Semantic Matching Competition: Is Bert Itself A Good Answer

Abstract

The Natural language sentence matching (NLSM) task is of great significance for both industrial community and NLP community. Driven by the GAIIC competition(GAIIC, 2021), this work aims to investigate how various technologies can help adapting Chinese corpus pre-trained BERT into the desensitized data domain and achieving higher performance, which include data augmentation, self-ensemble, self-distillation, multi-task training strategy, embedding alignment, etc. Through detailed experiments, we verified the effectiveness of proposed method and by integrating them into the final model we successfully improved the BERT baseline from 0.9054 to 0.9548 AUC score and narrowed the offline-online AUC gap from 0.1039 to 0.0714.

1. Introduction

Natural language sentence matching (NLSM) is one of the most fundamental topics in the field of NLP. The aim of it is to identify the relationship between two sentences and is widely applied in sentence paraphrase, natural language inference and machine comprehension tasks. In industrial Question-Answering system and Dialogue system, NLSM is the common solution for intention recognition.

To be specific, what semantic matching deals with is Query-Question matching: Firstly, QA system usually maintains a large and high-quality standard question-answer set. Secondly, the system will roughly retrieve(i.e. BM25, TF-IDF, (Robertson & Zaragoza, 2009)) a batch of questions according the query from users. In this batch of questions, semantic matching will predict which one is the most semantically coherent with the query and then return the pre-set answer. In this scenario, the experience of users is of importance which gives high demands on the precision and speed of semantic matching.

After the great success of BERT(Devlin et al., 2018) on various downstream tasks, there are a lot of researches aiming to improve the pre-training method and significantly promote the progress of NLP community. ERINE(Sun et al., 2019) improves the masking process of BERT through entity-level masking and phrase-level masking; XLNet(Yang et al., 2019b) solves the difference in pre-training and fine-tuning phase by replacing direct mask with Permutation Language Model(PLM). RoBERTa(Liu et al., 2019) evaluates the training of BERT and proposes points such as deleting the NSP task loss and using dynamic masking to let the model more robust. Nevertheless, these methods mainly focus on the pre-training phase, while the fine-tuning phase on the NLSM is not thoroughly studied.

In this work, we choose the BERT(base) model as our baseline and fine-tune it on the GAIIC 2021 competition, which provides a new short-text, open-domain data set. This competition is supported by OPPO, a mobile phone company, who would like to improve their QA system in the mobile voice assistant. It is worth noticing that this is a desensitized data, which is common and practical in the industry due to ethical issues. For instance, when providing semantic matching service for client companies, they can not afford the risk of leaking private information, the query samples from their customers, so we would have to train our model and design business strategy on desensitized data. In addition, as described in (Lample & Conneau, 2019), this could also be considered as the situation that we need to adapt the BERT model which is pre-trained on a source language into a new language which holds certain patterns similar to the source language (in this case, the original Chinese corpus and the desensitized data share similar patterns and properties).
Therefore, this work aims to explore how to utilize pre-trained BERT for NLSM task on desensitized data. We also evaluate how technologies impact the fine-tuning phase of the model and the final performance, which includes data augmentation, self-ensemble, self-distillation, multi-task training strategy, embedding alignment, etc.

2. Data set and task

2.1. GAIIC2021 Competition

In this section, we will describe the dataset we used, the task we performed as well as the model evaluation metric we applied. Our dataset (GAIIC, 2021) which consists of 100000 labeled training data and 25000 unlabeled test data, is provided by a TIANCHI competition named as Global AI Innovation Contest(GAIIC). Each sample of the training data consists of two short texts and a binary label. The test data share the same format with training data but without labels. Most importantly, **all the texts has been desensitised to numbers** for the sake of the privacy of the customers, which makes our work more challenging and interesting. In the rest of the paper we will refer the desensitized text as ciphertext, while the original Chinese text as plaintext. An example of our training data is shown below.

As shown in the table above, the texts have already been tokenized in unknown ways, thus we cannot perform different tokenization or other pre-processing methods which

Text 1	Text 2	Label
29 35 36 29 71	29 37 36 29 82	1
1 2 3 4 100 502	11 13 15 18 981	0
My name is Breeno	Breeno is my name	1
I don't want	I don't want to say anything	0

Table 1. Example of positive and negative sample in ciphertext and plaintext

could potentially be beneficial to our task(For better understanding, English queries are displayed here).

Using the dataset described above, our task is to build a model for classifying the data items into two classes according to their semantic similarity, where 1 represents that text 1 and text 2 have the same semantic meaning, while 0 means the opposite way. Our task shares a certain similarity with the LCQMC (Large-scale Chinese Question Matching Corpus) task (Liu et al., 2018), which could be seen as the Chinese version of the well known QQP (Quora Question Pairs) task (Iyer et al., 2017). However, compared with LCQMC, our data are desensitized, which makes our task harder. Besides, our training data are unbalanced (36% positive samples and 64% negative samples), which could jeopardize the model's performance, especially when this distribution for the test data varies significantly.

