DASA-Trans-STM: Adaptive Efficient Transformer for Short Text Matching using Data Augmentation and Semantic Awareness

Jiguo Liu[1,2]*, **Chao Liu**[1,2], **Meimei Li**[1,2]*, **Nan Li**[1,2], **Shihao Gao**[1,2], **Dali Zhu**[1,2]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
liujiguo@iie.ac.cn

Abstract

Rencent advancements in large language models (LLM) have shown impressive versatility across various tasks. Short text matching is one of the fundamental technologies in natural language processing. In previous studies, the common approach to applying them to Chinese is segmenting each sentence into words, and then taking these words as input. However, existing approaches have three limitations: 1) Some Chinese words are polysemous, and semantic information is not fully utilized. 2) Some models suffer potential issues caused by word segmentation and incorrect recognition of negative words affects the semantic understanding of the whole sentence. 3) Fuzzy negation words in ancient Chinese are difficult to recognize and match. In this work, we propose a novel adaptive Transformer for Chinese short text matching using *Data Augmentation and Semantic Awareness* (DASA), which can fully mine the information expressed in Chinese text to deal with word ambiguity. DASA is based on a Graph Attention Transformer Encoder that takes two word lattice graphs as input and integrates sense information from N-HowNet to moderate word ambiguity. Specially, we use an LLM to generate similar sentences for the optimal text representation. Experimental results show that the augmentation done using DASA can considerably boost the performance of our system and achieve significantly better results than previous state-of-the-art methods on four available datasets, namely MNS, LCQMC, AFQMC, and BQ.

1 Introduction

Short text matching (STM) plays an essential role in semantic similarity recognition. The main task of STM aims to predict whether two sentences are semantically equivalent or not. STM is a fundamental component in many NLP applications including information retrieval (Ensan and Al-Obeidat, 2019; Arabzadeh et al., 2020; Wang et al., 2020), question answering systems (Liu et al., 2018; YueLiu et al., 2019; Wang et al., 2020; Wu et al., 2020) and dialogue systems (Yu et al., 2014; Gao et al., 2018; Feng et al., 2019), etc.

Recent years have seen great progress in deep learning methods for text matching (Mueller and Thyagarajan, 2016; Chen et al., 2016; Gong et al., 2017; Lan and Xu, 2018). However, almost all of these models were initially proposed for English text matching. For Chinese language tasks, earlier approaches either used Chinese characters directly as input or segmented sentences into words before feeding them into a STM model. While character-based models often outperform word-based ones, a key limitation is that they do not fully leverage explicit word-level information, which has been demonstrated to be useful for semantic similarity matching (Li et al., 2019, 2020).

However, a large number of Chinese words are polysemous, which brings great difficulties to semantic understanding. There are more polysemy in short texts than in long texts, as short texts usually have less contextual information, making it difficult for the model to capture the correct meaning. As is shown in Figure 1, the word "露出去 (disclose)" in red in source text actually has two meanings: one is to describe "泄露 (leakage)" and another is "流出 (outflow)". Intuitively, if other words in the context have similar or related meanings, the probability of of their occurrence will increase. In addition, the negative word " 不能 (not be)" in blue in source text needs to be considered in semantic understanding, which plays a strong turning role in the semantics of the whole sentence.

Furthermore, it is inevitable to make word segmentation errors. This will result in semantic ambiguity, inconsistency and changes, and errors in final matching. For example, if the word segmentation fails to output "露出去 (disclose)" in source

*Corresponding author

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 4595–4611

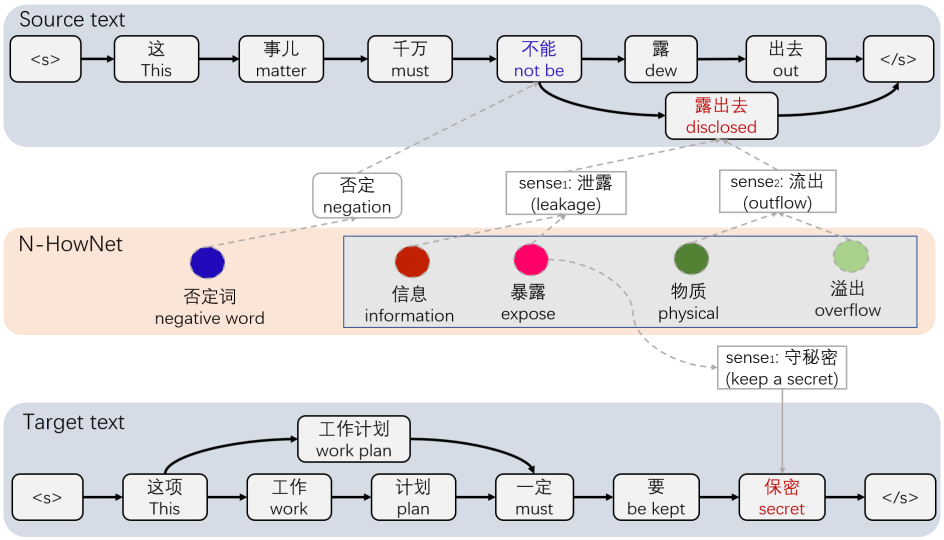

Figure 1: An example of word segmentation and the potential word ambiguity.

text, we will lose useful sense information. In Chinese, "露 (dew)" " 出去 (out)" is a bad segmentation, which deviates the correct meaning of "露出去 (disclose)". It has been shown that correct Chinese word segmentation is important for text matching (Lai et al., 2019).

To address the above issue, we propose a novel **D**ata **A**ugmentation and **S**emantic **A**wareness (DASA) method to consider both data augmentation and semantic information for Chinese short text matching tasks. We first use an LLM to generate similar sentences. Next, we introduce N-HowNet as an external knowledge to integrate semantic information of words in semantic awareness layer. The key insight comes from reductionism in linguistics, where lexicon can be described with the minimum indivisible units of meaning, i.e. semantic units, are defined as sememes (Bloomfield et al., 1926; Dong et al., 2003). Then, we use several segmentation paths to form our lattice graph and construct a set of senses according to the word. We further encode through the sequence Transformer Encoder that takes two word lattice graphs as input and integrates sense information.

Finally, we conducted extensive experiments on three public datasets and unique Military Network Security (MNS) datasets to evaluate the proposed model. We find that our framework is quite effective for various STM, which achieves state-of-the-art (SoTA) performances for widely-used benchmark datasets. In particular, we obtain 89.70%, 86.80%, 85.35%, and 95.65% F1 on LCQMC, AFQMC, BQ, and MNS datasets respectively.

Our contributions of this paper can be summarized as follows:

- We propose a Data Augmentation and Semantic Awareness (DASA) framework for Chinese short text matching, which can effectively eliminate semantic ambiguity in Chinese text by integrating the external knowledge base.
- The experimental results show that our proposed model can considerably boost the performance of our STM system, and achieve significantly better results than previous SoTA methods and variant models.
- We construct a new MNS dataset and also observe that DASA has better generalization on shorter texts. We demonstrate that both data augmentation and semantic information are important for text matching modeling, especially on shorter texts.

2 Related Work

BERT-based Models. BERT-based models have shown its powerful performance on various natural language processing (NLP) tasks including text matching. For Chinese text matching, BERT (Devlin et al., 2018) takes a pair of sentences as input and each Chinese character is a separated input token. Although character-based models can overcome the problem of data sparsity to some degree (Li et al., 2019), a key limitation is

that they do not fully leverage explicit word-level information. To tackle this problem, some variants of original BERT have been proposed. MacBERT (Cui et al., 2021) is proposed to mitigate the gap between the pre-training and fine-tuning stage by masking the word with its similar word, which has proven to be effective on various downstream tasks. ERNIE (Sun et al., 2019) is designed to learn language representation enhanced by knowledge masking strategies, which includes entity-level masking and phrase-level masking. Sentence-BERT (Reimers et al., 2019) utilized siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine-similarity. BERT-flow (Huang et al., 2013) is proposed to transform the anisotropic sentence embedding distribution to a smooth and isotropic Gaussian distribution through normalizing flows that are learned with an unsupervised objective.

Deep Text Matching Models. The natural language processing method based on deep learning has been widely adopted for short text matching. These approaches can be divided into two categories: representation-based models (Lai et al., 2019; Huang et al., 2013; He et al., 2016) and interaction-based models (Chen et al., 2016; Gong et al., 2017; Yin et al., 2016; Wang et al., 2017).

