. Maximum single distribution: CNY 5,000. Daily cap: CNY 50,000;\3. Unclaimed balances will be refunded through original payment channels within 24 hours;\4. Phase I red packet deductions will be drawn from enterprise ***accounts: (Accounts must support balance payments. Ensure sufficient **[account balance]** before distribution. Note: Fan accounts (i.e., personal ***accounts) currently lack balance payment functionality and cannot distribute red packets. Verify account type in advance);\5. Exercise caution when granting operational permissions to prevent financial security risks from credential misuse. ## Primary Administrator Authorization Configurations	Can claimed red packets (originally termed "cash red packets") be refunded? # # Context Merchants seek to recall distributed red packets after discovering operational errors or other issues post-user claim. # # Refund Eligibility Regrettably, refunds cannot be processed for red packet amounts already deposited into users' *** accounts.\If merchants wish to reclaim such funds, they must directly negotiate with users through private channels. # 	/

Table A6: Information Overview and Sample Examples of MerchantQuestions-S2S and MerchantDocs-S2P

datasets. The processing logic is as follows:

1. In the semantic textual similarity task with a five-point scale, similarity scores of 4 or higher are considered positive, scores between 2 and 4 are considered weak-positive, and scores below 2 are considered negative.

2. In reranking and retrieval tasks, positive and negative labels are directly mapped to each row. weak-positive samples are randomly augmented using unsupervised methods provided by the Unsupervised Sample Supplier (described in Section A.2), including random shuffling, random deletion, random insertion, number replacement, syntax-aware degradation, and positive-negative mixing.

To ensure efficient fine-tuning experiments, we use fewer than 50,000 rows of processed data as the training set for each task. All training methods (HMCL and the original recommended fine-tuning methods) use the same training datasets and data rows. The only difference is that HMCL is a 3-level contrastive framework, which can utilize weak-positive data (if available) during training, while other frameworks cannot. Through ablation experiments in Table 3, we handle similar 2-level contrastive frameworks by removing the weak-positive column but keeping the total number of training rows unchanged (this approach yields better training performance than treating weak-positive data as positive). This ensures a fair comparison.

Besides, we additionally constructed an experiment to evaluate the effectiveness of RAG using the public CRAG dataset. In this experiment, we first processed the data from the dev set according to the original split of the CRAG dataset. The processing method is as follows:

For the top 5 webpage snapshots associated with each query, we used an LLM (Qwen-72b) to label the content and invited human annotators to re-check whether "the current paragraph contains content that directly answers the question." Those labeled as "contains" were treated as positive items, those labeled as "does not contain" were treated as weak-positive items, and one randomly selected webpage snapshot completely unrelated to the current query was used as a negative item. If, after organizing the data according to the above rules, there were still queries missing positive items (i.e., no content directly answering the question), we randomly selected one paragraph labeled as "does not contain" to fill the position of the positive item. If there were missing weak-positive items, we randomly selected one "contains" paragraph and one "unrelated" paragraph, combining them to serve as a weak-positive item.

The data organized in the above manner was used as the training set. We fine-tuned the BGE-M3 model using different frameworks for evaluation, with all frameworks utilizing the exact same training data.

The test set data from the original CRAG split was used as the evaluation set. During the experiment, each round involved loading a text representation model as the retriever to retrieve the top 3 most relevant webpage snapshots from the full set of webpage snapshots for each question. These retrieved results were then passed to a generative model (here, we consistently used qwen2.5-72b-instruct as the generative model) for answer generation. After generating the answers, they were evaluated alongside the standard answers provided by CRAG using a referee model (here, we consistently used bailing-80b as the referee model). The differences between the generated answers and the standard answers were scored as follows: a fully consistent answer received 1 point, a partially consistent answer received 0.5 points, and an inconsistent answer received 0 points. Finally, the scores for all questions were averaged to obtain the final score.

The prompt used for answer generation in the RAG task is as follows (completely adopting the official prompt from Task 1 of the CRAG evaluation set):

You are given a Question, References and the time when it was asked in the Pacific Time Zone (PT), referred to as "Query Time". The query time is formatted as "mm/dd/yyyy, hh:mm:ss PT". The references may or may not help answer the question. Your task is to answer the question in as few words as possible.