2.2. Evaluation Measure

As the requirement of the competition, we adopted the AUC score as our model's evaluation metric. AUC score is the area under the ROC curve, which can be calculated as

$$AUC = \frac{\sum_{i \in dev\_set} \text{rank}(i) - \frac{M(1+M)}{2}}{M * N} \tag{1}$$

where rank(i) is the rank of i^{th} data's prediction score in the development set, M is the number of data which is labeled as 1 and N is the number of data which is labeled as 0. AUC score is in the range of $[0.5, 1]$, where 0.5 represents the worst case a classifier can perform (it learned nothing or just learned the distribution of the labels) and 1 means the classifier is perfect. Different from precision and recall which can be cheated easily by predicting the most faithful sample as 1 only and predicting all samples as 1, the AUC metric will give the lowest score (0.5) to both of these two cases. Meanwhile, compared to accuracy which is the most common metric for classification tasks, AUC score will give the lowest score to classifiers which make predictions stochastically or predicting one label constantly, while accuracy may be high due to the unbalanced label distribution. Therefore, AUC is a good metric for our task which is performed on an unbalanced dataset.

3. Methodology

In this section, we will firstly state the overall direction of our solutions and the reasons for doing so. Then we will detail the methods we adopted and the motivation for adopting them in each corresponding subsection. The results and analysis of taking these methods will be illustrated in section 4.

In this project, we selected BERT (base) as our baseline (detailed shown in subsection 4.1) for performing sentence similarity classification (binary) tasks on ciphertext data. Firstly, the powerful BERT model narrowed our options of modifying the model's architecture, because it has taken the interactions between embedding to the extreme by adopting multi-head attention mechanism and deep bidirectional architecture, which has been shown to be the critical point for achieving success in semantic matching tasks and many downstream tasks (Wang et al., 2017). It may be also the reason why many variants of BERT mainly focus on modifying the pre-training strategy with the architecture unchanged, such as BERT$_{\text{WMM}}$ (Cui et al., 2019) adopted whole word mask on Chinese and RoBERTa (Liu et al., 2019) adopted dynamic mask with NSP (next sentence prediction) task canceled. Secondly, the ciphertext data narrowed our options for architecture modification further, partly because we cannot exploiting external knowledge in BERT like what has been done in (Yang et al., 2019a), but also because we have no other features except the ciphertext. Therefore, we mainly focus on training strategy, the classifier architecture, data augmentation as well as ensemble and distillation, with the whole BERT architecture unmodified. The following subsections will describe all these five methods in detail correspondingly.

3.1. Baseline: Introduction & Selection

At the start of our project, we selected BiMPM (Bilateral Multi-Perspective Matching) model (Wang et al., 2017) and well-known BERT Model (Devlin et al., 2018) as our two candidates for baseline. BiMPM gave the state-of-art result on LCQMC task before BERT and BERT have significantly promoted all tasks on GLEU benchmark that includes QQP(Question Query Pairing) task which could be seen as the English version of LCQMC. As mentioned in section 2, on account of our task share great similarity with the LCQMC task, both of the two models are worth trying. Here we will introduce the two models briefly and leave the comparison & analysis to section 4.1.

BiMPM adopted an architecture consisted of two BiLSTM layers and a "multi-perspective" matching layer between them. To be more specific, the first BiLSTM layer will encode two given sentences text 1 and text 2 to two series of token embeddings, then the two encoded sentences will be matched bidirectionally. In each matching direction, every token embedding of text 1 or text 2 will be matched to all the token embeddings of another encoded sentence using multiple strategies (perspectives). Finally, the last BiLSTM layer is applied for aggregating all the matching results into a fixed-length vector, which will be used for performing classification tasks by a fully connected layer. The main idea of BiMPM is rather a sample that is first produce contextualized dynamic embedding for each tokens using BiLSTM and let them be fully interacted with tokens came from another sentence to achieve fine-grained interaction between sentences and then use another BiLSTM integrated all the matching results for classification.

Different from BiMPM which has a task-specific architecture, BERT is a powerful pre-training model, which means a pre-trained BERT can achieve state-of-art results on various tasks with only one additional classification layer added and without significant task-specific modification of architecture. The input of the BERT model can be one sentence or two sentences with a special token [CLS] added at the beginning and another special token [SEP] added at both the middle of two sentences and the ending position: [CLS] text1 [SEP] text2 [SEP]. Except for the [CLS] token, BERT will output a contextualized token embedding for each token of the input at the corresponding position. While the output at [CLS] position is the result of the pooling of remaining token embeddings, which is usually used in downstream classification tasks. BERT's great capacity mainly comes from adopting transformer (vas) and deep bidirectional architecture. To be more specific, compared with ELMo (Embeddings from Language Models) (Peters et al., 2018) which is a powerful pre-training model proposed before BERT, BERT is mainly benefited from the fact that it adopted transformer blocks rather than the LSTM units used in ELMo. The usage of transformer block also makes BERT a deep bi-directional model as it enables BERT to implement much complete and fine-grained interaction between embeddings, while ELMo is a superficial bi-directional model for its only concatenated the results of two uni-directional LSTM together. Compared with GPT (Generative Pre-Training) model (Radford et al., 2018), which had a uni-directional architecture by using the attention mask for performing a generative task such as machine translation, BERT adopted bidirectional architecture which makes it more powerful for classification tasks.