Most representation-based models are based on Siamese architecture, which has two symmetrical networks (e.g. LSTMs and CNNs) to extract high-level features from two sentences. Then, these features are compared to predict text similarity. Huang *et al.* (Huang et al., 2013) proposed a new latent semantic models with a deep structure that project queries and documents into a common low-dimensional space. He *et al.* (He et al., 2016) proposed a novel Text-Attentional Convolutional Neural Network (Text-CNN) that particularly focuses on extracting text-related regions and features from the image components. Mueller *et al.* (Mueller and Thyagarajan, 2016) proposed a Bi-directional Long Short Term Memory (BiLSTM) that is another type of Siamese architecture used for encoding each sentence. Lattice-CNN (Lai et al., 2019) is also proposed to deal with the potential issue of Chinese word segmentation. It takes word lattice as input and pooling mechanisms are utilized to merge the feature vectors produced by multiple CNN kernels over different n-gram contexts of each node in the lattice graph. However, these frameworks ignore the lower-level interactive features between the two indispensable texts.

Interaction-based models make up for this deficiency by using the attention mechanism to obtain the interactive features of words or phrases between two texts, which has been applied to many deep learning tasks and achieved significant performance improvement. Yin *et al.* (Yin et al., 2016) proposed a general attention based convolutional neural network (AB-CNN) for modeling a pair of sentences. Wang *et al.* (Wang et al., 2017) proposed a bilateral multi-perspective matching (BiMPM) model for natural language sentence matching tasks. Chen *et al.* (Chen et al., 2016) proposed an Enhanced Sequential Inference Model (ESIM), which achieves state-of-the-art results on various matching tasks. Lai *et al.* (Lai et al., 2019) proposed a novel lattice based CNN model (LCNs) to utilize multi-granularity information inherent in the word lattice while maintaining strong ability to deal with the introduced noisy information for matching based question answering in Chinese.

Contrastive Learning Models. Many researchers have increased attention on text similarity based on contrastive learning. Gao *et al.* (Gao et al., 2021) utilized the dropout of SimCSE model to generate two different sense embedding as a positive example for comparison, which greatly improves state-of-the-art sentence embeddings on semantic textual similarity tasks. Yan *et al.* (Yan et al., 2021) proposed ConSERT, a contrastive framework for self-supervised sentence representation transfer, that adopts contrastive learning to fine-tune BERT in an unsupervised and effective way. Chuang *et al.* (Chuang et al., 2022) proposed DiffCSE, an unsupervised contrastive learning framework for learning sentence embeddings. Empirical results on semantic textual similarity tasks and transfer tasks both show the effectiveness of DiffCSE compared to current state of-the-art sentence embedding methods. Liu *et al.* (Liu et al., 2023) proposed a short Text Matching model that combines contrastive learning and external knowledge. This model uses a generative model to generate corresponding complement sentences and uses the contrastive learning method to guide the model to obtain more semantically meaningful encoding of the original sentence.

3 Methodology

3.1 Overall Framework

We formulate the short text matching problem in this paper as follows. Formally, we can represent each example of the STM task as a triple (S^a, S^b, y), where $S^a = \{c_1^a, c_2^a, \ldots, c_n^a\}$ is a sentence with a length n, $S^b = \{c_1^b, c_2^b, \ldots, c_m^b\}$ is the second sentence with a length m, c_i^a and c_j^b denote the i-th character and j-th character in the sentences respectively, $y \in \mathcal{Y}$ is the label representing the relationship between S^a and S^b, and $\mathcal{Y}$ is a set of task-specific labels. The STM task can be represented as estimating a conditional probability $\Pr(\mathrm{y}|\mathrm{S^a}, \mathrm{S^b})$ based on the training set, and predicting the relationship for testing examples by $y^* = \arg\max_{y \in \mathcal{Y}} \Pr(\mathrm{y}|\mathrm{S^a}, \mathrm{S^b})$. Concretely, for a paraphrase identification task, S^a and S^b are two sentences, $\mathcal{Y} = (0, 1)$, where $y = 1$ means that S^a and S^b are paraphrase of each other, and $y = 0$ otherwise. The goal of a text matching model $\boldsymbol{\lambda}(S^a, S^b)$ is to predict whether the semantic meaning of S^a and S^b is equal.

Figure 2 shows the overall framework of our proposed DASA model for Chinese short text matching. Given two original Chinese sentences S^a and S^b. First, we use a ChatSG for data augmentation and obtained two new Chinese sentences $\mathcal{C}^a$ and $\mathcal{C}^b$. Then, instead of segmenting each sentence into a word sequence, we use four segmentation tools and keep these segmentation paths to form a word lattice graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. $\mathcal{Q}^+(x_i)$ is the set including the node x_i itself and the nodes which are directly connected by x_i. Each node $x_i \in \mathcal{V}$ corresponds to a word w_i which is a character subsequence starting from the t_1-th character to the t_2-th character in the sentence. For two nodes $x_i \in \mathcal{V}$ and $x_j \in \mathcal{V}$, if x_i is adjacent to x_j in the data augmentation sentence, then there is an edge between them. Thus for each sample, we have two graphs $\mathcal{G}^a = (\mathcal{V}^a, \mathcal{E}^a)$ and $\mathcal{G}^b = (\mathcal{V}^b, \mathcal{E}^b)$, and our graph matching model is to predict their similarity.

3.2 Data Augmentation

Because the number of synonymous sentences is far less than the number of non-synonymous sentences, the available short text matching data is very rare. To solve this problem, we use an LLM to generate similar sentences to expand the dataset and improve the performance of short text matching. LLM fully utilizes multi-granularity data information and the advantages of large-scale language models (Radford et al., 2018, 2019; Brown et al., 2020; Zhou and Xu, 2019; Ouyang et al., 2022). We decompose the synonymous sentence generation task into two stages, each containing several turns of QA, which refer to the dialogue with LLM. LLM is implemented by transforming the zero-shot Similarity Generation (SG) task into a multi-turn question-answering problem with a two-stage framework. Given a sentence x and question prompt q, the model is desired to predict two tuples $T(x) = \{(s_1, y_1), (s_2, y_2)..., (s_n, y_n)\}$, where each tuple $(s_i, y_i) \in \mathbb{R}^{n \times T}$. Formally for an output tuple (s, y), we can express the process as:

$$\Omega((s, y)|x, q) = \zeta(\Omega(s|x, q_1), ..., \Omega(s|x, q_r)), \tag{1}$$

where r is the number of question using the template, ζ is an optimal function.

3.2.1 Stage I

For one sample, this stage generally includes only one turn of QA. In order to find the similar sentences, we first utilize the task-specific templates and the list of sentences to construct the question. Then we combine the question and sentence as input to LLM. To facilitate answer extraction, we ask the system to reply in the list form. If the sentence does not contain any similar sentences, the system will generate a response with NONE Token.

3.2.2 Stage II

This stage generally includes multiple QA turns. In advance, we design a series of specific templates for similar sentence types according to the scheme of the task. The template define a chain of question templates and the length of the chain is usually greater than one. We perform multi turns QA in the order of previously extracted sentence types as well as the order of templates. To generate a question, we need to retrieve the template with the similar sentence type and fill the corresponding slots if necessary. Then we access LLM and get a response. Finally, we compose structured information based on the elements extracted in each turn.

3.3 Semantic Awareness

3.3.1 Word Embedding

We first concat the character-level sentences after data augmentation to form a new sequence

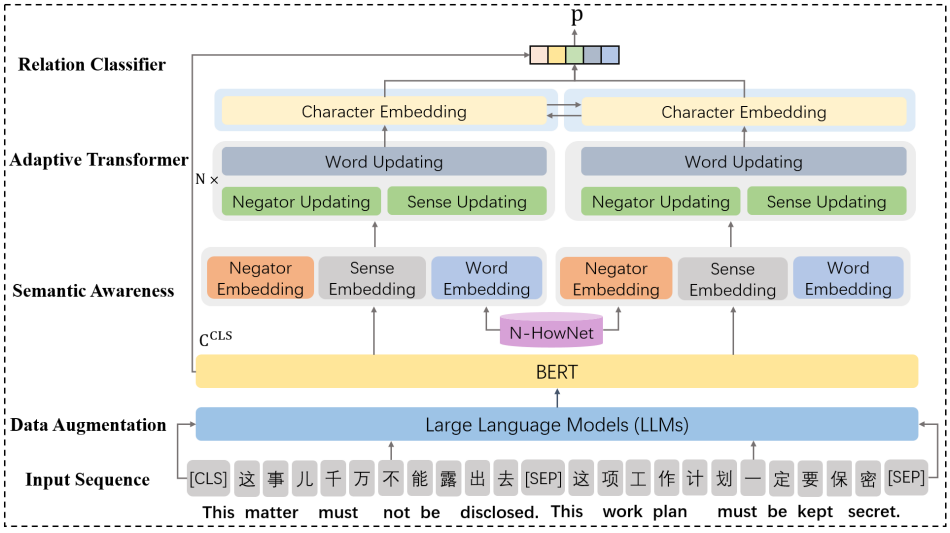

Figure 2: Overview of our proposed DASA model.