Please follow these guidelines when formulating your answer:

1. If the question contains a false premise or assumption, answer "invalid question".

2. If you are uncertain or don't know the answer, respond with "I don't know".

Question
{question}
Query Time
{query_time}
References
{contexts}

The prompt used for evaluation in the RAG task is as follows (originally in Chinese in the code,

translated here):

 # Task Description

You are a professional fact consistency evaluation assistant. Please strictly compare the "Reference Answer" with the "Generated Answer" and assess the degree of conformity based on the following three levels:

*1. **Fully Consistent**: All key facts in the generated answer are completely consistent with the reference answer, with no errors or omissions.*

*2. **Partially Consistent**: The generated answer contains some correct facts but also includes errors or omits critical information.*

*3. **Inconsistent**: The generated answer contains critical factual errors or is entirely unrelated to the reference answer.*

 # Input Format

"Question": "[Question text]",

"Reference Answer": "[Standard answer text]",

"Generated Answer": "[Text to be evaluated]"

 # Output Requirements

1. Include the following fields:

- "Evaluation Level": "Fully Consistent/Partially Consistent/Inconsistent"

- "Reason": "The rationale for selecting the evaluation level"

2. Evaluate strictly based on the provided text content; do not introduce external knowledge.

3. Recognize semantically equivalent expressions (e.g., "50%" and "half" should be considered consistent).

 # Example Input:

"Question": "What is the main transmission route of the novel coronavirus, and what is its incubation period?",

"Reference Answer": "The novel coronavirus mainly spreads through droplets, with an incubation period typically ranging from 1 to 14 days.",

"Generated Answer": "The virus spreads through the air, and symptoms may appear within about two weeks after infection."

Output: "Evaluation Level": "Partially Consistent", "Reason": "The generated answer correctly identifies the incubation period (14 days/two weeks) but inaccurately describes the transmission route ('airborne' vs. 'droplets' in the reference answer)."

 # Start Evaluation Input:

"Question": {question},

"Reference Answer": {reference},

"Generated Answer": {generated}

Output:

Besides, detailed information about our custom evaluation datasets will be provided in Section A.4.

A.2 Unsupervised sample supplier

In the HMCL training framework, to extend each input sample into a three-dimensional space of positive/weak-positive/negative, we design an unsupervised sample supplier composed of multiple generation methods. As described in the main text, this module can integrate various unsupervised approaches and randomly selects one during generation. The implemented methods that used in experiments include:

1. Positive Sample Generation

a. Random Repetition: Repeat words in the input query N times, where $N \in [1, 0.3 \times Len(words)]$. Words are randomly selected and duplicated in their original positions.

b. Random Insertion of Neutral Words: Insert a semantically neutral word (e.g., "the", "so", "a") or punctuation into the query.

c. Syntax-aware Paraphrasing: Based on syntax tree analysis:

- Reposition adverbs (requires ≥ 1 adverb in the sentence)
- Convert between simple present and present continuous tenses
- Remove copula verbs (if present)

If no conditions are met, apply random repetition.

2. Weak-Positive Sample Generation

a. Random Shuffling: Fully randomize word order or swap word pairs 7 times.

b. Random Deletion: Delete $N \in [1, 4]$ words (activated only if $Len(words) > 8$).

c. Random Insertion: Insert $N \in [2, 5]$ [MASK] or [UNK] tokens.

d. Number Replacement: Replace numbers with random alternatives if present.

e. Syntax-aware Degradation: Based on syntax tree analysis:

- Swap adjectives/adverbs or replace with [MASK] (requires multiple adjectives/adverbs)
- Insert random adverbial phrases
- Convert between present and past tenses

f. Positive-Negative Mixing: Concatenate the positive and negative samples of this query to create hybrid weak-positives.

3. Negative Sample Generation: Randomly select samples from other instances as negatives.

The composition of the supplier is highly flexible. In our attempts, slightly changing the param-

eters of those methods does not show significant impact on the overall results if the positive/weak-positive/negative similarity gradient is still maintained.