Through experiments and analysis shown in section 4.1, we ultimately selected Bert as our Baseline model for its much higher AUC score achieved on our ciphertext data. The classical usage of Bert in downstream tasks can be divided into fine-tuning-based strategy and feature-based strategy, where fine-tuning-based strategy adds an additional layer for classification and feature-based strategy combines contextualized token embeddings produced by Bert with other features to perform downstream tasks. On account of computational resource limitation and the lack of other information and features, $\text{BERT}_{\text{Base}}$ pre-trained on Chinese corpus (the plaintext data version of GAIIC2021 is in Chinese) combined with classical fine-tuning strategy becomes our baseline.

3.2. Enhanced Training Strategy: Combination of Supervised and Self-supervised Learning

Even though the fine-tuning based strategy works well and brings us about 0.91 AUC score, we think it does not fully exploit the capacity of BERT model, because the complexity of our classification task is disproportionate with the number of parameters that need to be adjusted, that is the complexity of the "assignment" does not match the talent of a "student", so the "student" can learn few from the task. In another word, the relatively simple classification task cannot adjust huge numbers of parameters to optimal values. Meanwhile, the vanilla training strategy does not take full advantage of data available neither, for it only uses training data, when test and validation data could make contribution to the generalization and fitting ability of the model. Therefore we add an auxiliary task for promoting BERT's optimization procedure. Inspired by the idea of predicting masked token (Masked Language Model, MLM) (Devlin et al., 2018) used in BERT's pre-training procedure, here we adopted a new training strategy that combines original similarity classification task with masked token prediction task by deploying MLM in fine-tuning procedure. In order to perform these two tasks together, we have to ensure the development and test data attend the secondary task only and will never engage with the similarity classification task. Inspired by the attention mask (vas) that is used in transformer's decoder for preventing the ground truth label from being learnt by current generation task, we assign the data in test and development set a fake label which will be masked out when doing similarity classification task. Meanwhile, we also need to handle the inconsistency of two prediction tasks, where the output of similarity classification task is a binary label, while the output of MLM task is the index of the token in the vocabulary. To this end, we add two special tokens into vocabulary that are "same" and "different" which represent 1 and 0 label respectively. By doing so, we transform the binary similarity classification task into a multi-dimensional token prediction task successfully, which is consistent with the masked word prediction task.

Create MLM Task: According to the original paper(Devlin et al., 2018), we firstly choose 15% words of two sentences randomly. Then for these selected words, we randomly substitute 80% percents of them with a special token [MASK], 10% of them with a random token in the vocabulary and with the remaining 10% of words unchanged. Need to mentioned that, different with static masking strategy used in original paper, here we adopted a dynamic masking strategy, that is for a same sentence the words that are chosen to be substituted by [MASK] are unfixed for each epochs, which has shown to be beneficial for many tasks that include LCQMC (Liu et al., 2019).

Create Prediction Labels: As mentioned above, we adopted a multi-task fine-tuning strategy, thus our prediction labels contain the ground truth value for both tasks at corresponding positions. For the masked word prediction task, the labels are either the vocabulary indexes of masked words that include the randomly substituted words or the index of padding character 0. For example if the original input is $[22, 34, 56, 78]$ and randomly masked sequence is [[MASK], 34, 88, 78], then the corresponding labels for masked word prediction task will be $[22, 0, 56, 0]$. For the similarity classification task, the label will be either the vocabulary index of "same"/"different" for training set or index for padding character 0, that is if the data comes from training set then its label will be 5 or 6 according to its binary label is 0 or 1 respectively, otherwise the label will be 0 which means the data comes from test or development set. After obtaining the labels for both tasks, we concatenate them together in the format of [label_similarity]+[text1_MLM]+[SEP_index].+[text2_MLM] + [SEP_index], which is consistent with the input and output format.

Create Outputs Mask: As mentioned above we need to ensure test and development set will not attend the similarity classification task for preventing data leakage and we also need to ensure the unchanged tokens will not attend the masked word prediction to save computational resources. We achieve both purposed by adopting outputs mask which will mask out all the outputs that at the position with label equal to the index of padding character that includes the outputs of test and development data, unchanged token as well as the padding character. The implementation of outputs mask is rather simple, as we only keep the outputs at the positions with label not equal to zero [labels!=0].

By adopting this enhanced training strategy, we make fuller use of the capacity of BERT, taking full advantage of data available and beat the baseline within 10 epochs, while the baseline has been trained for 30 epochs. Meanwhile, by using this new strategy we can do the two tasks jointly rather than do pre-training and fine-tuning separately, which not only greatly promote the convergence speed, but also save lots of time and computational resources. Additionally, by adopting the MLM auxiliary task, the Bert could been seen as has been better pre-trained on the new desensitization language, which can help our model generalize to other unseen data better, just like a well pre-trained Bert can be beneficial to all downstream tasks on the same language.