$\mathcal{C} = \{[\text{CLS}], c_1^a, ..., c_n^a, [\text{SEP}], c_1^b, ..., c_m^b, [\text{SEP}]\}$, and then feed them to the encoding layer to obtain the contextual representations for each character. Then, we use a feed forward network (FFN) to obtain a feature-wise score vector for each character, which is denoted by ψ. After that, we can normalize it with feature-wise multi-dimensional softmax, which can be formulated as:

$$\mathbf{u}_k = \text{softmax}_k(\psi(\mathbf{c}_k)). \tag{2}$$

The corresponding character embedding $\mathbf{c}_k$ is weighted with the normalized scores $\mathbf{u}_k$ to obtain the contextual word embedding, which can be formulated as:

$$\mathbf{v}_i = \sum_{k=p}^{q} \mathbf{u}_k \odot \mathbf{c}_k \quad (p \le k \le q). \tag{3}$$

3.3.2 Sense Embedding and Negator Embedding

The word embedding $\mathbf{v}_i$ contains only contextual character information, which may suffer from the issue of polysemy in Chinese. In this paper, we incorporate N-HowNet as an external knowledge base that integrates negation words in ancient Chinese to express the semantic information of words.

For each word w_i, we denote the set of senses as $\mathcal{S}^{(w_i)} = \{s_{i,1}, s_{i,2}, ..., s_{i,k}\}$, where $s_{i,k}$ is the k-th sense of w_i. Specifically, if the word w_i contains the negative word, we mark it as Υ_z to indicate negative semantics, where z is the number of negative words. Then we denote its corresponding sememes as $\mathcal{O}_{\Upsilon_z}^{(s_{i,k})} = \{o_{i,k}^1, o_{i,k}^2, ..., o_{i,k}^n\}$. We use multi-dimensional attention function to calculate each sememe's representation $o_{i,k}^n$ as:

$$\mathbf{o}_{i,k}^n = \chi(\mathbf{e}_{i,k}^n, \{\mathbf{e}_{i,k}^{n'} | o_{i,k}^{n'} \in \mathcal{O}_{\Upsilon_z}^{(s_{i,k})}\}), \tag{4}$$

where $\mathbf{e}_{i,k}^n$ is the embedding vector. Then, for each sense $s_{i,k}$, its embedding $\mathbf{s}_{i,k}$ is obtained with attentive pooling of all sememe representations.

$$\mathbf{s}_{i,k} = \varrho(\{\mathbf{o}_{i,k}^n | o_{i,k}^{n'} \in \mathcal{O}_{\Upsilon_z}^{(s_{i,k})}\}). \tag{5}$$

3.4 Adaptive Transformer Encoding

Context information is now separated from semantic information. In order to obtain more useful information, we propose an adaptive word lattice graph transformer. It first takes $\mathbf{v}_i$ and $\mathbf{s}_{i,k}$ as initial word representation $\mathbf{h}_i^0$ for word w_i and initial sense representation $\mathbf{g}_{i,k}^0$ for sense $s_{i,k}$ respectively, and then iteratively updates them with three sub-steps.

3.4.1 Updating Sense and Negator Representation

At u-th iteration, the first sub-step is to update sense representation from $\mathbf{g}_{i,k}^{u-1}$ to $\mathbf{g}_{i,k}^u$. For a word with multiple meanings, which meaning should be used usually depends on the contextual information in the sentence. Therefore, when updating the representation, each sense will first aggregate useful information from the forward and backward words of x_i,

$$\begin{aligned}
\mathbf{r}_{i,k}^{u,fw} &= \chi(\mathbf{g}_{i,k}^{u-1}, \{\mathbf{h}_j^{u-1} | x_j \in \mathcal{Q}_{fw}^+(x_i)\}), \\
\mathbf{r}_{i,k}^{u,bw} &= \chi(\mathbf{g}_{i,k}^{u-1}, \{\mathbf{h}_j^{u-1} | x_j \in \mathcal{Q}_{bw}^+(x_i)\}), \\
\mathbf{r}_{i,k}^{u} &= [\mathbf{r}_{i,k}^{u,fw} \oplus \mathbf{r}_{i,k}^{u,bw}],
\end{aligned} \tag{6}$$

where $\mathbf{r}_{i,k}^{u,fw}$ and $\mathbf{r}_{i,k}^{u,bw}$ are two forward and backward directions parameters of multi-dimensional attention functions, $\mathbf{r}_{i,k}^{u}$ is an aggregation parameter, $\mathcal{Q}_{fw}^{+}(x_i)$ is the set including x_i itself and all its reachable nodes in its forward direction, $\mathcal{Q}_{bw}^{+}(x_i)$ is the set including x_i itself and all its reachable nodes in its backward direction. Then, each sense updates its representation with Γ, which controls the fusion of contextual information and semantic information (Cho et al., 2014),

$$\mathbf{g}_{i,k}^{u} = \Gamma(\mathbf{g}_{i,k}^{u-1}, \mathbf{r}_{i,k}^{u}). \tag{7}$$

The second sub-step is to update the negator representation Υ_z, which can be formulated as:

$$\Upsilon_z = \begin{cases} \Upsilon, z\%2 = 1 \\ \emptyset, z\%2 = 0 \end{cases} \tag{8}$$

where Υ is a sentence with negative semantics, $\emptyset$ is a sentence without negative semantics, $\mathrm{mod}(\%2)$ is a decision operator that determines whether there is negative semantics.

3.4.2 Updating Word Representation

The third sub-step is to update the word representation from $\mathbf{h}_i^{u-1}$ to $\mathbf{h}_i^{u}$. The word w_i first obtains semantic information from its sense representations with the multi-dimensional attention,

$$\mathbf{m}_i^{u} = \chi(\mathbf{h}_i^{u-1}, \{\mathbf{g}_{i,k}^{u} | s_{i,k} \in \mathcal{S}^{(w_i)}\}), \tag{9}$$

and then updates its representation with Γ,

$$\mathbf{h}_i^{u} = \Gamma(\mathbf{h}_i^{u-1}, \mathbf{m}_i^{u}). \tag{10}$$

After multiple iterations, the final word representation $\mathbf{h}_i^{L}$ contains not only contextual word information but also semantic knowledge. For each sentence, we use $\mathbf{h}_i^{a}$ and $\mathbf{h}_i^{b}$ to denote the final word representation respectively.

3.5 Relation Classifier

To incorporate word representation into characters, we use characters in sentence $\mathcal{C}^a$ to introduce the process. For each character c_t^a, we get $\hat{\mathbf{c}}_t^a = \varrho(\{\mathbf{h}_i^a | w_i^a \in \mathcal{W}^{(c_t^a)}\})$ by pooling the useful word information, where $\mathcal{W}^{(c_t^a)}$ is a set including words which contain the character c_t^a. The semantic knowledge enhanced character representation $\mathbf{y}_t$ is given by $\mathbf{y}_t^a = \xi(\mathbf{c}_t^a + \hat{\mathbf{c}}_t^a)$, where $\mathbf{c}_t^a$ is the contextual character representation, ξ denotes layer normalization. We aggregate information from sentence $\mathcal{C}^a$ and $\mathcal{C}^b$ respectively using multi-dimensional attention, which can be formulated as $\mathbf{r}_t^p = \chi(\mathbf{y}_t^a, \{\mathbf{y}_{t'}^a | c_{t'}^a \in \mathcal{C}^a\})$ and $\mathbf{r}_t^q = \chi(\mathbf{y}_t^b, \{\mathbf{y}_{t'}^b | c_{t'}^b \in \mathcal{C}^b\})$ respectively.

