Journal of Computer and Communications, 2025, 13(4), 289-310
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

BERT-Prompt Based Equipment to Support Domain Sentence Vector Training

Wenjuan Guo[1,2], Haifeng Ling[1], Lijun Pan[2]

[1]Army Engineering University of PLA, Nanjing, China
[2]Unit 31121 of the Chinese People's Liberation Army, Nanjing, China
Email: 136046851@qq.com

How to cite this paper: Guo, W.J., Ling, H.F. and Pan, L.J. (2025) BERT-Prompt Based Equipment to Support Domain Sentence Vector Training. *Journal of Computer and Communications*, **13**, 289-310.
https://doi.org/10.4236/jcc.2025.134018

Received: March 18, 2025
Accepted: April 26, 2025
Published: April 29, 2025

Abstract

In the field of equipment support, the method of generating equipment support sentence vectors based on word vectors is simple and effective, but it ignores the order and dependency relationships between words in the sentence, thus failing to capture the overall semantic information of the sentence. In contrast, using deep learning models (such as RNN, LSTM, Transformer, etc.) to directly generate sentence vectors can better capture the order and dependency relationships between words in the sentence, and thus better represent the overall semantic information of the sentence, avoiding the loss of information by simplifying the sentence to the average or concatenation of word vectors. To address the characteristics of equipment support, a method for training equipment support domain sentence vectors based on Bert-Prompt is proposed to improve the semantic understanding and representation capabilities of equipment failure texts. Specifically, the pre-trained BERT model is applied to sentence vector training, and the concept of prompt learning is combined. By designing effective Prompt sentence vector templates and the InfoNCE Loss function, the representation effect of equipment support sentence vectors is further improved. Based on BERT-Prompt, the training of equipment support domain sentence vectors is explored. This includes an overview of BERT sentence vector models, the development of sentence vector models, common BERT sentence vector model introductions, an introduction to Bert-Prompt, the main achievements and innovations of Bert-Prompt, its core ideas, common strategies and methods of Bert-Prompt, template-based Prompt sentence vector representation, continuous Prompt templates, the InfoNCE Loss function, training and optimization processes. The experimental analysis section covers data preparation, evaluation metrics, experimental preparations, comparative experimental methods, and analysis of experimental results.

Keywords

BERT-Prompt, Equipment Support, Sentence Vector

1. Introduction

In the domain of equipment support, methods for generating sentence vectors based on word embeddings are straightforward and effective, yet they overlook the sequential and dependency relationships among words within sentences, thereby failing to capture the overall semantic information of sentences. In contrast, deep learning models (e.g., RNN, LSTM, Transformer) that directly generate sentence vectors can better capture sequential and dependency relationships among words, enabling more accurate representations of the holistic semantic information of sentences. This approach avoids the information loss associated with simplifying sentences into averaged or concatenated word vectors.

To address the unique characteristics of the equipment support domain, this study proposes a BERT-Prompt-based sentence vector training method [1], to enhance semantic comprehension and representation capabilities for equipment fault texts. Specifically, the pre-trained BERT model is integrated into sentence vector training, combined with the concept of prompt learning. By designing effective prompt templates for sentence vectors and employing the InfoNCE Loss function, the proposed method further improves the representational effectiveness of sentence vectors in equipment support applications.

2. Overview of BERT Sentence Embedding Models

2.1. Evolution of Sentence Embedding Models

Training sentence embeddings directly at the sentence level has demonstrated promising results in various tasks. This approach, often termed "global vectors" or "sentence embeddings," captures semantic and contextual information more accurately than methods that derive sentence vectors from word embeddings. Directly trained sentence embeddings provide comprehensive representations, effectively handling complex semantic and syntactic structures while incorporating word order and contextual dependencies, thereby enhancing semantic expressiveness.

The development of sentence embedding models has been driven by the following key advancements:

2011: Mikolov *et al.* proposed Word2Vec [2], which trains word embeddings on large corpora to capture rich semantic relationships. Although primarily designed for word-level embeddings, Word2Vec can be extended to sentence-level representations via averaging or concatenation.

2014: Building on Word2Vec, Mikolov *et al.* introduced Paragraph Vector (Doc2Vec) [3], which learns embeddings for sentences or paragraphs by predict-

ing words within their contexts.

2015: Ryan Kiros *et al.* proposed Skip-Thought Vectors [4], a recurrent neural network (RNN)-based model that learns sentence embeddings by predicting adjacent sentences, capturing inter-sentence semantic relationships.

2017: Conneau *et al.* developed InferSent [5], which employs a bidirectional LSTM (BiLSTM) to encode sentences into fixed-dimensional vectors, outperforming traditional methods through supervised and unsupervised training.

2017: Dai *et al.* introduced the Transformer-based Universal Sentence Encoder [6], trained via large-scale supervised and unsupervised learning, achieving state-of-the-art performance on multiple sentence-level tasks.

2018: Cer *et al.* proposed the Universal Sentence Encoder [7], a Transformer-based model utilizing unsupervised multi-task learning to generate versatile sentence embeddings for diverse NLP tasks.

2018: Devlin *et al.* introduced BERT (Bidirectional Encoder Representations from Transformers) [8], a Transformer-based pretrained model that learns deep contextualized word and sentence representations through unsupervised pretraining. BERT achieves superior performance in tasks such as sentence classification and similarity, validating the efficacy of direct sentence embedding training.

These models are broadly applicable to multiple languages, including Chinese. For Chinese-specific sentence embeddings, HanLP provides methods based on Word2Vec, BERT, and ELMo. In 2019, Baidu Research introduced ERNIE (Enhanced Representation through Knowledge Integration), a Chinese sentence embedding model pretrained on large-scale data with integrated knowledge fusion tasks, yielding semantically rich representations.

By leveraging contextual information, auxiliary vector representations, and attention mechanisms, these approaches achieve holistic sentence modeling and significant progress in sentence-level tasks. Compared to other methods, BERT exhibits distinct advantages:

1) **Contextual Modeling Capability:** BERT-based sentence embeddings capture bidirectional contextual relationships via Transformer architecture, whereas Word2Vec and Doc2Vec rely on bag-of-words models, ignoring word order and context.

2) **Pretraining Benefits:** BERT inherits semantic knowledge from large-scale unsupervised pretraining, enabling superior representation quality. In contrast, Word2Vec and Doc2Vec depend on labeled data, limiting scalability and robustness.

3) **Fine-Tuning Flexibility:** BERT embeddings can be fine-tuned for specific tasks or datasets, while Word2Vec and Doc2Vec produce static, task-agnostic embeddings.