The pseudocode for the HMCL training framework and the pseudocode for the above simple unsupervised sample supplier are presented in Algorithm 1 and Algorithm 2, respectively.

In practice, users can further adapt the supplier to specific tasks: adding new methods into this supplier, even integrating real-time LLM-generated samples during training.

A.3 Defination and examples of weak-positive

The core of the proposed HMCL framework in this paper is the design of a weak-positive anchor to define an "intermediate state" of reduced similarity. In tasks involving such intermediate states (e.g., five-level scoring in similarity evaluation tasks, or in RAG scenarios where retrieved segments can fully answer the question, partially help infer the answer, or be completely irrelevant), this allows the model to effectively learn from examples of similarity, partial similarity, and complete dissimilarity, better capturing the characteristics of the target task in few-shot settings. Additionally, this anchor transforms the contrast between positive and negative samples from a 2-level to a 3-level scale, further differentiating them.

Therefore, in general, a weak-positive refers to content that is not directly related to the core part of the query or the specific details of its main topic. While it does not directly answer the query, it can still provide some inspiration or indirect help. In terms of ranking within the same group of samples, its relevance is significantly weaker than that of positive samples but stronger than all negative samples.

When constructing our custom dataset, we used to ask annotators to label weak-positive samples. The questionnaire provided to them is already included in Appendix A.3.

However, the specific definition of weak-positive may vary depending on the application scenario. Below are a few examples to illustrate this in detail:

(1) Retrieval of relevant knowledge titles for RAG-based generation (MerchantQuestions-S2S):

Query: What is the calculation methodology for total user assets?

Positive: How is the total user assets calculated?

Weak-positive: What does "total user assets" mean?

Here, the positive title is almost a perfect synonym for the query, and the knowledge under this title should directly answer the query. On the other hand, the weak-positive title does not address the user's core concern about how to calculate total user assets. From this perspective, the two are unrelated. However, this weak-positive title means that the knowledge would give a definition of "total user assets". When the RAG generation module receives this knowledge, it might infer the calculation method of total user assets based on its definition and common sense. Therefore, while the weak-positive is not entirely relevant to the query, it has significant potential to contribute to answering the question.

By learning such examples, the model understands that when no perfectly matching positive sample exists, it should select knowledge like the weak-positive example above from a pool of less relevant candidates to increase the likelihood of generating correct results.

(2) Passage retrieval (FiQA2018):

Query: How to deposit a cheque issued to an associate in my business into my business account?

Positive: Have the check reissued to the proper payee.

Weak-positive: I might have an answer! I imagine they're capitalizing on people's laziness. I live in the Bay Area where some people probably don't mind paying $35 to not have to walk 100 feet to the office and drop off a check...(too long, omitted) Just have the associate sign the back and then deposit it. It's called a third party cheque and is perfectly legal...(too long, omitted)

Here, the weak-positive one is a random mix of a positive sample and a negative sample from the dataset, which also aligns with real-world scenarios of "weak-positive". Compared to the positive sample, which directly answers the query, this weak-positive contains some useful information (e.g., having the associate sign the back of the check), but it is buried within a large amount of irrelevant content (e.g., complaints about laziness in the Bay Area). As a result, the content and theme are not clearly focused.

In a retrieval system, it is obviously best to prioritize showing users the direct answer like positive one. However, if no optimal positive sample exists, displaying a partially relevant weak-positive is still acceptable because users can extract the useful information themselves by reading through the content.

Besides, in Table 8, we evaluate a 4-level contrastive setup by adding a weak-negative tier and assessing the effect. Here, a weak-negative is a sample whose relevance to the query is higher than a true negative but lower than a weak-positive. This is the basic definition of weak-negative, and in different tasks, weak-negative may have different specific connotations and manifestations, in order to meet the above conditions.