Then, we utilize the multi-perspective cosine distance (Wang et al., 2017) to compare $\mathbf{r}_t^p$ and $\mathbf{r}_t^q$,

$$\mathbf{d}_k = \mathrm{cosine}(\mathbf{w}_k^{cos} \odot \mathbf{r}_t^p \odot \Upsilon_z, \mathbf{w}_k^{cos} \odot \mathbf{r}_t^q \odot \Upsilon_z), \tag{11}$$

where $k \in \{1, 2, , ..., P\}$ (P is number of perspectives), $\mathbf{w}_k^{cos}$ is a parameter vector that assigns different weights to different dimensions of messages. Then, we can obtain the final character representation,

$$\hat{\mathbf{y}}_t^a = \mathrm{FFN}([\mathbf{m}_t^p, \mathbf{d}_t]), \tag{12}$$

where $\mathbf{d}_t \triangleq [d_1, d_2, ..., d_P]$, and FFN($\cdot$) is a feed forward network with two layers. Similarly, we can obtain the final character representation $\hat{\mathbf{y}}_t^b$ for each character c_t^b. For each sentence $\mathcal{C}^a$ and $\mathcal{C}^b$, The sentence representation vector $\mathbf{z}_a$ and $\mathbf{z}_b$ can be formulated as $\mathbf{z}^a = \varrho(\hat{\mathbf{y}}_t^a | \hat{\mathbf{y}}_t^a \in \hat{\mathbf{Y}}^a\})$ and $\mathbf{z}^b = \varrho(\hat{\mathbf{y}}_t^b | \hat{\mathbf{y}}_t^b \in \hat{\mathbf{Y}}^b\})$ respectively.

With two sentence vectors $\mathbf{z}^a$ and $\mathbf{z}^b$, our model will predict the similarity of two sentences,

$$p = \mathrm{FFN}([\mathbf{c}^{CLS}, \mathbf{z}^a, \mathbf{z}^b, |\mathbf{z}^a - \mathbf{z}^b|, \mathbf{z}^a \odot \mathbf{z}^b]), \tag{13}$$

With N training samples $\{S_i^a, S_i^b, y_i\}|_{i=1}^{N}$, the objective function $\mathcal{L}_{stm}$ is to minimize the binary cross-entropy loss to train the model:

$$\mathcal{L}_{stm} = -\sum_{i=1}^{n} (y_i log(p_i) + (1 - y_i) log(1 - p_i)), \tag{14}$$

where $y_i \in \{0, 1\}$ is the label of the i-th training sample, $p_i \in \{0, 1\}$ is the prediction of our model.

4 Experiments

In this section, we evaluate our method on manually marked and public datasets, and show that our system outperforms baselines[1]. Accuracy (ACC.) and F1 are used as evaluation metrics for this work.

4.1 Baseline Setting

In this work, the baselines mainly include two groups of models: previous SoTA models and our variant models. We have described the models in the comparisons of our main experiments. The models are listed as follows:

[1]Baseline setting is in Appendix A.2.1 for details.

Table 1: Performance of various models on LCQMC, AFQMC and BQ test datasets. Among them, the results are average scores using different seeds.

Models	LCQMC		AFQMC		BQ	
	ACC.	F1	ACC.	F1	ACC.	F1
BERT (Devlin et al., 2018)	85.73	86.86	73.70	74.12	84.50	84.00
MacBERT (Cui et al., 2021)	86.80	87.78	-	-	84.89	84.29
MacBERT-ext (Cui et al., 2021)	86.68	87.71	74.07	74.35	84.71	83.94
ERNIE (Sun et al., 2019)	87.04	88.06	73.83	73.91	84.67	84.20
LET (Lyu et al., 2021)	84.81	86.08	-	-	83.22	83.03
LET-BERT (Lyu et al., 2021)	88.38	88.85	-	-	85.30	84.98
Text-CNN (He et al., 2016)	72.80	75.70	-	-	78.52	69.17
BiLSTM (Mueller and Thyagarajan, 2016)	76.10	78.90	64.68	54.53	73.51	72.68
Lattice-CNN (Lai et al., 2019)	82.14	82.41	-	-	78.20	78.30
BiMPM (Wang et al., 2017)	83.30	84.90	-	-	81.85	81.73
ESIM (Chen et al., 2016)	82.58	84.49	-	-	81.93	81.87
KSTM (Liu et al., 2023)	89.00	90.20	-	-	87.62	88.44
CLLM-GEN (Liu et al., 2024)	87.00	86.85	-	-	84.450	84.10
CLLM-CLS (Liu et al., 2024)	85.85	85.10	-	-	83.50	83.45
GPT-4 (OpenAI et al., 2024)	87.00	86.95	-	-	84.50	84.45
SA-Trans-STM	86.50	86.20	85.34	83.10	82.70	82.50
DA-Trans-STM	86.15	86.00	85.00	85.90	81.50	83.00
DASA-LSTM-STM	89.61	88.24	86.63	**87.50**	84.98	83.40
DASA-Trans-STM(Ours)	**89.90**	**89.70**	**88.90**	86.80	**86.60**	**85.35**

Previous SoTA Methods. We compare our models with four types of baselines: BERT-based models, representation-based models, interaction-based models and contrastive learning models. BERT-based models mainly include three baselines: BERT, MacBERT and ERNIE. Representation-based models mainly include three baselines: Text-CNN, BiLSTM, and Lattice-CNN. Interaction-based models mainly include two baselines: BiMPM and ESIM. Contrastive learning models mainly include two baselines: SimCSE and ConSERT. To illustrate how well our model can handle short text matching tasks, we compare them with our presented model DASA-Trans-STM.

Variant Models. To analyze the contribution of each component in our model, we ablate the full model and demonstrate the effectiveness of each component.

- **SA-Trans-STM:** This model is a part of our model without the data augmentation component. Obviously, we are to verify the effectiveness of this component on model improvement.
- **DA-Trans-STM:** This model is a part of our model without the semantic-aware component. Obviously, we are to verify the effectiveness of this component on model improvement.
- **DASA-LSTM-STM:** This model is a variant of our model, but we utilize LSTM instead of Transformer in the sequence encoding layer.

4.2 Datasets

We have constructed a new MNS dataset for comprehensive experiments. We used LLMs to collect politically sensitive sentences, such as defense and military, weapons and equipment, industrial information, etc. Other publicly available datasets include LCQMC, AFQMC, and BQ. The LCQMC dataset is a question semantic matching dataset constructed by Harbin Institute of Technology in COLING2018. The AFQMC dataset is the dataset of ANT Financial ATEC: NLP Problem Similarity Calculation Competition, and it is a dataset for classification task. The BQ dataset is a question matching dataset in the field of banking and finance.

Table 2: Performance of various models on MNS test datasets. Among them, the results are average scores using different seeds.

Models	Pre-Training	Interaction	MNS	
			ACC.	F1
BERT (Devlin et al., 2018)	✓	✓	80.15	82.34
MacBERT (Cui et al., 2021)	✓	✓	81.45	81.90
MacBERT-ext (Cui et al., 2021)	✓	✓	-	-
ERNIE (Sun et al., 2019)	✓	✓	83.76	84.38
LET (Lyu et al., 2021)	✓	✓	84.23	83.35
LET-BERT (Lyu et al., 2021)	✓	✓	85.52	86.92
Text-CNN (He et al., 2016)	✗	✗	79.32	80.73
BiLSTM (Mueller and Thyagarajan, 2016)	✗	✗	83.34	81.35
Lattice-CNN (Lai et al., 2019)	✗	✗	82.05	81.84
ESIM (Chen et al., 2016)	✗	✓	84.33	84.34
KSTM (Liu et al., 2023)	✓	✓	-	-
CLLM-GEN (Liu et al., 2024)	✓	✓	86.00	86.10
CLLM-CLS (Liu et al., 2024)	✓	✓	86.50	85.00
GPT-4 (OpenAI et al., 2024)	✓	✓	87.50	85.00
SA-Trans-STM	✓	✓	90.34	90.51
DA-Trans-STM	✓	✓	89.10	89.55
DASA-LSTM-STM	✓	✓	93.56	93.30
DASA-Trans-STM(Ours)	✓	✓	**95.32**	**95.65**

4.3 Discussion on SoTA Methods

The results on public datasets and manually marked datasets[2] are shown in Table 1 and Table 2 respectively. We have gathered several experiment findings from the results. All the experiments in Table 1 and Table 2 are running five times using different seeds and we report the **average scores** to ensure the reliability of results.