3.3. Embedding Alignment: Solutions of Frequency Analysis and Word2vec

Fine-tuning based strategy usually converge fast on most of tasks (3 or 4 epoch), when the model has been fully pre-trained on a large amount of data in the same language. However, in our case, the desensitization data is a brand new digital language for Chinese pre-trained Bert model, even though the corresponding plaintext data is also in Chinese. Therefore, the desensitization data can confused pre-trained BERT model easily **at the beginning of training** and then slow down the convergence. The main reason caused this phenomenon is the semantic inconsistency of tokens in our digital vocabulary and the embedding assigned to it in the embedding layer of BERT. For example, if the plaintext (semantic meaning) of "22" in the ciphertext is word "you" in English and the index of "22" in our digital vocabulary is 0, we hope that the embedding assigned to it can reflect the semantic meaning of "you". However, in the real case, the embedding assigned to "22" will be the embedding that reflect the semantic meaning of 0^{th} token in BERT's vocabulary, which is padding character. To handle the semantic inconsistency, we want to find a way for **aligning the desensitization data to the embeddings** that can reflect their semantic meaning.

Frequency Analysis Solution: In cryptography, frequency analysis is a traditional and classical cryptanalysis method for decoding ciphertext such as Caesar Cipher. Frequency analysis means decoding the ciphertext by aligning the characters appeared in ciphertext to the alphabets appeared in plaintext corpus according to their frequencies. For example, if "k" in ciphertext has the similar frequency with "a" in plaintext corpus, then the plaintext of "k" is likely to be "a", especially when the frequencies are high. Inspired by this decryption idea, we reuse the pre-trained embeddings we firstly reorder each row of embedding layer according to the frequency order (from high to low) of their corresponding plaintext tokens. Then we reorder our digital vocabulary in the same way. By doing so, the token in our vocabulary that is with highest frequency will be assigned with the pre-trained static embedding of the plaintext token that is also with highest frequency, which can reflect its semantic meaning better. As shown in section 4.3, this strategy help us gain 15% AUC score at the first epoch, which means the convergence is greatly accelerated. Meanwhile, as a naive decoding strategy, frequency analysis itself cannot decode the desensitization sentence, because it only works well on the few high frequency tokens ("you" "me" "the"), while the intermediate and low frequency words (proper nouns, numbers) usually contain more valuable information. Therefore, there are no ethical concerns.

Word2vec Solution: Except reusing and aligning the pre-trained static embeddings for Chinese, we can regarding the desensitization data as a new language and train new static embeddings that can **reflect the semantic meaning** of them. Here we adopt skip-gram (Mikolov et al., 2013) model for training embeddings. The embeddings in skip-gram is actually the weight parameters.By performing the context word prediction task on large amounts of data the weight matrix can be well trained and the embeddings can reflect the semantic meaning of token well. We firstly train the skip-gram model on all the data we have. Then we substitute the weight matrix of BERT's embedding layer to our new embeddings in the order of our digital vocabulary. By doing so, the embedding assigned to desensitization token "22" will be the embedding of "22" pre-trained in skip-gram model. Adopting this method, the AUC score is improved by about 5% percent at the first epoch.

3.4. Data Augmentation

The capacity of BERT has shown to be further improved by adopting data augmentation (Raffel et al., 2020). Here we adopted two simple data augmentation strategies for two different purposes respectively. Firstly, during training we randomly exchanged the positions of two texts with probability of 0.5 that is ([CLS] text 1 [SEP] text 2[SEP]) and ([CLS] text 2 [SEP] text 1 [SEP]). On account of the existence of position embedding and segmentation embedding, the final embedding, which is the sum of position embedding, segmentation embedding and token embedding, for each tokens are different in these two formats, and thus the different input order may confuse the model. By doing so, our model could be more robust to the irrelevant variations of the order of input sentences pairs. Secondly, noticing the unbalanced distribution of binary labels(positive samples proportion is 36.4%), we expended the positive samples through "label propagation". To be more specific, if we have three sentences: A,B and C, where (A,B) and (B,C) are positive pairs which are labeled to be semantic consistent, then we could produce new positive sample (A,C). According to this principle, we implemented the union-find algorithm to explore all possible positive pairs, which is to consider each sentence as a node in the data space and merge them into different disjoint sets based on the existing edges provided by origin positive samples. Then we generate new edges in each connected graph and produce new positive samples. Due to the fact that our target data set is labeled under the mobile voice assistant scenario, which means many queries are paired with the same standard question, we were able to expend the scale of positive sample by nearly a half, which helped the model predict positive pairs better and improve the overall AUC score consequently.