However, it is very difficult to define the over-fine level in most practical tasks, so we run the study for the 4-level contrastive setup only on STS benchmarks, which provide high-quality expert semantic similarity labels on a 0–5 scale. We operationalize the four levels as follows: 0–1 as negatives, 1–2 as weak-negatives, 2–4 as weak-positives, and 4–5 as positives. Therefore, we find that under the STS labeling standards, the detailed definitions for those 4 levels are as follows: Positives are pairs that differ only in tense or voice; weak-positives exhibit minor changes in modifiers (e.g., adjectives, adverbs, personal pronouns) that leave the core action or event intact; weak-negatives alter the core predicate and/or object enough to substantially change the main meaning while retaining similar sentence structure, background and scene; and negatives are pairs that are essentially unrelated. An example for the weak-negative in Table 8 experiment is as follows:

Query: a man is playing a guitar.

Positive: the man is playing the guitar.

Weak-positive: the girl is playing the guitar.

Weak-negative: A man is playing a trumpet.

Negative: A woman is riding a horse.

Nevertheless, although the STS task provides rigorous labels and this gradient difference of semantic similarity seems to be understandable, Table 8 shows that the effect of the 4-level contrastive learning is still significantly deteriorated. This indicates that the 4-level training framework is very fragile in application. It is not only difficult to synthesize and generate confident training data, but also difficult to achieve ideal results through training.

A.4 Self-made benchmarks

To prevent public evaluation sets, which have been extensively used in recent years for foundation model training, from leading to insignificant benefits of continued task-specific fine-tuning and failing to effectively differentiate our method from other fine-tuning approaches, we collect consultation data from an intelligent assistant to construct new S2S and S2P benchmarks. The data relates to the retrieval and recall process of the merchant consultation and Q&A system. The recalled document snippets are all publicly available on external web pages. The consultation questions have been manually anonymized, and the use of this data has been approved by our security review. The original evaluation set is in Chinese. To better illustrate the composition of the benchmarks, we include two examples from each benchmark in Table A6, presented with English literal translations.

In MerchantQuestions-S2S, all samples consist of single sentences:

The query is a user's question. The positive sample corresponding to the input query is a knowledge base title directly related to the query. The negative sample is an irrelevant knowledge base title. The weak-positive sample is a knowledge base title that is partially related or potentially helpful for addressing the query.

In MerchantQuestions-S2P, each sample comprises:

A query (user's question), A positive sample (a document segment relevant to the user's question), A negative sample (a document segment largely irrelevant or incapable of answering the user's question). As observed, both positive and negative samples are paragraphs segmented from formatted web documents, containing numerous formatting symbols and redundant information. This constitutes the primary challenge in comprehension, encoding, and the key objective of fine-tuning.

The labels for this dataset were manually annotated by our institution's employees (These individuals have a preliminary understanding of the knowledge related to responding to those merchant inquiries). During annotation:

Annotators were asked to answer the question "Please evaluate the relateness degree between the query and items in the recall list" and label each title accordingly For the S2S task, they were presented with a user's question and 10 search-retrieved titles, they have choices including: Relevant (2 points, positive), Not directly relevant but potentially helpful (1 point, weak-positive), Irrelevant (0 points, negative). For S2P, annotators were presented with a user's question and 10 search-retrieved paragraphs. They classified each paragraph as: Containing information that answers the question (positive) or unable to answer the question (negative). We collected these annotated data, randomly sampled instances, and processed them into

Training Method	STS12	STS13	STS14	STS15	STS16	SICK-R	STS-B	Average
10 thousand training sentences								
SimCSE	**63.04**	75.19	64.97	74.63	**75.48**	66.68	68.01	69.71
DiffCSE	61.92	72.90	63.95	76.89	72.98	**70.48**	69.99	69.87
ESimCSE	62.70	72.06	63.39	73.11	72.88	69.21	68.34	68.81
InfoCSE	61.58	72.84	63.72	76.66	72.79	69.66	70.30	69.65
HMCL	62.97	**76.63**	**67.35**	**77.79**	74.15	69.44	**72.20**	**71.51**
30 thousand training sentences								
SimCSE	61.46	73.91	65.89	75.03	**76.93**	67.41	69.00	69.95
DiffCSE	63.55	75.54	66.62	78.00	75.39	69.66	70.57	71.33
ESimCSE	62.87	76.38	66.47	77.86	74.91	69.22	69.47	71.03
InfoCSE	**64.44**	75.65	66.90	78.32	75.80	**70.02**	70.99	71.73
HMCL	63.38	**77.69**	**68.62**	**78.92**	75.19	68.81	**72.89**	**72.21**
50 thousand training sentences								
SimCSE	59.50	71.50	64.71	73.84	75.25	66.42	66.71	68.28
DiffCSE	63.69	75.63	67.06	78.24	75.67	70.38	70.76	71.63
ESimCSE	**66.58**	76.86	69.75	78.10	**76.40**	67.48	71.25	72.35
InfoCSE	64.43	75.85	67.47	78.15	75.36	**71.06**	70.69	71.86
HMCL	62.81	**78.47**	**70.27**	**79.30**	75.21	69.65	**74.37**	**72.87**