First, we can find that the three variants of BERT (MacBERT, MacBERT-ext, ERNIE) all surpass the original BERT, which suggests using word level information during pre-training is important for Chinese matching tasks. Our model DASA performs better than all these BERT-based models. Compared with the baseline BERT which has the same initialization parameters, the ACC. of DASA-Trans-STM on LCQMC, AFQMC and BQ is increased by 4.17%, 15.2% and 2.10%, respectively. It shows that utilizing data augmentation and semantic awareness during fine-tuning phrases with DASA is an effective way to boost the performance of BERT for Chinese semantic matching. We also compare results with K-BERT (Liu et al. 2020), which regards information in N-HowNet as triples {word, contain, sememes} to enhance BERT, introducing soft position and visible matrix during the fine-tuning and inferring phases. The reported ACC. for the LCQMC test set of K-BERT is 86.9%. Our DASA-BERT is 2.71% better than that. Different from K-BERT, we focus on fusing useful information between word, sense and negation.

Second, compared with deep text matching models, we can find that our model DASA outperforms all baselines on public datasets. From Table 1, compared with Lattice CNN, the F1 score of DASA-Trans-STM has increased the most on the LCQMC and BQ dataset, which increased by 14.00% and 16.18% respectively. Similarly, compared with BiLSTM model, the F1 score of DASA-Trans-STM has increased by 32.27% on the AFQMC dataset.

Third, compared with LLMs(GPT-4) (OpenAI et al., 2024), the accuracy of DASA-Trans-STM has increased by 2.90%, 2.10% and 7.82% on the AFQMC, BQ, and MNS dataset. In addition, the performance of LLMs is related to the complexity of prompts.

Finally, from Table 2, we can observe that: **(1)** Compared with other models, DASA-Trans-STM model has the best performance in ACC. (86.30%

[2]Dataset Statistics are in Appendix A.2.2 for details.

on MNS). Our model achieved competitive performance by training on our training set, then evaluating on our testing set. **(2)** DASA-Trans-STM model is based on pre-training and interaction, and is superior to other similar models.

4.4 Ablation Study

All the components of our model play an important role in improving performance. If any component is missing, then the performance will decrease. We also conducted additional experiments on DASA-Trans-STM method with ablation consideration.

We first explore the effects of data augmentation module and adaptive transformer on short text matching tasks. As shown in Figure 7 (see Appendix for details), we have the following observations: **(1)** Compared with the other two variant models, the DASA-Trans-STM model performs best on four datasets. The ACC. of DASA-Trans-STM reached 95.32%, 89.90%, 88.90%, and 86.60%, respectively, with validation loss reduced to 0.005, 0.014, 0.014, and 0.019, respectively. The model reached convergence approximately after 20 iterations. **(2)** Compared with our model, the ACC. of the SA-Trans-STM model have decreased in different degrees (1.57%↓ on MNS, 0.14%↓ on LCQMC, 2.66%↓ on AFQMC, 1.98%↓ on BQ). Therefore, the ChatSG data augmentation component can well solve the issue of unbalanced data labels, and it also greatly promotes the performance of STM system. **(3)** In this study, we used LSTM as the sequence encoding layer to continuously verify the performance of Tranformer. It can be seen from Table 1 that the effectiveness of this model is better than other variant models, only inferior to our model. Most obviously, the F1 score of DASA-LSTM-STM model is 87.50% on AFQMC, which exceeded our model in this indicator (↑0.70%). Therefore, we found that Transformer can play a positive role in improving the performance of STM system by combining sense, negator and word information. **(4)** Specially, we can find that when the model iterates to 80 times, its accuracy has a downward trend in the four datasets, which is caused by being superior to overfitting.

At the same time, in order to test the effectiveness of semantic information, we also set up the experiment without N-HowNet. In this experiment, we remove the embedding and updating of semantic information in the model. Taking AFQMC dataset as an example, through experimental comparison, there is a 1.4% decrease in accuracy and 0.9% decrease in F1 score after removing N-HowNet. This experiment proves that semantic information provided by integrating external knowledge can increase the accuracy of Chinese short text similarity calculation.

5 Conclusion

In this work, we proposed a novel Data Augmentation and Semantic Awareness (DASA) method for Chinese short text matching, which can fully mine the information expressed in Chinese text to deal with word ambiguity. We first use an LLM to generate similar sentences. Then, we further utilize N-HowNet as an external knowledge to integrate sense and negator information to moderate word ambiguity. Specially, we use several segmentation paths to form our lattice graph and construct a set of senses according to the word. Our model takes two word lattice graphs as input. The model was trained and tested in an STM setting. In our view, compared with other pre-training models, it is proved that data augmentation and semantic information can better improve the performance of the model, especially on shorter texts. From the extensive experiments, we empirically demonstrate that our model is superior to most STM systems in the literature. Finally, we obtain 89.70%, 86.80%, 85.35%, and 95.65% F1 on LCQMC, AFQMC, BQ, and MNS datasets respectively.

Acknowledgement

This work is supported by the Research Funds for Institute of Information Engineering, Chinese Academy of Sciences (No. E3V0631104). We extend our gratitude to the anonymous reviewers for their insightful feedback, which has greatly contributed to the improvement of this paper.

Limitations

Due to the wide range of synonyms in the dataset, including many unfamiliar information, manually checking the quality of all text is a considerable challenge. We did not benchmark all baselines because: 1) limited by computation power, some models cannot be fine-tuned on a single NVIDIA A100 with 80GB GPU memory; 2) some models are not publicly available at the moment.

References

Faezeh Ensan and Feras Al-Obeidat. 2019. Relevance-based entity selection for ad hoc retrieval. *Information Processing & Management*, Volume 56, Issue 5, pages 1645-1666.

Negar Arabzadeh, Fattane Zarrinkalam, Jelena Jovanovic, Feras Al-Obeidat, Ebrahim Bagheri. 2020. Neural embedding-based specificity metrics for pre-retrieval query performance prediction. *Information Processing & Management*, Volume 57, Issue 4, pages 102248.

Junmei Wang, Min Pan, Tingting He, Xiang Huang, Xueyan Wang, Xinhui Tu. 2020. A pseudo-relevance feedback framework combining relevance matching and semantic matching for information retrieval. *Information Processing & Management*, Volume 57, Issue 6, pages 102342.

Yang Liu, Wenge Rong and Zhang Xiong. 2018. Improved text matching by enhancing mutual information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32, No. 1, pages 5269-5276.

YueLiu, Aihua Tang, FeiCai, Pengfei Ren, Zhibin Sun. 2019. Multi-feature based Question-Answerer Model Matching for predicting response time in CQA. *Knowledge-Based Systems*, Volume 182, pages 104794.

Tianyong Hao, and Yingying Qu. 2016. QSem: A novel question representation framework for question matching over accumulated question–answer data. *Journal of Information Science*, Volume 42, Issue 5, pages 583-596.

Jinmeng Wu, Tingting Mu, Jeyarajan Thiyagalingam, John Y. Goulermas. 2020. Building interactive sentence-aware representation based on generative language model for community question answering. *Neurocomputing*, Volume 389, pages 93-107.

Kai Yu, Lu Chen, Bo Chen, Kai Sun, and Su Zhu. 2014. Cognitive technology in task-oriented dialogue systems: Concepts, advances and future. *Chinese Journal of Computers*, Vol. 37, No. 18, pages 1-17.

Jianfeng Gao, Michel Galley, Lihong Li. 2018. Neural approaches to conversational AI. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1371-1374.

Jiazhan Feng, Chongyang Tao, Wei Wu, Yansong Feng, Dongyan Zhao, Rui Yan. 2019. Learning a matching model with co-teaching for multi-turn response selection in retrieval-based dialogue systems. *arXiv preprint arXiv:1906.04413*.

Jonas Mueller and Aditya Thyagarajan. 2019. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 30, No. 1, pages 2786-2792.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2016. Siamese recurrent architectures for learning sentence similarity. *arXiv preprint arXiv:1609.06038*.

Yichen Gong, Heng Luo, and Jian Zhang. 2017. Natural language inference over interaction space. *arXivpreprintarXiv:1709.04348*.

Wuwei Lan and Wei Xu. 2018. Neural network models for paraphrase identification, semantic textual similarity, natural language inference, and question answering. In *Proceedings of the 27th International Conference on Computational Linguistics*.

Xiaoya Li, Yuxian Meng, Xiaofei Sun, Qinghong Han, Arianna Yuan, and Jiwei Li. 2019. Is word segmentation necessary for deep learning of Chinese representations? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3242-3252.

Yanzeng Li, Bowen Yu, Mengge Xue, and Tingwen Liu. 2020b. Is word segmentation necessary for deep learning of Chinese representations? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3442-3448.

Yuxuan Lai, Yansong Feng, Xiaohan Yu, Zheng Wang, Kun Xu, and Dongyan Zhao. 2020b. Lattice cnns for matching based chinese question answering. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33, No. 1, pages 6634-6641.