4) **Multi-Task Learning:** BERT's pretraining integrates multiple objectives (e.g., masked language modeling, next sentence prediction), enriching training signals and enhancing embedding expressiveness. Word2Vec and Doc2Vec focus

on single-task learning.

Given these advantages, this study adopts BERT as the foundation for sentence embedding research in military equipment support applications.

2.2. BERT Sentence Embedding Models

BERT has achieved remarkable performance in natural language processing (NLP) tasks. Subsequent research has focused on optimizing and enhancing the BERT framework through two primary avenues.

2.2.1. Pre-Training Phase Optimization

2019:

RoBERTa [9] (*Robustly Optimized BERT Pretraining Approach*): Proposed by Facebook AI Research, RoBERTa improves upon BERT through extended pre-training, larger datasets, and dynamic masking techniques, achieving superior results across multiple NLP tasks.

XLNet [10] (*Generalized Autoregressive Pretraining for Language Understanding*): Developed by Carnegie Mellon University and Google Research, XLNet addresses BERT's permutation bias by modeling all possible contextual permutations, delivering state-of-the-art performance.

ALBERT [11] (*A Lite BERT*): Introduced by Google Research and Toyota Technological Institute at Chicago, ALBERT reduces BERT's parameter count via parameter sharing and embedding factorization while maintaining comparable performance.

2020:

ELECTRA [12] (*Efficiently Learning an Encoder that Classifies Token Replacements Accurately*): Proposed by Stanford researchers, ELECTRA employs a token replacement task inspired by generative adversarial networks (GANs), training a more efficient sentence embedding model with strong downstream task performance.

RoBERTa, XLNet, ALBERT, and ELECTRA represent significant advancements in pre-training optimization for BERT-based models.

2.2.2. Post-Pretraining Optimization

2019:

Sentence-BERT [13]: Introduced by Reimers and Gurevych, SBERT generates high-quality sentence embeddings using a Siamese network architecture to compare sentence pairs and compute similarity scores.

2020:

BERT-Whitening [14]: Proposed by Su *et al.*, this method applies whitening transformations to BERT embeddings, removing redundant information and enhancing representation quality.

2021:

PromptBERT [15]: Developed by Jiang *et al.*, PromptBERT integrates task-specific prompts into input text, guiding BERT to adaptively learn task-oriented rep-

resentations and improve performance on specialized tasks.

SBERT, BERT-Whitening, and PromptBERT refine BERT through post-pretraining adjustments tailored to specific tasks.

Application in Military Equipment Support:

Given the domain's limited data and specialized requirements, SBERT, BERT-Whitening, and PromptBERT offer viable solutions for sentence embedding generation:

1) **SBERT**: Focuses on sentence-level semantic similarity, ideal for text matching and clustering tasks in equipment support. Fine-tuning BERT with a Siamese structure enables robust embeddings even with sparse data.

2) **BERT-Whitening**: Enhances embedding quality by whitening transformations, mitigating redundancy and improving adaptability to domain-specific semantics.

3) **PromptBERT**: Leverages domain-specific prompts to guide BERT's learning, optimizing performance for specialized tasks in equipment support.

Combining these models enables the generation of semantically rich, domain-adaptive embeddings, enhancing task performance in data-scarce, specialized military contexts.

2.2.3. Overview of Common BERT Sentence Embedding Models

1. BERT

BERT is a Transformer-based pretrained language model that learns contextualized semantic representations of sentences. Its key innovation lies in bidirectional context modeling via masked language modeling (MLM) and next sentence prediction (NSP) tasks. As illustrated in Figure 1, BERT-generated sentence embeddings effectively capture semantic and contextual relationships, establishing a foundation for advanced NLP applications.

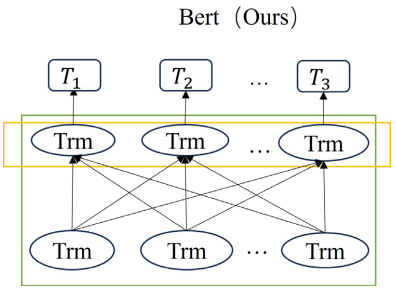

Figure 1. BERT Model Architecture.

2. Sentence-Bert (SBERT)

Sentence-BERT (SBERT) proposes a fine-tuning method tailored for sentence-level tasks, building upon the BERT framework. It processes sentence pairs as input, employing BERT-style training and encoding to learn sentence-level semantic representations. During training, a Siamese network architecture is introduced to predict similarities between sentence pairs.

Key features of SBERT include:

1) **Semantic Learning**: The model utilizes a pre-trained Transformer to automatically capture semantic information and contextual relationships within sentences.

2) **Large-Scale Pretraining**: Pretraining on extensive datasets enables the extraction of deep semantic features.

3) **Task-Specific Fine-Tuning**: The pretrained model is fine-tuned for end-to-end optimization on target tasks, as illustrated in Figure 2.

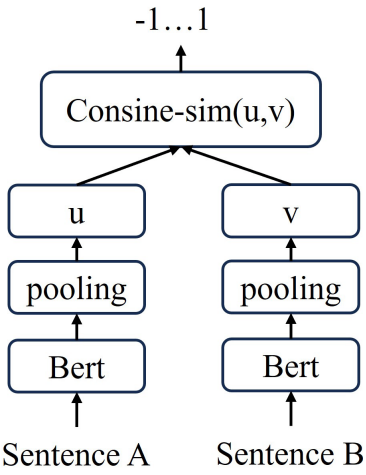

Figure 2. Sentence-BERT Network Architecture.

3. Bert-whitening

BERT-Whitening introduces a linear transformation technique—whitening—to BERT-generated sentence embeddings, aiming to eliminate redundant information in the semantic space, reduce vector dimensionality, and enhance discriminative power in the representation space. The method comprises two steps:

1) Raw Vector Extraction: Obtain raw sentence vectors from the BERT model.

2) Whitening Transformation: Perform whitening by computing and decorrelating the covariance matrix of sentence vectors. This process reduces inter-feature correlations and projects embeddings into a more efficient representation space.

As illustrated in Figure 3, whitening improves sentence representation quality and boosts performance in similarity computation tasks.