3.5. Self-Ensemble and Self-Distillation

Self-Ensemble: Ensemble is a classical method for improving the generalization ability of a model and the ensemble BERT usually give better generalization performance than a single BERT model due to the randomness of the stochastic gradient decent (SGD). Therefore, here we want to exploit the benefit of ensemble BERT. However, the ensemble BERT is computational expensive and memory consumed, for we have to fine-tuning multiple BERT model and keep them for prediction, which we cannot afford. In this case, the self-ensemble method (Xu et al., 2020) fit our requirements well. The self-ensemble method treats the intermediate training states of BERT at different epochs as base models. By doing so, we can obtain all the base models with a single training process, which is huge save of computational resources. Meanwhile, for decreasing the model complexity further, a weight averaging strategy is applied for combining all the base models into a single model:

$$\mathrm{BERT}_{\mathrm{SE}}(x;\bar{\theta}) = \mathrm{BERT}\left(x; \frac{1}{T}\sum_{\tau=1}^{T}\theta_t\right) \tag{2}$$

By adopting this method, we can gain benefit from ensemble Bert, with no additional computation consume and the lowest memory cost.

Self-Distillation: Even though the ensemble model can improve the generalization ability, the training processes is never affected by ensemble model, which means the training in following time steps are not on the basis of the ensemble model that obtained in previous time steps. Therefore, we applied self-distillation-average (SDA) method (Xu et al., 2020) to help training process gain benefit from ensemble model. Self-distillation method maintain two models during training, one is the student model which is the Bert model at current time step, and another one is the teacher model which is the ensemble model obtained by average student models at previous time steps. In the training, the student model will not only learn from ground truth labels (hard label) but also will learn from the prediction of teacher model (soft label). By doing so, the student model will be more robust and accurate, and lead to a better teacher model further. The learning process of student model is shown below, where CE is the cross-entropy loss, MSE is the mean square error and λ is a task-sensitive parameter)

$$\begin{aligned}\mathcal{L}_\theta(x,y) &= \mathrm{CE}(\mathrm{BERT}(x,\theta),y)\\ &\quad + \lambda\,\mathrm{MSE}(\mathrm{BERT}(x,\theta),\mathrm{BERT}(x,\bar{\theta}))\end{aligned} \tag{3}$$

Here we will give the pseudocode of our implementation, where k is the number of student models that will be averaged to produce teacher model, max_epoch is the number of epochs we trained the model and λ is the weight of the MSE loss which measure the difference of student's prediction and teacher's prediction. We will leave the parameter selection into the experiment selection 4.2.

Algorithm 1 Self-Ensemble & Self-Distillation

```
Input: $k$, $max\_epoch$, $\lambda$
Initialization: teacher, student=Bert(nn.Module),
model_list=[], flag=false
for $epoch = 0$ to $max\_epoch$ do
  if $flag$ then
    student=train(data, CE+$\lambda$MSE)
  else
    student=train(data, CE)
  end if
  if $len(model\_list) < k$ then
    model_list.append(student.weights)
  else
    model_list.pop(0)
    model_list.append(student.weights)
    teacher.weights=average(weights in model_list)
  end if
end for
```

4. Experiments and Analysises

In this section, we will demonstrate the effectiveness of the methods we adopted and analyze the phenomena that appeared in experiments in detail. Note that for saving computational resources we only train the model longer when necessary, that is if 10 epochs are enough for checking the effectiveness of the method, we will not train it longer.

4.1. Baseline Selection: BiMPM or BERT?

As mentioned in section 3.1, both BiMPM and BERT are worth trying for the good results achieved on tasks (LCQMC & QQP) that share great similarity with ours. Here we conducted a control experiment for baseline selection. For BERT 128 max_length we selected has already covered (99% of inputs length $len(text1) + len(text2) + 3$) and larger max_length will lead to more memory cost. Meanwhile, considering the limitation of computational and memory, we selected $\mathrm{BERT}_{\mathrm{Base}}$ as the candidate, which is pre-trained on a large Chinese corpus and and has 12 hidden layers, 768 hidden dimensions, 12-heads attention, and 110M parameters. For Bert, we selected batch size and learning rate as 32 and $2e-5$ respectively, as suggested in the original paper and keep all other hyperparameters as default. The hyper-parameters for BiMPM is shown in Table 2.

Table 2. The hyper-parameters value setting of BiMPM

learning rate	batch size	hidden_dim	dropout
0.001	32	100	0.3

As shown in Figure 1, the Bert model greatly outperformed the BiMPM. As mentioned in section 3.1 the superior performance of Bert mainly come from two aspects: 1. the transformer block (contain attention mechanism) enables much complete and fine-grained interaction of tokens' embeddings 2. Bert is a deep bidirectional model but BiMPM is a superficial bidirectional model. On account of the much higher AUC score achieved by Bert, we selected BERT$_{\text{Base}}$ as our base model.