Table A7: Details of comparison of the effectiveness of unsupervised training methods for text representation based on BERT$_{base}$.

	Classification 8 Tasks	**Clustering** 8 Tasks	**Pair Classification** 3 Tasks	**Reranking** 2 Tasks	**Retrieval** 10 Tasks	**STS** 9 Tasks	**Summarization** 1 Tasks	**Average** 41 Tasks
Qwen-GTE-7B	**88.52**	58.97	67.89	**68.45**	61.77	82.85	77.42	70.72
Qwen-GTE-7B-HMCL	88.19	**59.00**	**68.04**	68.30	**61.93**	**83.33**	**78.10**	**70.79**

Table A8. Overall performance of Qwen-GTE-7B fine-tuned with HMCL on a small mixed training set (FIQA2018, STS-B, SICK-R) across the MTEB benchmark (https://huggingface.co/spaces/mteb/leaderboard, 41 tasks when selecting English).

a standardized dataset compliant with the MTEB format. The anonymized version of the dataset is available at https://github.com/antgroup/hmcl-merchant-question.

A.5 Details of experiment results

Table 2 presents the detailed results of the unsupervised experiments for each task, which are listed in Table A7. As can be seen, although our HMCL framework is not specifically designed for unsupervised training from scratch (it is more suitable for supervised fine-tuning), it achieves overall optimal performance in few-shot training compared to recent prominent unsupervised frameworks. Additionally, its unsupervised mode is particularly well-suited for the evaluation sets of STS13-15 and STS-B, where it consistently outperforms other methods.

The HMCL framework introduced in this paper is designed for targeted fine-tuning to enhance specific tasks. Therefore, in other experiments, we mainly focus on comparing how different training frameworks improve performance on corresponding tasks after fine-tuning a base model using task-specific training sets. Generally, this approach involves creating one fine-tuned model per training set, targeting one specific evaluation task. However, the generalizability of the fine-tuned model then becomes a concern. To assess this, we combined the FIQA2018, STS-B, and SICK-R training sets mentioned in Table A5 to fine-tune Qwen-GTE-7B and evaluated its overall performance on the MTEB benchmark. This was done to observe the impact of few-shot fine-tuning on the degradation of general capabilities. The results, shown in Table A8, indicate that compared to the original pre-fine-tuned model, there were slight improvements in STS, Summarization, Retrieval, Clustering, and Pair Classification, while Classification and Reranking saw minor decreases. The final Task-level average score rose slightly from 70.72 to 70.79, remaining essentially stable. This demonstrates that under controlled training sample sizes, the HMCL framework enables models to quickly learn specialized tasks without significantly degrading their general capabilities. The fine-tuned

Similar Pair Level	STS12	STS13	STS14	STS15	STS16	STS-B	Average
Positive/Negative	64.08	74.84	68.59	75.84	76.54	70.80	71.78
Positive/Weak-Positive/Negative	62.81	78.47	70.27	79.30	75.21	74.37	73.41
Positive/Weak-Positive/Weak-Negative/Negative	62.67	73.68	67.70	76.67	74.74	73.45	71.48

Table A9: Details of the ablation studies of using 2-level pair samples, 3-level pair samples, and 4-level pair sample in HMCL framework.

model shows enhanced performance on specific tasks while maintaining reasonable performance in general scenarios, with minimal degradation.