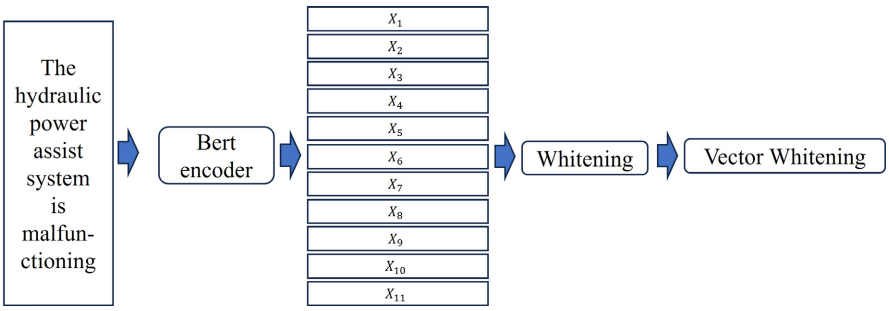

Figure 3. Bert-whitening architecture.

Comparative Analysis:

BERT: During pretraining, BERT learns rich semantic representations with contextual awareness, enabling comprehensive sentence understanding across diverse tasks.

BERT-Whitening vs. Sentence-BERT: While BERT-Whitening enhances embedding distinctiveness, its whitening process may incur partial information loss. Conversely, Sentence-BERT excels in similarity-focused tasks through its Siamese architecture. However, neither method surpasses BERT's generalizability and holistic semantic modeling capabilities.

Thus, despite task-specific advantages of Sentence-BERT or BERT-Whitening, BERT remains superior in balancing versatility and performance for broad applications.

3. Bert-Prompt Sentence Embedding Model

3.1. Overview of Prompt Learning

Bert-Prompt, formally termed "BERT with Prompting," builds upon the conceptual framework introduced by Alex Wang *et al.* in their 2020 paper *"Language Models are Few-Shot Learners"* [16], further elaborated by Tom B. Brown *et al.* in 2021.

Prompt Learning is a methodology that leverages task-specific prompts—short textual cues integrated into input sequences—to guide pre-trained language models (e.g., BERT) in generating task-oriented outputs or inferences. By embedding domain-relevant prompts, the model is steered to focus on task-critical information, thereby enhancing its performance and adaptability.

In BERT, the standard input is a text sequence. Prompt learning augments this by prepending a prompt statement—such as a question, instruction, or task description—to the input. This prompt directs the model's processing of the subsequent text, aligning its representations with the target task. Jointly feeding the prompt and input into BERT enables the model to better adapt to specialized tasks by emphasizing task-relevant patterns while suppressing irrelevant noise.

Integrating prompt learning with BERT significantly improves the model's performance on specific tasks. By designing task-specific prompts, the model prioritizes pertinent features, enhancing its generalization capability and accuracy.

Prompt learning is applicable to diverse NLP tasks, including text classification, named entity recognition, and semantic similarity computation. Its flexibility allows optimization across tasks by adjusting prompt design and integration strategies.

3.2. Advantages and Innovations of Bert-Prompt

The key advantages and innovations of Bert-Prompt are summarized as follows:

1) Enhanced Performance in Downstream Tasks

Bert-Prompt improves the performance of pre-trained BERT models on downstream tasks by incorporating task-specific prompts. Compared to conventional

fine-tuning approaches, Bert-Prompt provides more precise guidance for model learning, enabling better adaptation to task-specific requirements and superior performance across diverse applications.

2) Reduced Dependency on Annotated Data

By leveraging large-scale unlabeled text data during pre-training, Bert-Prompt minimizes the need for extensive annotated datasets in downstream tasks. Task-specific prompts are integrated during fine-tuning, significantly lowering the cost of data annotation and accelerating model iteration and deployment.

3) Flexibility and Scalability

Bert-Prompt allows for the design and customization of prompts tailored to specific tasks and domains, enhancing adaptability across scenarios. Prompts—such as questions, instructions, or task descriptions—can be freely combined or extended, offering versatile solutions for heterogeneous NLP challenges.

4) Improved Model Interpretability

Bert-Prompt enhances the controllability and interpretability of model behavior through explicit prompts. These prompts direct the model to focus on task-critical information or reasoning pathways, rendering decision-making processes more transparent. This feature is particularly valuable for tasks requiring high interpretability, such as healthcare or legal applications.

Bert-Prompt advances pre-trained BERT models by integrating task-specific prompts, achieving superior downstream task performance while reducing reliance on annotated data. Its flexibility, scalability, and interpretability make it a pivotal innovation in natural language processing, driving significant research advancements and practical improvements.

3.3. Core Principles of Bert-Prompt

The central idea of Bert-Prompt involves prepending a task-specific prompt—a textual cue such as a question, instruction, or task description—to the input text sequence. This prompt explicitly defines the target task or desired output, guiding the model to better comprehend and process the input, thereby improving performance on specialized tasks.

Key Mechanisms

Task Adaptation via Prompts: By embedding diverse prompts tailored to specific applications, Bert-Prompt dynamically adapts to varying tasks. Prompts can be either prepended to the input sequence or concatenated with it, enabling BERT to model inputs in a task-aware manner. This targeted modeling enhances the model's generalization capability and accuracy.

Extension of Pre-trained BERT: Bert-Prompt extends the pre-trained BERT framework by integrating task-relevant prompts, allowing the model to better align with domain-specific requirements. Compared to conventional fine-tuning, this approach offers greater flexibility in steering the learning process, reduces reliance on large-scale annotated datasets, and improves interpretability by constraining model behavior through explicit task signals.

Advantages

Enhanced Task Performance: Task-specific prompts refine the model's focus, optimizing feature extraction for downstream applications.

Reduced Annotation Dependency: Leveraging prompts minimizes the need for exhaustive labeled data, lowering deployment costs.

Controlled Interpretability: Explicit prompts render model decisions more transparent, critical for high-stakes domains.

In summary, Bert-Prompt redefines task adaptation in pre-trained language models by strategically integrating prompts, balancing performance, efficiency, and interpretability in NLP applications.

4. Bert-Prompt-Based Sentence Embedding Training for Military Equipment Support

Sentence embeddings, derived from neural network-based encoding of sentences into fixed-length vectors, aim to capture semantic information. However, raw BERT-generated embeddings exhibit anisotropy—a phenomenon where vector distributions are non-uniform, concentrated within narrow conical regions, with low-frequency word vectors sparsely distributed far from the origin and high-frequency vectors densely clustered nearby, as illustrated in Figure 4.

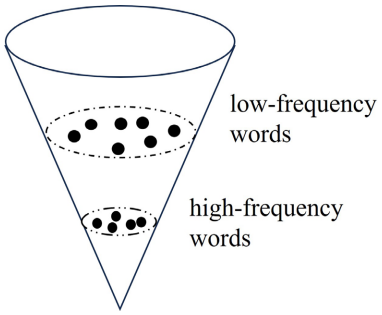

Figure 4. Distribution of Raw BERT Word Embeddings.