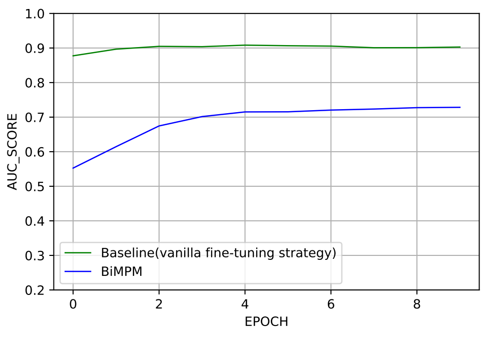

Figure 1. Comparison of BiMPM and BERT$_{\text{Base}}$

4.2. Enhanced Training Strategy

In this subsection we are going to explore the effectiveness of enhanced training strategy in practice, compared with the vanilla fine-tuning strategy adopted in baseline. For doing control experiment we use the BERT$_{\text{Base}}$ with default setting (same with section 4.1) for both two experiments and only changed the training strategy. The AUC curves, which are evaluated on development set, for two strategies are shown in Figure 2.

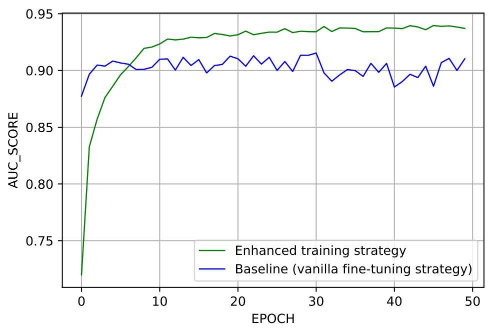

Figure 2. Comparison of vanilla and enhanced training strategy

As shown in Figure 2, even though the vanilla fine-tuning strategy achieved a much higher AUC score at the first epoch, the training in following steps is unstable with large oscillation and the final AUC score is less improved compared with the first epoch. The reason is that, our similarity classification task is relatively simple to the powerful BERT model, thus a large amounts of parameters cannot be adjusted appropriately using this simple classification task. In another word, we do not fully exploit the capacity of BERT and a lots of parameters are not well-trained. Consequently, the performance is less improved in the later training stages and the training is unstable due to the parameters that can not be trained appropriately. Different with vanilla strategy, the enhanced strategy lead to a much stable training process and beat the highest performance (0.9090 AUC) of vanilla strategy within 10 epochs and ultimately outperform the vanilla strategy by about 3 AUC scores (0.9396 AUC). The benefit mainly come from the added MLM auxiliary task, which increased the complexity of task and can help the BERT model learn better embeddings in nature, which is beneficial to generalization even for different tasks on same language (our desensitization data) (Devlin et al., 2018).However, as shown in Figure 2, the enhanced strategy shows a great performance degradation at the beginning. This degradation mainly caused by that the model does not perform well on MLM task, and thus MLM task largely distracts the attention of the model at the beginning, and as the training progresses the distraction is increasingly smaller.

4.3. Embedding Alignment:

The simple similarity classification task obscured an important issue that the semantic mismatching between tokens and the embedding assigned to them in the embedding layer, due to the mismatch of the task complexity and model's power. This issue was exposed in much complex MLM task. The semantic mismatching jeopardised the model's performance on MLM task and further greatly distracted the model's attention on similarity classification task. As stated in section 3.3 we proposed two solutions for mitigating the semantic mismatching happened in embedding layer. In this section we are going to explore the practical effectiveness of frequency alignment solution and word2vec solution and compare them with each other. All the experiments used the BERT$_{\text{Base}}$ model with the enhanced training strategy illustrated in 4.2 and 3.2 and both trained for 10 epochs which are enough for demonstrating the effectiveness. The model keeps the same setting shown in the section 4.1.

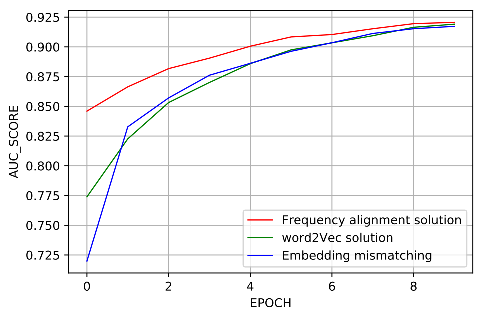

Figure 3. The effectiveness of embedding alignment solutions

As shown in Figure 2, both word2vec and frequency alignment solutions can promote the convergence of the model. The frequency alignment solution brought us about 0.15 additional AUC scores at the first epoch compared with random initialized embeddings (the misaligned embedding could be seen as a random initialization for tokens in our vocabulary), while word2vec brought us about 0.065 AUC score at the first epoch. Therefore, frequency alignment is a better solution for the semantic mismatching happened in the embedding layer. The reason of why frequency alignment can outperform the word2vec solution is that the pre-trained embedding has been trained on million data, and thus can reflect the semantic meaning of the corresponding token more accurately. By using frequency alignment, we can partly reuse (works well for high frequency words) the pretrained embeddings that with high quality. However, our embedding produced by skip-gram model is less qualified but still better than the random initialization. The reason is that our data for training the skip-gram is limited (25000 texts), while the skip-gram model in original paper was trained on 1.6 billion data (Mikolov et al., 2013). Besides, as our expected, the two strategies and the random initialization version will converge to the similar point at the 10^{th} epoch, because the initialization can only promote the convergence speed and its effect will gradually diminished as the progress of training, unless the parameter cannot be trained appropriately (like our simple task case).