We recommend that if our application scenarios can be clearly divided into distinct domains (e.g., mathematics, coding, news, etc., or applications with varying similarity/dissimilarity criteria), it is still advisable to use separate models fine-tuned on specialized datasets for each domain. These models can then be organized using a classification-based architecture or a Mixture-of-Experts (MoE) framework to better address problems across different fields. However, if the primary application goal is singular, we can focus on strengthening that specific direction while relying on the model's retained general capabilities to handle occasional issues from other domains.

The detailed results of the ablation experiments in Table 7 for each task are presented in Table A9. Clearly, under the HMCL training framework, using three contrastive groups not only achieves overall optimal performance but also yields the best results in most tasks. With fewer than three groups, the full potential is not fully realized. Conversely, with more than three groups, the finer segmentation of samples makes it difficult to maintain the hierarchical differences in similarity during sample construction (for example, it cannot be guaranteed that weak-positives are always better than weak negatives, or that weak negatives are always less similar than negatives). This leads to a significant drop in performance. Therefore, in the domain of text contrastive learning, excessively increasing the number of contrastive groups does not meet expectations.

A.6 Other approaches tried

In addition to the standard HMCL framework reported above, we also explore other approaches during the experimental phase. For example, we attempt to incorporate a momentum method similar to ESimCSE and the ANN-style sampling strategy to enhance the richness and effectiveness of negative samples in HMCL. However, the overall performance improvement is not very significant compared to the increased training cost. Therefore, we decide not to include this method in our standard approach. Nevertheless, in scenarios with high requirements, this method can be used in tasks such as retrieval to optimize the final results.

As mentioned in Appendix A.2, using LLMs into the unsupervised sample supplier to complement the training data is feasible. However, due to the substantial increase in training time and cost when incorporating LLMs, we also exclude this LLM-generating module from our standard mode of HMCL. **All experiments reported in this paper (to ensure fairness) do not use this sample augmentation method.**

At last, the language polishing of this article was completed with the help of AI assistants, including Qwen-max, GPT-5, Gemini-2.5 and DeepSeek.

Algorithm 1 Training with HMCL framework

Require: Dataset $\mathcal{D} = \{q_i, p_i, w_i, n_i\}_{i=1}^{N}$; encoder f_θ; batch size B; the HMCL loss $L(\cdot)$

Define a cosine similarity function $s(q, a)$ between representations
for epoch $= 1, \ldots, T$ **do**
  for each batch $\mathcal{B} = \{q^{(b)}\}_{b=1}^{B}$ from $\mathcal{D}$ **do**
    for $b = 1$ to B **do**
      if $q^{(b)}$ has labeled positive p_{label} **then**
        $p^{(b)} \leftarrow p_{\text{label}}$
      else
        $p^{(b)} \leftarrow \textsc{GeneratePositive}(q^{(b)})$ ▷ fallback to unsupervised
      if $q^{(b)}$ has labeled weak positive w_{label} **then**
        $w^{(b)} \leftarrow w_{\text{label}}$
      else
        $w^{(b)} \leftarrow \textsc{GenerateWeakPositive}(q^{(b)}, p^{(b)})$ ▷ fallback to unsupervised
      if $q^{(b)}$ has hard-negative n_{label} **then**
        $n^{(b)} \leftarrow n_{\text{label}}$
      else
        $n^{(b)} \leftarrow$ None ▷ explicitly mark missing
    Compute representations for all items once:
    $\{q\_e^{(b)} \leftarrow f_\theta(q^{(b)}),\ p\_e^{(b)} \leftarrow f_\theta(p^{(b)}),\ w\_e^{(b)} \leftarrow f_\theta(w^{(b)})\}_{b=1}^{B}$
    if any $n^{(b)}$ is not None **then**
      $\{n\_e^{(b)} \leftarrow f_\theta(n^{(b)}) \mid n^{(b)} \neq \text{None}\}$
    $\mathcal{L} \leftarrow 0$
    for $b = 1$ to B **do**
      $s\_p^{(b)} \leftarrow s\big(q\_e^{(b)}, p\_e^{(b)}\big)$
      $s\_w^{(b)} \leftarrow s\big(q\_e^{(b)}, w\_e^{(b)}\big)$
      $S\_n^{(b)} \leftarrow \Big\{s(q\_e^{(b)}, p\_e^{(j)}),\ s(q\_e^{(b)}, w\_e^{(j)}),\ s(q\_e^{(b)}, q\_e^{(j)}),\ s(p\_e^{(b)}, p\_e^{(j)}) \mid j \neq b\Big\}$
      if $q^{(b)}$ has hard-negative **then**
        $S\_n^{(b)} \leftarrow S\_n^{(b)} \cup \big\{s(q\_e^{(b)}, n\_e^{(b)})\big\}$
        ▷ See Equation (14) for full loss definition
      $\mathcal{L}^{(b)} \leftarrow L\big(s\_p^{(b)}, s\_w^{(b)}, S\_n^{(b)}\big)$
      $\mathcal{L} \leftarrow \mathcal{L} + \mathcal{L}^{(b)}$
    $\mathcal{L} \leftarrow \mathcal{L}/B$; update $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$