Leonard Bloomfield. 1926. A set of postulates for the science of language. *Language*, Vol. 2, No. 3, pages 153-164.

Zhendong Dong and Qiang Dong. 2003. HowNet-a hybrid language and knowledge resource. In *Proceedings of the international conference on natural language processing and knowledge engineering*, pages 820-824.

Jacob Devlin, Mingwei Chang, Kenton Lee and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *arXiv preprint arXiv:1810.04805*.

Xiaoya Li, Yuxian Meng, Xiaofei Sun, Qinghong Han, Arianna Yuan and Jiwei Li. 2019. Is word segmentation necessary for deep learning of Chinese representations? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3242-3252.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pages 3504-3514.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1904.09223*.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333-2338.

Yuxuan Lai, Yansong Feng, Xiaohan Yu, Zheng Wang, Kun Xu and Dongyan Zhao. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33, No. 1, pages 6634-6641.

Tong He, Weilin Huang, Yu Qiao and Jian Yao. 2016. Text-attentional convolutional neural network for scene text detection. In *IEEE transactions on image processing*, Vol. 25, No. 6, pages 2529-2541.

Zhiguo Wang, Wael Hamza and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*.

Wenpeng Yin, Hinrich Schutze, Bing Xiang and Bowen Zhou. 2017. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, Vol. 4, pages 259-272.

Tianyu Gao, Xingcheng Yao and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. *arXiv preprint arXiv:2105.11741*.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljacic, Shang-Wen Li, Wen-tau 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.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, Vol. 1, No. 8, pages 9.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell and et al. 2019. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 1877-1901.

Wangchunshu Zhou and Ke Xu. 2019. Learning to compare for better training and evaluation of open domain natural language generation models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, No. 5, pages 9717-9724.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray and et al. 2022. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, Vol. 35, pages 27730-27744.

Boer Lyu, Lu Chen, Su Zhu and Kai Yu. 2021. Let: Linguistic knowledge enhanced graph transformer for chinese short text matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, No. 15, pages 13498-13506.

Boer Lyu, Lu Chen, Su Zhu and Kai Yu. 2021. Let: Linguistic knowledge enhanced graph transformer for chinese short text matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, No. 15, pages 13498-13506.

Shulin Liu, Chengcheng Xu, Hao Liu, Tinghao Yu and Tao Yang. 2024. Are LLMs Effective Backbones for Fine-tuning? An Experimental Investigation of Supervised LLMs on Chinese Short Text Matching. *arXiv preprint arXiv:2403.19930*.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal. 2024. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*.

Zhiguo Wang, Wael Hamza and Radu Florian. 2017. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)*, pages 4144-4150.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio. 2014. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724-1734. Association for Computational Linguistics.

Junyi Sun. 2012. Jieba chinese word segmentation tool.

Ruixuan Luo, Jingjing Xu, Yi Zhang, Zhiyuan Zhang, Xuancheng Ren and Xu Sun. 2019. Pkuseg: A toolkit for multi-domain chinese word segmentation. *arXiv preprint arXiv:1906.11455*.

Zhongguo Li and Maosong Sun. 2009. Punctuation as implicit annotations for Chinese word segmentation. *Association for Computational Linguistics*, Vol. 35, No. 4, pages 505-512.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Xin Liu, Qingcai Chen, Chong Deng, Huajun Zeng, Jing Chen, Dongfang Li and Buzhou Tang. 2018. Lcqmc: A large-scale chinese question matching corpus. In *Proceedings of the 27th international conference on computational linguistics (COLING)*, pages 1952-1962.

Liang Xu, Xuanwei Zhang and Qianqian Dong. 2020. CLUECorpus2020: A large-scale Chinese corpus for pre-training language model. *arXiv preprint arXiv:2003.01355*.

Jing Chen, Qingcai Chen, Xin Liu, Haijun Yang, Daohe Lu and Buzhou Tang. 2018. The bq corpus: A large-scale domain-specific chinese corpus for sentence semantic equivalence identification. In *Proceedings of the s2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4946-4951.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, Vol. 15, No. 1, pages 1929-1958.

Ruiqiang Liu, Qiqiang Zhong, Mengmeng Cui, Hanjie Mai, Qiang Zhang, Shaohua Xu, Xiangzheng Liu and Yanlong Du. 2023. The Short Text Matching Model Enhanced with Knowledge via Contrastive Learning. *arXiv preprint arXiv:2304.03898*.

A Appendix

In this appendix, we first clarify more details about LLM for data augmentation in Section A.1. Then, we provide more details about experimental settings, including baseline setting, dataset statistics and hyper-parameters setting in Section A.2. Afterwards, we provide more details about parameter sensitivity analysis and case study in Section A.3. Finally, we introduce the computational infrastructure of our model in Section A.4.

A.1 Data Augmentation

We use LLM pretrained language models. These models are trained on a broad distribution of Internet data and are adaptable to a wide range of downstream tasks, but have poorly characterized behavior. Starting from these models, we then train models with three different techniques:

Supervised fine-tuning (SFT). The input for this stage is to randomly select a batch of data from the prompts submitted by the test user. Then, we manually perform high-quality responses on the extracted prompt data to obtain <prompt, answer> data pairs. By fine-tuning the GPT-3 model with high-quality answers, it helps the model better understand input instructions in the first stage.

Reward modeling (RM). The RM structure is the model that removes the final embedding layer of the SFT trained model. Its inputs are Prompt and Reponse, and its output is reward value. For each Prompt, LLM will randomly generate k outputs ($4 \leq k \leq 9$). In order to speed up comparison collection, we present labelers with anywhere between k = 4 and k = 9 responses to rank. This produces $\binom{k}{2}$ comparisons for each prompt shown to a labeler.

Specifically, the loss function for the reward model is:

$$\begin{aligned} \text{loss}(\theta) = -\frac{1}{\binom{k}{2}} E_{(x,y_w,y_l)\sim D} \\ [\log(\sigma(r_\theta(x,y_w) - r_\theta(x,y_l)))], \end{aligned} \tag{15}$$

where $r_\theta(x, y)$ is the scalar output of the reward model for prompt x and completion y with parameters θ, y_w is the preferred completion out of the pair of y_w and y_l, and D is the dataset of human comparisons.

Reinforcement learning (RL). we fine-tuned the SFT model on our environment using PPO (Schulman et al., 2017). The environment is a bandit environment which presents a random customer prompt and expects a response to the prompt. Given the prompt and response, it produces a reward determined by the reward model and ends the episode. In addition, we add a per-token KL penalty from the SFT model at each token to mitigate over-optimization of the reward model.

We also experiment with mixing the pretraining gradients into the PPO gradients, in order to fix the performance regressions on public NLP datasets. We call these models "PPO-ptx". We maximize the following combined objective function in RL training:

$$\begin{aligned} \text{objective}(\hbar) = E_{(x,y)\sim D_{\pi_\hbar^{\text{RL}}}}[r_\theta(x,y) \\ -\beta \log(\pi_\hbar^{\text{RL}}(y|x)/\pi^{\text{SFT}}(y|x))] \\ +\gamma E_{x\sim D_{pretain}}[log(\pi_\hbar^{\text{RL}}(x))], \end{aligned} \tag{16}$$

where $\pi_{\emptyset}^{\mathrm{RL}}$ is the learned RL policy, π^{SFT} is the supervised trained model, and D_{pretrain} is the pretraining distribution. The KL reward coefficient β, and the pretraining loss coefficient γ, control the strength of the KL penalty and pretraining gradients respectively. For "PPO" models, γ is set to 0.

A.2 Experimental Settings

A.2.1 Dataset Statistics

MNS Datasets. We have constructed a new MNS dataset for comprehensive experiments. In particular, we have used ChatGPT to collect politically sensitive sentences, such as defense and military, weapons and equipment, industrial information, etc. Then, the generated similar sentences are checked and any ambiguities or irregularities are corrected. The label of each sentence is manually marked. As shown in Table 3, the data volumes of training set, test set and development set are 2,760, 1,660, and 1,890, respectively, and the total number of all samples is 6,310.

Public Datasets. We conducted experiments on three mainstream Chinese short text matching benchmarking datasets.

- **LCQMC (Liu et al., 2018):** Its format consists of sentence pair number, two sentences to be compared and four columns of similarity labels. It contains 260,068 pieces of data in total, including 238,766 for training set, 12,500 for test set and 8,802 for development set. Each pair is associated with a binary label indicating whether two sentences have the same meaning or share the same intention. Positive samples are 30% more than negative samples.