4.4. Data Augmentation

Here we are going to test the effect of two data augmentation methods proposed in section 3.3, both of two methods are implemented on the basis of enhanced training strategy and frequency alignment solution that is used for embedding alignment. Here we still use the $\text{BERT}_{\text{Base}}$ with the same setting with the previous experiments (positive sample augmentation strategy will take two times training time for the larger amount of training data) . Due to the computational resources limit, we only trained the model for 10 epochs for both data augmentation strategies, which are relatively sufficient for demonstrate the usefulness of these two strategies. Note that, for the random position exchange strategy we kept the amounts of data unchanged for each epoch, by exchanging the position of text 1 and text 2 with probability of 0.5. In this way, the benefit of robustness could be demonstrated better. The performance of adopting random position exchange strategy is shown in Figure 4 (a).

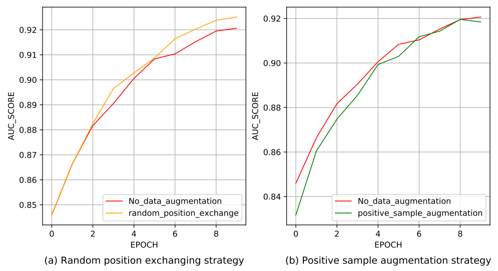

Figure 4. The performances of two data augmentation strategies

As shown in Figure 4 (a), by adopting random position exchanging strategy, our model's generalization performance has been boosted further (about 0.05 0.5% at 10^{th} epoch), the reason is that our model become more robust to the changes of irrelevant features.

The relatively worse experimental results of positive sample augmentation strategy (shown in figure 4 (b)) is out of our expectation, for the model is supposed to gain benefit from the more balanced training set or at least gain benefit from the additional training data. After analysis, we think this is because augmented positive sample is not as good as we thought. We expended the positive sample according to the principle: (A, B, 1), (B, C, 1) − > (A, C, 1). However, we did not add a limit on the number of times of label passes, which is impractical because the semantic error will keep increasing along with the label propagation process. For example: (A, B, 1), (B, C, 1), (C, D, 1) (D, E, 1) may lead to (A, E, 0) in the real case. Due to the time and resources limit we would like to leave the improvement methods to the further work.

4.5. Self-Ensemble & Self-Distillation

In this section, we aim to explore the practical effectiveness of self-ensemble and self-distillation method. On account of we just trained the data augmentation enhanced model for 10 epochs to check the usefulness of the data augmentation methods, here we excluded the data augmentation methods and focus on the effusiveness of self-ensemble and self-distillation method only. As described in section we have two hyperparameters k and λ, here we set $k = 2$ for it give the highest performance on the text classification tasks shown in original paper (Xu et al., 2020) an $\lambda = 1$ for it give the highest performance on almost all tasks stated in origin

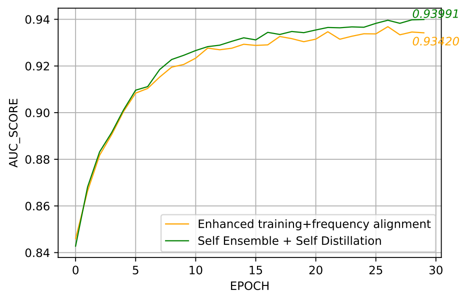

Figure 5. The performances of self-ensemble and self-distillation method

As shown in Figure 5, the ensemble and distillation applied model (enhanced model) achieved a higher AUC score for almost all epochs and ultimately outperform the original model by about 0.5 AUC scores. The benefit mainly come from two aspects. Firstly, the prediction is made by the teacher model and teacher model is the combination of two well-trained student model, which make it more robust and accurate. Besides, by using self-distillation a virtuous circle is created in the training process that is the more robust and accurate teacher will lead to a better student model, and multiple better student models will lead to a better teacher model further. However, we noticed the enhanced model did not outperform the original model at the first 5 epochs. This is because at the beginning of training the student models were not well trained. Consequently as the combination of these student models, the teacher model will not give obvious performance improvement. However, as the training progresses, a better trained student model will become the start point of the virtuous circle. Need to mention that, we only selected the hyperparameter according to the prior knowledge and thus this method can potentially bring us more benefits by doing hyperparameter searching appropriately, because k is a task sensitive hyperparameter. Due to the resources limit we will leave the hyperparameter searching for future work.