Algorithm 2 Unsupervised Sample Supplier

function GENERATEPOSITIVE(x)
  $L \leftarrow \text{length}(x)$
  $m \leftarrow \text{sample\_uniform}(\{\text{rep}, \text{ins}, \text{syn}\})$
  $\mathcal{T}_{\text{neutral}} \leftarrow$ {"the", "so", "thus", "too", "there", "a", "an", "...", ","} ▷ predefined neutral tokens
  if $m = \text{rep}$ **then**
    $N \leftarrow \text{rand\_int}(1, \lfloor 0.3L \rfloor)$
    $P \leftarrow \text{sample\_positions}(L, N)$
    for all $i \in \text{sort\_asc}(P)$ **do**
      $\text{duplicate\_inplace}(x, i)$
  else if $m = \text{ins}$ **then**
    $t \leftarrow \text{sample\_uniform}(\mathcal{T}_{\text{neutral}})$
    $i \leftarrow \text{rand\_int}(0, L)$
    $\text{insert}(x, i, t)$
  else
    $T \leftarrow \text{parse\_syntax}(x)$
    if $\text{has\_adverb}(T)$ **then**
      $\text{reposition\_adverb}(x, T)$
    else if $\text{tense}(x) = \text{PresSimple}$ **then**
      $\text{to\_pres\_cont}(x)$
    else if $\text{tense}(x) = \text{PresCont}$ **then**
      $\text{to\_pres\_simple}(x)$
    else if $\text{has\_copula}(T)$ **then**
      $\text{remove\_copula}(x, T)$
    else
      $x \leftarrow$ RANDOMREPETITION(x)
  return x
function GENERATEWEAKPOSITIVEDEGRADE(x)
  $T \leftarrow \text{parse\_syntax}(x)$ ▷ parse syntactic tree
  $\text{ops} \leftarrow \left[\text{INSERTADVERBIALPHRASE}(x),\ \text{SIMPLIFYVERBTENSE}(x)\right]$
  if $\text{count\_adj\_adv}(T) \geq 2$ **then**
    Add SWAPADJADVORMASK(x) to ops
  $N \leftarrow \text{rand\_int}(1, |\text{ops}|)$
  $\text{selected} \leftarrow \text{random\_subset}(\text{ops}, N)$
  for each *op* in selected **do**
    EXECUTE(*op*)
  return x
function GENERATEWEAKPOSITIVEMIXED(p, n)
  $w \leftarrow \text{concat}(p, n)$ ▷ hybrid: correct + incorrect content
  return w
function RANDOMREPETITION(x)
  $L \leftarrow \text{length}(x)$
  if $L < 2$ **then**
    return x
  $N \leftarrow \text{rand\_int}(1, \lfloor 0.3L \rfloor)$
  $P \leftarrow \text{sample\_positions}(L, N)$
  for all $i \in \text{sort\_asc}(P)$ **do**
    $\text{duplicate\_inplace}(x, i)$
  return x