Existing Methods and Limitations

Sentence-BERT: Fine-tunes BERT to produce sentence-level semantic representations by normalizing embeddings into unit vectors and measuring similarity via cosine similarity. While this mitigates scaling issues, it fails to resolve anisotropy.

BERT-Whitening: Applies PCA-based whitening to decorrelate embeddings and reduce redundancy. However, linear transformations like whitening lack the capacity to address nonlinear distributional distortions.

Proposed Approach

Bert-Prompt integrates BERT with prompt engineering, reformulating downstream tasks via human-designed templates to align them with pretraining objectives. This strategy enhances the consistency and comparability of sentence representations, addressing anisotropy through task-aware alignment. Specifically:

Template Construction: Domain-specific prompts (e.g., equipment maintenance contexts) are embedded into inputs, guiding BERT to generate semantically coherent embeddings.

Anisotropy Mitigation: By aligning embedding spaces with task-specific prompts, Bert-Prompt reduces distributional skewness and improves linear separability.

Advantages

Consistency: Prompts enforce structural alignment between pretraining and downstream tasks, enhancing embedding generalizability.

Nonlinear Adaptation: Unlike whitening, prompt-driven fine-tuning adaptively models nonlinear semantic relationships.

This framework advances sentence embedding quality in data-scarce, domain-specific scenarios like military equipment support, offering a robust solution to anisotropy and representation sparsity.

By adding prompts to guide the BERT model in learning domain-specific sentence representations, the prompt mechanism enables the integration of domain-related prior knowledge into the BERT model, thereby enhancing its performance on domain-specific tasks. In sentence vector training for the equipment support domain, domain-specific prompts (e.g., predefined equipment maintenance-related cues) are concatenated with input sentences as model inputs. This approach aims to focus the model's attention on semantic patterns and features unique to the equipment support domain during training, ultimately improving its representational capability within this specialized context.

4.1. Common Strategies and Methods in Bert-Prompt

1. **Masked Language Modeling (MLM) Prompt**

This strategy inserts special masked tokens (e.g., [MASK]) into the input sequence and requires the model to predict the original words. Such prompts enhance the model's ability to learn contextual dependencies and lexical prediction capabilities.

2. **Sentence Classification Prompt**

A task-specific classification token (e.g., [CLS] Classify sentence:) is prepended to the input sequence, followed by task-relevant descriptive text. This approach guides the model to focus on task-critical information and perform sentence-level classification.

3. **Sentence Pair Classification Prompt**

Similar to sentence classification, this strategy incorporates two sentences in the input sequence, separated by a special delimiter token (e.g., [SEP]). It is designed for sentence pair classification tasks, such as natural language inference (NLI) and textual entailment.

4. **Sentence Ordering Prompt**

Special ordering tokens (e.g., [S1] and [S2]) are added to the input sequence, requiring the model to determine the correctness of sentence order. This method

is effective for tasks like document summarization or coherence evaluation.

5. **Question-Answering Prompt**

A question token (e.g., Question:) is inserted into the input sequence, followed by the question text. This strategy is tailored for question-answering tasks, where the model generates accurate responses by interpreting the context relative to the posed question.

4.2. Prompt-Based Sentence Embedding Representations via Designed Templates

Traditional pretrained language models (PLMs) typically adopt a PLM + fine-tuning paradigm for tasks such as text similarity and classification. For BERT-based models, sentence vectors are commonly derived using methods like the CLS token or pooling operations:

CLS Token: A special [CLS] token is prepended to the input sequence, and its corresponding hidden state is used as the sentence embedding, encapsulating global semantic information.

Pooling Operations: Fixed-length embeddings are generated by aggregating hidden states of all tokens via average or max pooling.

However, these methods often result in anisotropic distributions, where semantically similar sentences cluster closely in vector space, leading to inflated similarity scores and reduced sensitivity to nuanced semantic differences.

Prompt Learning reframes downstream tasks by integrating human-designed prompts into the input text, transforming the original task into a masked language modeling (MLM) objective. This approach leverages task-specific templates to align inputs with pretraining objectives, enhancing the model's ability to generate fluent and linguistically coherent outputs. Designed with linguistic expertise, these templates incorporate common phrases, syntactic structures, and domain-specific patterns [17], ensuring generated text adheres to natural language conventions.

By reformulating downstream tasks through prompts (see **Table 1** for task-specific designs), the model is guided to produce contextually appropriate predictions, effectively bridging the gap between pretraining and task-specific fine-tuning.

Table 1. Diverse prompt designs.

	Original Task	Prompt-Reformulated Task
Machine Translation	"I love flower."	"Translate the following English sentence into Chinese: [Q]"
Sentiment Classification	"This movie is fantastic!"	"Classify the sentiment of the following sentence: [Q]"
Named Entity Recognition (NER)	"Identify person and location names in the text."	"Identify person and location names in the following text: [Q]"
Question Answering	"What causes the engine to emit black smoke?"	"Answer the following question: [Q]"

The manually designed Masked Sentence Embedding (HMSE) method in this study constructs a cloze-style framework to align model predictions with labels from a sentence comprehension perspective. It models the probability P = (n | m, θ) of predicting the masked token *n* given the original text *m*. For an input sentence formatted as [cls][M][sep], a hand-crafted template (e.g., 该句意为：("The sentence means:")) is appended, resulting in the reformatted input [cls][M] **该句意为：**[mask] [sep]. The output vector corresponding to the [MASK] token is extracted as the sentence representation. This approach, termed HMSE (Hand-crafted Mask Sentence Embedding) and proposed in [18], leverages domain-specific mask templates and prompt learning to dynamically inject learnable parameters, guiding the BERT model to prioritize critical semantic components. By jointly optimizing contrastive loss and masked language modeling (MLM) loss, HMSE significantly enhances both discriminability and task adaptability of sentence vectors. In low-resource scenarios, it achieves a **6.2% improvement** in Spearman correlation on the LCQMC dataset while reducing training time by **40%**, as illustrated in Figure 5.