- **AFQMC (Xu et al., 2020):** All data are from the actual application scenarios of Ant Financial's financial brain, that is, two sentences described by users in a given customer service are determined by algorithms to determine whether they represent the same semantics. The data volumes of training set, test set and development set are 61,486, 20,496, and 20,495, respectively, and the total number of all samples is 102,477.

- **BQ (Chen et al., 2018):** Comprising question text pairs extracted from one year of online banking system logs, it is the largest question matching dataset in the banking domain. The BQ dataset contains 120,000 pieces of data in total, including 100,000 for training set, 10,000 for test set and 10,000 for development set. The number of positive and negative samples are the same.

A.2.2 Hyper-parameters Setting

The input word lattice graphs are produced by the combination of four segmentation tools: jieba (Sun et al., 2012), pkuseg (Luo et al., 2019), thulac (Li and Sun, 2009) and snownlp. Table 4 shows the values of hyper-parameters for our models, which as fixed according to previous work in the literature without grid-search adjustments for each individual dataset. Specifically, the dimensions of both word and sense representation are 128. The hidden size is also 128. We dynamically adjust the learning rate and automatically decrease the learning rate lr according to the decrease of loss. Dropout (Srivastava et al., 2014) is applied to both word and sense embeddings with a rate of 0.2. Stochastic gradient descent (SGD) is used for optimization, with an initial learning rate of 0.0015 and a warmup rate of 0.1. As for batch size, we use 32 for NSC, LCQMC and 64 for AFQMC, BQ.

A.3 Experimental details

A.3.1 Parameter Sensitivity

We evaluate our model on different settings of the parameters. Specifically, we are concerned about the impact of dropout, learning rate decay, text length and segmentation.

Influences of dropout on performance. We compared the results achieved by our model with and without dropout layers, and show those results in Table 5. All other hyper-parameters remain the same as our best model. After using dropout, the F1 score has improved in each dataset. This demonstrates the effectiveness of dropout in reducing overfitting. Dropout is essential for state of the art performance, and the improvement is statistically significant. Our model achieved an essential and improved performance, because of introducing dropout.

Influences of learning rate decay on performance. We analyzed the parameter sensitivity of learning rate decay, and compared the results achieved by our model with and without learning

Table 3: Statistics of four benchmarking datasets.

Datasets	Types	Train	Test	Dev	Total
LCQMC	Question semantic matching	238.766k	12.500k	8.802k	260.068k
AFQMC	ANT financial	61.486k	20.496k	20.495k	102.477k
BQ	Question semantic matching	100.000k	10.000k	10.000k	120.000k
MNS	Military network security	2.76k	1.66k	1.88k	6.31k

Table 4: Hyper-parameter values.

Parameter	Value	Parameter	Value
initial learning rate lr	0.0015	lr weight decay	0.05
warmup rate	0.1	bert lr mult	30
batch size	32	iterations $iter$	100
embedding dim	128	rate of embeddings	0.2
dropout	0.5	layer size	128

rate decay. Similarly, all other hyper-parameters remain the same as our best model. After using learning rate decay, the accuracy has improved on each dataset (see Table 5). Therefore, learning rate decay is very effective in finding global optimization.

Influences of text length on performance. We evaluated our model on different text length. From Table 5, we can observe that the shorter the text length, the more obvious the improvement effect of utilizing sense information. For example, when the text length is$\leq$15, our model achieves the best performance. Compared to text length $\geq$22, the F1 score has been improved in different degrees (4.04%↑ on LCQMC, 2.53%↑ on AFQMC, 3.33%↑ on BQ, 3.13%↑ on MNS). On the one hand, the reason is that concise texts usually have rare contextual information, which is difficult for model to understand. However, N-HowNet brings a lot of useful external information to these weak-context short texts. Therefore, it is easier to perceive the similarity between texts and gain great improvement. On the other hand, longer texts may contain more wrong words caused by insufficient segmentation, leading to incorrect sense information. Too much incorrect sense information may confuse the model and make it unable to obtain the original semantics. Therefore, long texts should focus more on contextual information and fully explore the semantic feature information of forward and backward sentences. Instead, short text does not contain enough contextual information, N-HowNet can provide more semantic information.

Influences of segmentation on performance. To explore the impact of using different segmentation inputs: jieba, pkuseg, thulac and snownlp, we carry out experiments using DASA-Trans-STM on LCQMC, AFQMC, BQ, and MNS test datasets. As shown in Table 5, we can find that jieba performs better on LCQMC, AFQMC, and BQ datasets. Specifically, pkuseg performed better on the MNS datasets, with an F1 score of 95.20%. The reason may be that pkuseg is more suitable for segmentation in military network security scenarios. Overall, segmentation tools have a significant impact on the performance of our model. If the segmentation does not contain the correct word, our semantic information will not exert the most significant advantage.

A.3.2 Case Study

We compare DASA-Trans-STM between the model with and without sense and negator information. The model without sense and negator fails to judge the relationship between sentences which actually have the same intention, but DASA-Trans-STM performs well. From Table 6 and Table 7, the content in red, blue, and green represents polysemy, negation, and important entity words, respectively. In the first case, we observe that both sentences contain the word " 保密工作(confidential work)", which has only

Table 5: List of parameter sensitivity analysis, including the influences of dropout, learning rate decay and text length on LCQMC, AFQMC, BQ, and MNS test datasets (F1 score).

Parameter Category	Variable	LCQMC	AFQMC	BQ	MNS
dropout	✓	89.70	86.80	85.35	95.65
	✗	87.90	85.50	83.28	93.36
learning rate decay	✓	89.70	86.80	85.35	95.65
	✗	88.35	85.37	84.90	94.50
text length	≤15	90.89	87.63	86.83	96.48
	16-18	89.50	86.95	85.73	95.88
	19-21	89.35	86.50	85.11	95.10
	≥22	86.85	85.10	83.50	93.35
segmentation	jieba	87.94	88.73	87.38	94.10
	pkuseg	87.82	86.01	84.35	95.20
	thulac	87.50	85.10	84.50	94.50
	snownlp	87.20	84.45	83.12	93.10

one sense described by sememe "work". Moreover, the sense of "摸鱼(slack off)" has two sememes "水中摸鱼(fishing in the water)" and "偷懒(laziness)". Among them, the second sememe is more compatible with Text 2. In the second case, there are two common sememes " 这项任务(this task)" and "这项工作(this job)". In Text 1, there are two sememes for "啃硬骨头(bite the bullet)", one is to " 啃食骨头(Gnaw on bones)", and another is "不怕困难(Not afraid of difficulties)". It provides a powerful message that makes "啃硬骨头(bite the bullet)" attend more to the first sense.

Figure 3 and Figure 4 in Appendix illustrate the 2-shot prompts for LCQMC and BQ, respectively. The performance of 2-shot GPT-4 on BQ is much worse than that of supervised models. This is mainly because BQ is a dataset of real customer service questions from WeBank Inc., and a full understanding of the sentences' meaning requires background information about this bank. For example, questions in BQ usually mention specific products or a particular function in the bank's app. This background knowledge is unknown to LLMs and is also impossible to provide entirely in the prompt.

In addition, we conducted an analysis of qualitative fault cases, including segmentation-induced errors, incorrect sense assignments, and bad LLM outputs. (1) Segmentation Errors: E.g., "苹果手机 (Apple phone)" missegmented as "苹果 (apple)" + "手机 (mobile)" -> corrected via lattice LSTM + N-HowNet disambiguation. (2) LLM

#two-shot prompt for LCQMC

请判断以下两个句子Q1和Q2的语义是否完全相同。输出格式：先输出"相同"或"不同"，再给出原因。
示例1：
Q1：打印机和电脑连接应该如何设置
Q2：如何把无线的电脑连接到打印机上
输出： 不同。原因：Q1未明确说明连接方式，可能包含有线连接或无线连接；而Q2明确限定为"无线"连接，排除了其他可能性。因此两者语义范围不同。
示例2：
Q1：看图猜一电影名
Q2：看图猜电影
输出： 相同。原因：Q1中的"猜一电影名"和Q2中的"猜电影"在常见语境下均指向通过图片猜测对应的电影名称，核心任务一致。虽然Q1强调"一"个具体名称，但Q2的"猜电影"在同类场景中通常默认指向猜名称（而非其他属性），因此二者语义范围无本质差异。

Figure 3: An illustration of 2-shot prompt for LCQMC.