Algorithm 2 Unsupervised Sample Supplier (continued)

function GENERATEWEAKPOSITIVE(x, p)
  $L \leftarrow \text{length}(x)$
  $w \leftarrow x$ ▷ initialize weak positive as input
  $m \leftarrow \text{sample\_uniform}(\{\text{delete}, \text{insert}, \text{number}, \text{degrade}, \text{mix}, \text{shuffle}\})$
  if $m = \text{delete}$ **then**
    if $L > 8$ **then**
      $N \leftarrow \text{rand\_int}(1, 4)$
      $w \leftarrow \text{delete\_random}(w, N)$
    else
      $m \leftarrow \text{shuffle}$
  else if $m = \text{insert}$ **then**
    $N \leftarrow \text{rand\_int}(2, 5)$
    for $t = 1$ to N **do**
      $pos \leftarrow \text{rand\_int}(0, \text{length}(w))$
      $tok \leftarrow \text{sample\_uniform}(\{[\text{MASK}], [\text{UNK}]\})$
      $\text{insert}(w, pos, tok)$
  else if $m = \text{number}$ **then**
    if $\text{has\_number}(x)$ **then**
      $w \leftarrow \text{replace\_numbers}(w)$
    else
      $m \leftarrow \text{shuffle}$
  else if $m = \text{degrade}$ **then**
    $w \leftarrow$ GENERATEWEAKPOSITIVEDEGRADE$(()w)$
  else if $m = \text{mix}$ **then**
    $n^- \leftarrow \text{sample\_uniform}(\mathcal{D} \setminus \{x\})$
    $w \leftarrow$ GENERATEWEAKPOSITIVEMIXED$(()p, n^-)$
  if $m = \text{shuffle}$ **then**
    if $\text{bernoulli}(0.5)$ **then**
      $w \leftarrow \text{shuffle}(w)$
    else
      for $t = 1$ to 7 **do**
        $i \leftarrow \text{rand\_int}(1, \text{length}(w))$
        $j \leftarrow \text{rand\_int}(1, \text{length}(w))$
        $\text{swap}(w, i, j)$
  return w
function SIMPLIFYVERBTENSE(x)
  $T \leftarrow \text{parse\_syntax}(x)$
  if $\text{has\_prog\_form}(T)$ **then** ▷ e.g., is eating → ate
    $\text{to\_past\_simple}(x, T)$
  else if $\text{tense}(x) = \text{PresSimple}$ **then** ▷ eats → ate
    $\text{to\_past\_simple}(x)$
  else if $\text{tense}(x) = \text{PastSimple}$ **then** ▷ ate → eats / was → is
    $\text{to\_pres\_simple}(x)$
  return x

Algorithm 2 Unsupervised Sample Supplier (continued)

function SWAPADJADVORMASK(x)
  $T \leftarrow \text{parse\_syntax}(x)$
  if $\text{count\_adj\_adv}(T) \geq 2$ **then**
    $\text{swap\_random\_adj\_adv}(x, T)$
  else if $\text{has\_adj\_adv}(T) \wedge \text{bernoulli}(0.5)$ **then**
    $\text{replace\_with\_mask}(x, T)$
  return x
function INSERTADVERBIALPHRASE(x)
  $L \leftarrow \text{length}(x)$
  if $L < 2$ **then**
    return x
  $i \leftarrow \text{rand\_int}(0, L)$
  $\mathcal{P} \leftarrow \{\text{on, at, in, to, inside, into, for, outside, which, that, where, from}\}$
  $\mathcal{A} \leftarrow \{\text{an, a, the, some, any}\}$
  $\mathcal{M} \leftarrow \{[\text{MASK}], [\text{UNK}]\}$
  prep $\leftarrow \text{sample\_uniform}(\mathcal{P})$
  art $\leftarrow \text{bernoulli}(0.8)$? $\text{sample\_uniform}(\mathcal{A}) : \epsilon$
  mask_count $\leftarrow \text{rand\_int}(1, 4)$
  phrase $\leftarrow$ [prep]
  if art $\neq \epsilon$ **then**
    Append art to phrase
  for $k = 1$ to mask_count **do**
    Append $\text{sample\_uniform}(\mathcal{M})$ to phrase
  $\text{insert\_at}(x, i, \text{phrase})$
  return x