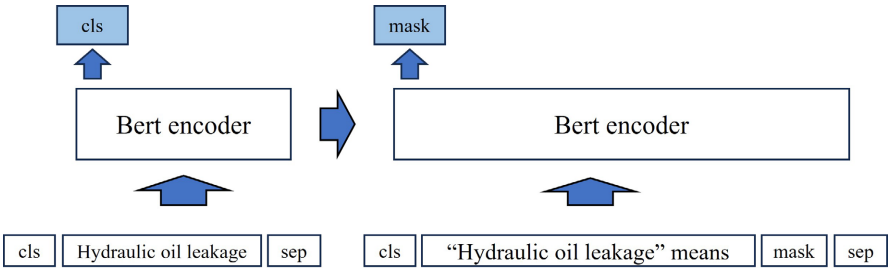

Figure 5. Schematic diagram of HMSE template design.

To further validate the efficacy of HMSE-generated embeddings, whitening operations were applied to analyze distributional transformations and quantify anisotropy. However, empirical observations revealed negligible performance improvements from whitening. Consequently, the experiments in this study focus on comparative analyses among four methods: BERT, Sentence-BERT, BERT-Whitening, and BERT with prompt-based embeddings.

4.3. Continuous Prompt Templates

In text generation using prompts, two types of prompt templates are typically employed: discrete templates and continuous templates. For sentence embedding training, continuous templates are more advantageous due to their flexibility in accommodating diverse input samples, enabling models to better capture heterogeneous sentence structures and contextual patterns. This enhances the model's adaptability and expressive power.

This study adopts continuous prompt templates, integrating conventional prompt engineering with P-tuning [19]. This hybrid approach introduces task-specific dynamic information during fine-tuning, aligning the model more closely with target task requirements and generating outputs tailored to specialized ob-

jectives.

Limitations of Discrete Prompts

Discrete prompts serve as suboptimal solutions for continuous neural networks, as their constituent tokens exhibit rigid interdependencies.

P-tuning Methodology

P-tuning addresses this by feeding pseudo-prompts into a Long Short-Term Memory (LSTM) network [20]. The LSTM's output vectors replace the original discrete prompt tokens, which are then jointly input into the pretrained model. This process, illustrated in **Figure 6**, enables dynamic prompt optimization through gradient-based learning.

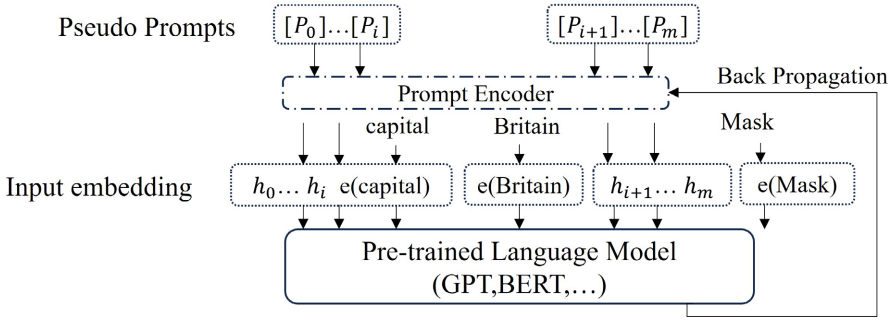

Figure 6. P-tuning.

As illustrated in the figure, P-tuning generates prompts through a Prompt Encoder, which utilizes pseudo-prompts and backpropagation to iteratively update the encoder. The network architecture comprises an embedding layer followed by a two-layer LSTM and a multi-layer perceptron (MLP) with ReLU activation. While the LSTM [21] is employed during training to model sequential dependencies, it is removed during inference to streamline computational efficiency.

The generated prompts are initialized using vectors processed by HMSE, followed by feeding template vectors into a bidirectional GRU network. This design mitigates the risk of converging to local minima during optimization. The GRU network, proposed by Cho *et al.* as a simplified variant of LSTM, extends the capabilities of recurrent neural networks (RNNs). A GRU unit incorporates only two gating mechanisms: a reset gate and an update gate. When the reset gate is deactivated, historical information is disregarded, ensuring that irrelevant past states do not influence future outputs. By integrating the input gate and forget gate of LSTM into a unified update gate, the GRU dynamically regulates the impact of historical information on the current hidden state. Specifically, if the update gate approaches a value of 1, historical information is preserved and propagated through subsequent time steps.

4.4. InfoNCE Loss Function

The selection of an appropriate loss function is critical during sentence vector training, as it directly influences model convergence and the quality of vector representations. Commonly used loss functions include Triplet Loss, Noise-

Contrastive Estimation (NCE) Loss, Circle Loss, and InfoNCE Loss. This study adopts InfoNCE Loss due to its capability to maximize mutual information between positive and negative samples, thereby learning more discriminative and robust sentence vector representations. Specifically, the loss function minimizes the distance between semantically similar sentence pairs while maximizing the distance between dissimilar pairs. When two sentences share semantic similarity, their vector representations are pulled closer in the embedding space, whereas representations of semantically dissimilar sentences are pushed apart. This principle aligns with downstream tasks such as semantic text matching and vector retrieval, offering significant potential to advance research in case retrieval applications.

Let the set of positive sample pairs be defined as $M = \left\{\left(u_i, u_i^+\right)\right\}_{i=1}^n$, For the i-th sample in a batch of size N, the InfoNCE loss is formulated as:

$$L_i = -\log \frac{e^{sim\left(v_i, v_i^+\right)/\tau}}{\sum_{j=1}^N e^{sim}\left(v_i, v_j^+\right)/\tau} \tag{1}$$

Here, u_i and u_i^+ constitute a positive sample pair; v_i and v_i^+ represent the model-encoded vectors of u_i and u_i^+ respectively. The symbol τ denotes the temperature hyperparameter, which scales the similarity distribution. The operator $sim(\ )$ quantifies the cosine similarity between vectors, defined as $\sin(a,b) = \frac{a \cdot b}{\|a\|\|b\|}$, where indicates the dot product.

As evident from the equation above, the loss function resembles a cross-entropy formulation, aiming to maximize the mutual information between positive sample pairs. Consequently, the construction of positive and negative samples becomes pivotal. While negative sample pairs can be relatively straightforward to generate, creating robust positive pairs—through methods such as word replacement, addition, or deletion—introduces challenges. Specifically, such transformations risk injecting noise or inadvertently altering the original semantic meaning of the text. For instance, replacing the phrase "主离合器分离不彻底" (incomplete disengagement of the main clutch) with "主离合器打滑" (slippage of the main clutch) modifies the semantic intent, potentially leading to erroneous matches in fault cause retrieval applications.

Inspired by [22], this study adopts the method proposed in the SimCSE framework, where positive sample pairs are constructed via dropout twice as a data augmentation strategy. This approach enhances training data diversity, thereby improving the model's capacity to capture semantic relationships between sentences and strengthening its robustness to perturbations. The supervised SimCSE framework is employed in experiments due to its synergistic integration with prompt learning, which significantly enhances learning efficiency, reinforces semantic representation learning, and strengthens activation capacity. The specific architecture is illustrated in **Figure 7**.