#two-shot prompt for BQ

请判断以下两个句子Q1和Q2的问题意图是否相同。相同的判断标准：问题询问的核心意图相同就算相同，问题中的时间、数值的差异可以忽略。输出格式：先输出结论，"相同"或"不同"，再给出原因。
示例1：
Q1：为什么我无法看到额度
Q2：为什么开通了却没有额度
输出： 不同。原因：Q1询问的是无法看到额度的原因，可能包含未开通、系统故障等多种情况；而Q2明确限定为"已开通但无额度"，聚焦于开通后的异常问题（如审核未通过、信用评分不足等）。二者问题场景的限定范围不同，核心意图存在差异。
示例2：
Q1：下周有什么好产品
Q2：一月份有哪些理财产品
输出： 不同。原因：Q1询问的是"下周"推荐的"好产品"，范围较广且未限定产品类型（可能包括理财、基金等）；而Q2明确聚焦于"一月份"的"理财产品"，时间范围和产品类型均更具体。虽然时间差异可忽略，但Q2限定了产品类别，而Q1未明确类型，导致二者核心意图的覆盖范围不同。

Figure 4: An illustration of 2-shot prompt for BQ.

Table 6: Case 1 using polysemy and negator information to get the correct answer.

Case 1: Text	**Sememe**
Text 1: 对待保密工作不能摸鱼 Don't slack off when dealing with confidential work	摸鱼→ Sense1: 水中摸鱼(Fishing in the water) Sense2: 偷懒(Laziness)
Text 2: 对待保密工作不能偷懒 Don't be lazy when dealing with confidential work	偷懒→ Sense1: 偷懒(Being lazy) Sense2: NULL

Table 7: Case 2 using polysemy information to get the correct answer.

Case 2: Text	**Sememe**
Text 1: 这项任务难度很大, 大家要敢于啃硬骨头 This task is very difficult, everyone should dare to bite the bullet	啃硬骨头→ Sense1: 啃食骨头(Gnaw on bones) Sense2: 不怕困难(Not afraid of difficulties)
Text 2: 这项工作难度很大, 大家要敢于攻克 This job is very difficult, everyone should dare to conquer it	攻克→ Sense1: 克服困难(overcome difficulties) Sense2: NULL

#two-shot prompt for AFQMC

请判断以下两个句子Q1和Q2的语义是否完全相同。输出格式：先输出"相同"或"不同"，再给出原因。
示例1:
Q1：你可以帮我删掉借呗里的账单记录吗
Q2：蚂蚁借呗还款记录可以清除吗
输出： 不同。原因：Q1询问的核心意图是请求删除账单；而Q2明确限定为"还款记录"的清除可行性。前者范围更广，后者仅针对还款行为产生的记录。二者针对的记录类型和操作对象存在差异，因此两者核心意图不同。
示例2:
Q1：怎么把花呗的款补上
Q2：花呗怎么领饲料
输出：不同。原因：Q1核心意图是了解还款的操作流程，而Q2中"领饲料"通常关联特定活动，意图是获取活动奖励而非处理还款。二者涉及的花呗功能方向相反，因此核心诉求无关联。

Figure 5: An illustration of 2-shot prompt for AFQMC.

#two-shot prompt for MNS

请判断以下两个句子Q1和Q2的语义是否完全相同。输出格式：先输出"相同"或"不同"，再给出原因。
示例1:
Q1：领导的批示非常重要，我们必须要认真贯彻
Q2：领导的指示至关重要，我们必须认真贯彻执行
输出： 相同。原因：Q1中的"批示"和Q2中的"指示"均指领导提出的要求或指令，核心意图均强调对领导要求的重视和执行必要性。二者在常见上下文中均指向需遵循的上级决策，因此问题意图的核心（强调执行领导要求）无本质差异。
示例2:
Q1：这项工作是我们的重中之重，不可有任何疏漏
Q2：这项工作是我们的首要任务，不可有任何漏洞
输出： 相同。原因：Q1中的"重中之重"与Q2中的"首要任务"均强调工作的最高优先级和关键性，核心意图均指向对该项工作的高度重视和严格完成要求。后半句"不可有任何疏漏"和"不可有任何漏洞"虽用词不同，但均要求规避错误，本质诉求相同。因此，二者问题意图在强调工作重要性和执行严谨性上一致。

Figure 6: An illustration of 2-shot prompt for MNS.

Noise: <1% of augmented sentences reduced accuracy; filtering heuristics (e.g., semantic similarity >0.8) removed low-quality paraphrases. (3) Sense Mismatches: In MNS, "突击 (assault)" assigned incorrect military sense -> resolved via task-specific fine-tuning of sense embeddings.

A.3.3 Cross-Lingual Adaptation

To discuss cross-lingual applicability, we briefly outline how this method can be extended to other languages using resources like English WordNet and Japanese dictionaries to emphasize broader applicability. (1) English Experiments: Test on MRPC (Paraphrase Detection) using WordNet synsets as semantic nodes, replacing HowNet. (2) Japanese Validation: Evaluate on Japanese Paraphrase with EDR dictionary integration (ongoing trials show approximately 78.5% accuracy).

A.4 Computing Infrastructure

All the experiments are conducted on Nvidia GeForce MX250 GPUs (32GB memory). Other configuration includes 2 * Intel(R) Core(TM) i7-10510U CPU @1.80GHz, 500GB DDR4 RAM and 2 * 512GB M.2 SSD, which is sufficient for all the baselines.

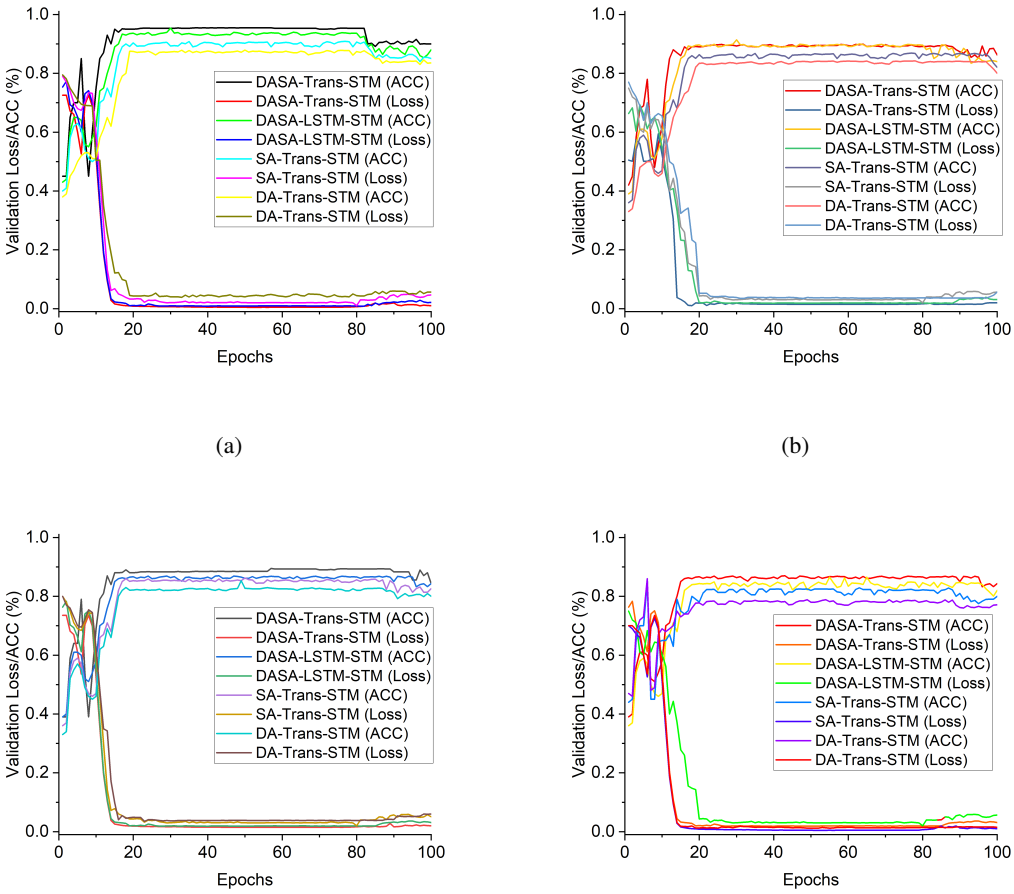

Figure 7: Ablation studies: (a) Performances on MNS datasets; (b) Performances on LCQMC datasets; (c) Performances on AFQMC datasets; (d) Performances on BQ datasets.