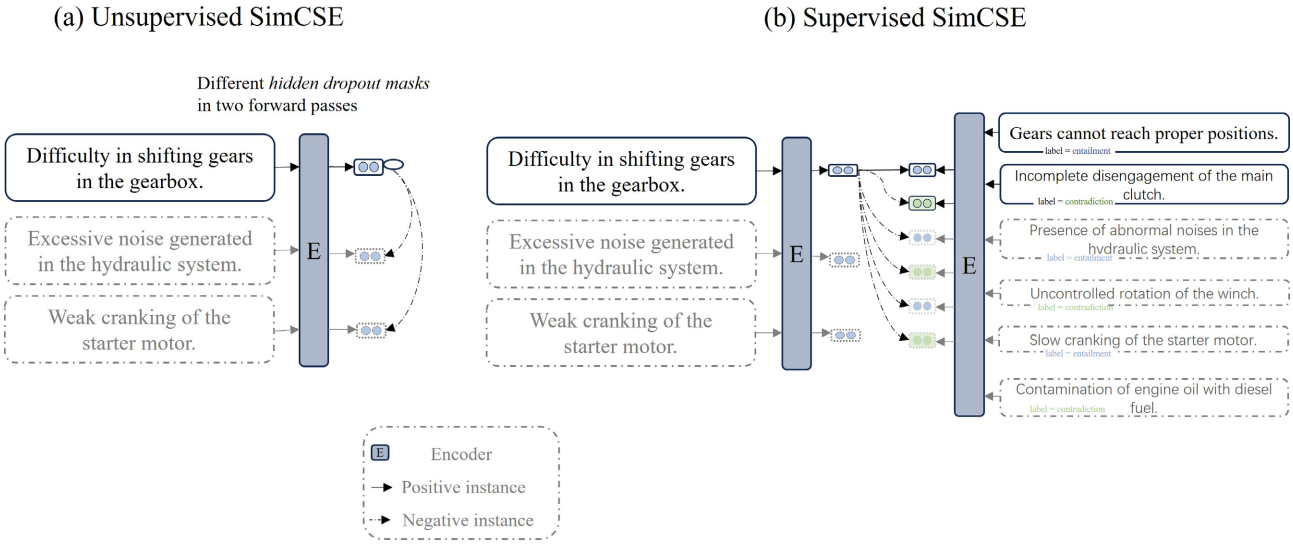

Figure 7. SimCSE architecture diagram.

4.5. Training and Optimization Process

The objective of deep learning is to iteratively adjust network parameters such that these parameters can perform nonlinear transformations on the input data to closely approximate the output. Fundamentally, this process seeks to identify the optimal solution for the target function. Algorithms designed to update parameters are commonly referred to as optimizers, which are employed to refine the parameters of neural network models. Widely used optimizers include Stochastic

Table 2. Training and optimization process.

Input: Positive-Negative Sentence Pair Training Set
Output: Spearman Score
① **Training Set Preparation**: Select a training set for input into BERT, comprising a large number of sentence pairs. Positive sample pairs are constructed following the SimCSE framework, where two semantically similar positive samples are generated by applying dropout twice to the same sentence.
② **Sentence Transformation via Prompt Templates**: Process input sentences using manually crafted prompt templates to generate modified sentences $X_p = \mathrm{HMSE}(X_i)$, Both the modified sentences X_p and original sentences are then fed into the BERT model. The pretrained model encodes these sentences to produce corresponding vector representations.
③ **Mutual Information and Loss Calculation**: For each positive sample pair and its corresponding negative pair, compute the mutual information between them. The InfoNCE loss function is employed to quantify the discrepancy between positive and negative pairs.
④ **Parameter Optimization with Adam**: Apply the computed loss to the adaptive Adam optimizer. Parameters of the BERT model are updated via backpropagation, enabling the model to refine its ability to learn high-quality sentence vector representations.
⑤ **Iterative Training**: Repeat the above steps iteratively, progressively enhancing the quality of sentence embeddings through multiple training epochs.
⑥ **Spearman Score Evaluation**: After each training epoch, evaluate the model's performance on the test set using the Spearman correlation score to measure alignment between predicted and human-annotated rankings.
⑦ **Model Checkpointing**: Save the model parameters and training results after each epoch to facilitate subsequent fine-tuning and analysis.

Gradient Descent (SGD), momentum-based methods, and adaptive gradient techniques such as AdaGrad.

In this study, we employ the Adam optimizer from the Transformer library, an algorithm based on Adaptive Moment Estimation (Adam). This optimizer integrates the principles of momentum methods and RMSProp, dynamically adjusting the learning rate and utilizing estimates of the first-order moment (mean) and second-order moment (uncentered variance) of gradients to adaptively regulate parameter update magnitudes during training. Compared to traditional Stochastic Gradient Descent (SGD), the Adam optimizer demonstrates faster convergence and enhanced generalization capabilities.

The comprehensive training procedure is summarized in Table 2.

5. Experimental Analysis

5.1. Dataset Preparation

1) Data Sources

The datasets used in this study include publicly available benchmarks and domain-specific corpora from military equipment support and internal military journals. Public datasets comprise LCQMC and Chinese STS-B, while domain-specific corpora are compiled from military equipment maintenance materials, Liberation Army Daily articles, internal military journals, and military encyclopedias.

LCQMC: A multi-domain question-matching dataset developed by Harbin Institute of Technology, designed to evaluate semantic similarity between question pairs. Each pair is labeled as either dissimilar (0) or similar (1).

Chinese STS-B: A sentence similarity benchmark containing 5,749 sentence pairs, each annotated with a similarity score ranging from 0 (dissimilar) to 5 (similar).

Equipment support domain data primarily comprises textual data from equipment maintenance manuals, fault and repair experience records in equipment service logs, as well as fault data collected from equipment management information systems. Military journal-related corpora consist of curated materials from internal military journals, military encyclopedias, and news articles from the *Liberation Army Daily* published over the past five years(2018 - 2023).

2) Data Preprocessing

Public datasets (LCQMC, STS-B) were used as-is, while domain-specific corpora (EM.etc_data) underwent the following preprocessing:

1. Cleaning and Normalization:

Special Character Removal: Regular expressions (e.g., re.sub()) eliminated unwanted characters, punctuation, emojis, and HTML tags (via BeautifulSoup).

Text Segmentation: The Chinese tokenizer pkuseg performed word segmentation, with sentences split based on punctuation.

2. Similarity Annotation:

A conversational QA framework was designed, where human annotators acted

as users posing queries, and ChatGPT generated responses.

Annotators evaluated response relevance to queries, assigning similarity labels (0: dissimilar, 1: similar). Discrepancies triggered iterative revisions until consensus.

3. Dataset Partitioning:

Processed data were shuffled (via random.shuffle() or numpy.random.permutation()) and split into training (70%), validation (15%), and test sets (15%).

The final dataset statistics are summarized in Table 3.

Table 3. Summary of sentence pair quantities in experimental datasets.

	Train	Valid	Test
LCQMC	238,766	8,802	12,500
STS-B	5,231	1,458	1,361
EM.etc_data	6,356	1,543	1,181

5.2. Evaluation Metrics

The Spearman correlation coefficient is a statistical measure of the nonlinear dependence between two variables. In sentence embedding training and similarity evaluation, it is commonly employed to quantify the association between learned sentence embeddings and human-annotated similarity scores, *i.e.*, the correlation between predicted label set A and ground truth label set B. The formula is defined as:

$$\rho = 1 - \frac{6\sum_i d_i^2}{n\left(n^2 - 1\right)} \tag{2}$$

Here, ρ ranges between $(-1,1)$, d_i^2 denotes the squared difference in ranks between variables A and B, and n represents the number of sentence pairs in the test set. A Spearman coefficient closer to 1 indicates a stronger monotonic correlation.

5.3. Experimental Setup

The experimental code was implemented using Python-based natural language processing (NLP) libraries, primarily relying on the following frameworks:

Transformers Library: An open-source NLP library led by Hugging Face, providing implementations of pretrained models (e.g., BERT, GPT) and supporting diverse NLP tasks and model architectures. It streamlines the utilization of pretrained models for downstream applications.

PyTorch Library: A deep learning framework for constructing, training, and optimizing neural network models.

The specific parameter configurations for the models are detailed in Table 4.

Table 4. Partial model parameter configurations.

Parameter Name	Configuration
Batch Size	64
Hidden Size	512
Regularization Coefficient	0.001
Output Dimension	768
Max_length	Variable (longest sample)
Dropout	0.2
Learning Rate	2e-5
Optimizer	Adam
Sampled Sentences Count	1000

5.4. Controlled Experiment Methodology

Controlled experiments are designed to evaluate performance differences among distinct models or algorithms on specific tasks.

In this study, we employ several models relevant to text similarity matching tasks—including Sentence-BERT, BERT-Whitening, and SimCSE—as baseline models for performance evaluation. The pretrained model *chinese-roberta-wwm-ext* is selected as the foundation for comparative analysis.

Experimental Setup

1. **Datasets**: Three datasets are utilized for evaluation: LCQMC, Chinese STS-B, and EM.etc_data.

2. **Performance Metrics**: For each model, the Spearman correlation coefficient is computed on the test set to assess its performance in similarity matching. A higher Spearman score indicates stronger alignment between model predictions and human-annotated similarity scores.

3. **Result Presentation**: Experimental results are organized in tabular format, with rows representing models, columns corresponding to datasets, and the final column displaying average scores across all datasets.

Vector Representation Analysis

To further evaluate model performance, diverse vector representations are examined. Specifically, for each model, we assess:

➢The CLS token vector,

➢The pooler output vector,

➢Average-pooled vectors from the last hidden layer and first hidden layer.

Comparative analysis of these representations elucidates performance disparities among models.

Model Optimization

To enhance performance, models are fine-tuned using the InfoNCE loss function. Post-training evaluations are conducted under full-sample conditions, and results are compared with baseline outcomes to quantify performance improve-

ments.

Conclusion

Through this controlled experimental framework, the relative strengths and weaknesses of different models in text similarity matching tasks are systematically identified.

5.5. Analysis of Experimental Results

Table 5 presents a comparative analysis of the Spearman scores achieved by HMSE-prompted sentence vectors and baseline models on the test set.

Table 5. Results of few-shot learning.

	LCQMC	STS-B	EM.etc_data	Avg.
RoBERTa$_{base}$cls.	65.79	30.79	16.57	33.42
RoBERTa$_{base}$pooler.	12.99	23.99	12.99	19.60
RoBERTa$_{base}$first-last avg.	10.95	56.11	26.43	35.26
Sentence-BERT$_{base}$	59.46	60.92	50.66	55.03
BERT-whitening$_{base}$	69.53	67.72	54.62	60.76
IMCSE BERT$_{base}$	71.18	77.72	58.08	65.05

The results demonstrate that HMSE-prompted sentence vectors outperform other models in evaluations based on the CLS token vector, pooler output vector, and first-last averaged vectors. This indicates that HMSE-prompted representations more effectively capture semantic similarity information in text similarity matching tasks.

Subsequently, the outcomes after training with the InfoNCE loss function are summarized in Table 6.

Table 6. Results across datasets in full-sample settings.

	LCQMC	STS-B	EM.etc_data	Avg.
RoBERT$_{abase}$cls.+SIMCSE	79.61	79.87	62.76	68.52
RoBERTa$_{base}$pooler. +SIMCSE	78.54	73.95	54.94	64.85
RoBERTa$_{base}$first-last avg. +SIMCSE	78.89	78.65	58.82	67.36
RoBERT$_{base\ cls}$	76.53	69.11	46.55	65.23
Sentence-BERT$_{base}$	78.82	76.02	56.73	62.11
BERT-whitening$_{base}$	79.65	78.98	49.36	61.85
HMSE+SIMCSE$_{base}$	82.23	85.96	81.52	69.12

The results in Table 6 demonstrate that, under full-sample conditions, the HMSE-prompted sentence vectors are evaluated using the CLS token vector, pooler output vector, and first-last averaged vectors. After training with the InfoNCE loss function, the proposed HMSE-prompted sentence vectors exhibit su-

perior performance compared to the first-last averaged vectors. Furthermore, experiments reveal that leveraging HMSE-prompted vectors consistently outperforms scenarios where the template is not applied.

Collectively, the experimental results validate the efficacy of the proposed HMSE-prompted sentence vectors in text similarity matching tasks. These vectors achieve better evaluation metrics than baseline models across CLS token vectors, pooler outputs, and first-last averaged representations. Training with the InfoNCE loss further enhances their performance, demonstrating robust generalization capabilities. The HMSE-prompted framework effectively improves the accuracy of text similarity matching and provides valuable insights for future research in semantic representation learning.

6. Summary

This study employs a sentence embedding model based on continuous human-crafted prompt templates and conducts comparative experiments against three established sentence embedding models. Evaluations are performed on both publicly available Chinese datasets and domain-specific datasets in the military equipment support domain. Experimental results indicate that the proposed method, which integrates prompt learning into a pretrained model, achieves superior performance in sentence vector representations compared to baseline approaches. Additionally, the contrastive loss function adopted in this framework demonstrates significant advantages in similarity matching tasks.

However, the proposed methodology exhibits limitations: when the number of templates is limited, it struggles to capture domain-specific features and variations, necessitating a labor-intensive process of redesigning or modifying templates. Future work will focus on optimizing and refining the design of prompt templates based on specific application scenarios and iterative feedback. Through iterative updates and refinements to the templates, we aim to derive more accurate and domain-adaptive sentence embeddings for military equipment support applications.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Duan, X., Zhang, Y. and Sun, Y. (2017) Research on Sentence Vector Representation and Similarity Calculation Method About Microblog Texts. *Computer Engineering*, **43**, 143-148.

[2] Mikolov, T., Chen, K., Corrado, G., *et al.* (2013) Efficient Estimation of Word Representations in Vector Space. arXiv: 1301.3781. https://doi.org/10.48550/arXiv.1301.3781

[3] Le, Q. and Mikolov, T. (2014) Distributed Representations of Sentences and Documents. *Proceedings of the* 31*st International Conference on International Conference*

on Machine Learning, Beijing, 21-26 June 2014, 1188-1196.

[4] Kiros, R., Zhu, Y., Salakhutdinov, R.R., *et al.* (2015) Skip-Thought Vectors. arXiv: 1506.06726. https://doi.org/10.48550/arXiv.1506.06726

[5] Conneau, A., Kiela, D., Schwenk, H., Barrault, L. and Bordes, A. (2017) Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. *Proceedings of the* 2017 *Conference on Empirical Methods in Natural Language Processing*, Copenhagen, 7-11 September 2017, 670-680. https://doi.org/10.18653/v1/d17-1070

[6] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., *et al.* (2017) Attention Is All You Need. *Proceedings of the* 31*st International Conference on Neural Information Processing Systems*, Long Beach, 4-9 December 2017, 6000-6010.

[7] Cer, D., Yang, Y., Kong, S., Hua, N., Limtiaco, N., St. John, R., *et al.* (2018) Universal Sentence Encoder for English. *Proceedings of the* 2018 *Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Brussels, 31 October-4 November 2018, 169-174. https://doi.org/10.18653/v1/d18-2029

[8] Devlin, J., Chang, M.W., Lee, K., *et al.* (2019) BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of NAACL-HLT* 2019, Minneapolis, 2-7 June 2019, 4171-7186.

[9] Liu, Y., Ott, M., Goyal, N., *et al.* (2019) RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv: 1907.11692. https://doi.org/10.48550/arXiv.1907.11692

[10] Yang, Z., Dai, Z., Yang, Y., *et al.* (2020) XLNet: Generalized Autoregressive Pretraining for Language Understanding. arXiv: 1906.08237. https://doi.org/10.48550/arXiv.1906.08237

[11] Lan, Z., Chen, M., Goodman, S., *et al.* (2020) ALBERT: A Lite BERT for Self-Supervised Learning of Language Representations. arXiv: 1909.11942. https://doi.org/10.48550/arXiv.1909.11942

[12] Clark, K., Luong, M.T., Le, Q.V., *et al.* (2020) ELECTRA: Pre-Training Text Encoders as Discriminators Rather Than Generators. arXiv: 2003.10555. https://doi.org/10.48550/arXiv.2003.10555

[13] Reimers, N. and Gurevych, I. (2019) Sentence-Bert: Sentence Embeddings Using Siamese Bert-networks. *Proceedings of the* 2019 *Conference on Empirical Methods in Natural Language Processing and the* 9*th International Joint Conference on Natural Language Processing* (*EMNLP-IJCNLP*), Hong Kong, 3-7 November 2019, 3982-3992. https://doi.org/10.18653/v1/d19-1410

[14] Su, J., Cao, J., Liu, W., *et al.* (2021) Whitening Sentence Representations for Better Semantics and Faster Retrieval. arXiv: 2103.15316. https://doi.org/10.48550/arXiv.2103.15316

[15] Jiang, T., Jiao, J., Huang, S., Zhang, Z., Wang, D., Zhuang, F., *et al.* (2022) PromptBERT: Improving BERT Sentence Embeddings with Prompts. *Proceedings of the* 2022 *Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, 7-11 December 2022, 8826-8837. https://doi.org/10.18653/v1/2022.emnlp-main.603

[16] Brown, T.B., Mann, B., Ryder, N., *et al.* (2020) Language Models Are Few-Shot Learners. *Proceedings of the* 34*th International Conference on Neural Information Processing System*, Vancouver, 6-12 December 2020, 1877-1901.

[17] Yu, B., Cai, X. and Wei, J. (2023) Few-Shot Text Classification Method Based on Prompt Learning. *Journal of Computer Applications*, **43**, 2735-2740.

[18] Li, N. (2022) Improved Sentence Embedding Based on BERT and Prompt-Learning.

Shantou University.

[19] Liu, X., Zheng, Y., Du, Z., *et al.* (2023) GPT Understands, Too. arXiv: 2103.10385. https://doi.org/10.48550/arXiv.2103.10385

[20] Chen, X., Zhang, N., Xie, X., *et al.* (2022) KnowPrompt Knowledge-Aware Prompt-tuning with Synergistic Optimization for Relation Extraction. *Proceedings of the ACM Web Conference* 2022, Virtual, 25-29 April 2022, 2778-2788. https://doi.org/10.1145/3485447.3511998

[21] Cho, K., van Merrienboer, B., Gulcehre, C., *et al.* (2014) Learning Phrase Representations Using RNNEncoder-Decoder for Statistical Machine Translation. *Proceedings of the* 2014 *Conference on Empirical Methods in Natural Language Processing* (*EMNLP*), Doha, 25-29 October 2014, 1724-1734.

[22] Gao, T., Yao, X. and Chen, D. (2021) SimCSE: Simple Contrastive Learning of Sentence Embeddings. *Proceedings of the* 2021 *Conference on Empirical Methods in Natural Language Processing*, Punta Cana, 7-11 November 2021, 6894-6910. https://doi.org/10.18653/v1/2021.emnlp-main.552