4.6. Final Model and Online Evaluation

	Offline AUC	Online AUC
Vanilla Baseline	0.9054	0.8015
Enhanced Baseline	0.9392	0.8417
Final Model	**0.9548**	**0.8834**

Table 3. Offline and online AUC score of Vanilla Baseline, Enhanced Baseline, and Final Model

In this section, we integrated all techniques which has been proved to be effective above: Enhanced Training Strategy, Embedding Alignment, Data Augmentation, Self-Ensemble, and Self-Distillation, which jointly compose our final model and train it for 50 epochs. We then evaluate three models: Vanilla Baseline, Enhanced Baseline and Final Model on both offline and online data set. It is worth noticing that the offline data set is actually the validation set split from the whole data set(2W out of 10W samples), hence it generally holds the same data distribution as the training set. While the Online set is the test set whose ground truth labels are only accessible to the OPPO company and holds significantly different data distribution, which is a proper way to evaluate the generalization performance of the models in a real production scenario.

As shown in the table above, our final model achieves the highest AUC score on both offline and online evaluation. Furthermore, the extent of the degradation of the models on online data distribution demonstrates the true generalization performance in the real industry application. Our final model holds the smallest offline-online AUC gap which illustrates that the integration of these techniques applied in the fine-tuning phase successfully improves the generalization performance of the model in NLSM task.

5. Related work

There are mainly two kinds of models in NLSM task. One is representation-based model, which is to generate the favourable embedding for two sentences respectively and measure the similarity(i.e cosine similarity) of them in embedding space: DSSM(Huang et al., 2013), ARC-I(Hu et al., 2015). Another kind of models is interaction-based, which is to take two sentences as the input and use a interaction function to predict the similarity: ARC-II(Hu et al., 2015), MIX(Chen et al., 2018), BiMPM(Wang et al., 2017), BERT(Devlin et al., 2018). In recent years, interaction-based models have shown more dominating performance as they are able to be trained on large unlabeled corpus and then fine-tuned on the task.

Besides what has been introduced in section 1, there are some researches focus in the application of pre-trained BERT in NLSM task. FastBERT(Liu et al., 2020) proposed a self-distillation method which is different from the one used in section 3.5. It add student classifiers at each layer and use the fine-tuned backbone BERT as the teacher classifier. In the inference phase, if any student classifier yields low-uncertainty prediction during the forward propagation, it directly use it as the output and stop the following computation which makes the model sample-wise adaptive and much faster. Sentence-BERT(Reimers & Gurevych, 2019) utilizes the siamese network for modifying BERT specific for NLSM, which is to use BERT to generate embeddings for two sentences repectively and then simply calculate their cosine similarity. These two methods mainly focus on the improvement of the inference speed, while ours work pays more attention on fine-tuning BERT into new data set in unseen language.

6. Conclusions

In this work, we explored fine-tuning phase of BERT for the Natural language sentence matching (NLSM) task. Firstly, we show the strength of the $\text{BERT}_{\text{Base}}$ compared with BiMPM. Therefore, combined with the vanilla fine-tuning strategty, $\text{BERT}_{\text{Base}}$ becomes our baseline. Then we applied the enhanced training strategy and showed that with the help of MLM auxiliary task, the model beat the baseline within 10 epochs and achieved a better AUC score ultimately. Based on this enhanced baseline, we compared two embedding alignment methods and find that the frequency alignment is the most effective one regarding acceleration of the training process. Furthermore, our experiments in section 4.4 demonstrates that random position exchanging strategy can let the model become more robust, while the positive sample augmentation strategy method does not work as well as our expectation, which indicates more strict constraints is needed during label propagation to generate positive samples with higher quality. In addition, self-ensemble and self-distillation bring improvement to the validation performance. Finally, we integrated all techniques which has been shown to be useful into the final model and evaluate it on both offline and online data set. Compared with baseline models, it achieves the highest offline AUC score 0.9548 and the highest online AUC score 0.8834, meanwhile it holds the smallest offline-online AUC gap. So it is reasonable to conclude that our final model obtained the best classification and generalization performance under the real production scenario for semantic matching.

As for the future work, there remains several directions worth further exploration. Besides the searching of hyperparameter k(number of student models in self-distillation) and producing higher quality postive sample, . Moreover, the adversarial training technique such as FGM(Miyato et al., 2017) could help the model gain more robust weights and better generalization performance as the data distribution of online data usually varies a lot compared with the offline data.

References

Chen, Haolan, Han, Fred X, Niu, Di, Liu, Dong, Lai, Kunfeng, Wu, Chenglin, and Xu, Yu. Mix: Multi-channel information crossing for text matching. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 110–119, 2018.

Cui, Yiming, Che, Wanxiang, Liu, Ting, Qin, Bing, Yang, Ziqing, Wang, Shijin, and Hu, Guoping. Pre-training with whole word masking for chinese bert. *arXiv preprint arXiv:1906.08101*, 2019.

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

GAIIC. Global artificial intelligence innovation contest track three, 2021. URL https://tianchi.aliyun.com/competition/entrance/531851/information.

Hu, Baotian, Lu, Zhengdong, Li, Hang, and Chen, Qingcai. Convolutional neural network architectures for matching natural language sentences. *Advances in neural information processing systems*, 3, 2015.

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

Iyer, Shankar, Dandekar, Nikhil, and Csernai, Kornel. First quora dataset release: Question pairs. 2017.

Lample, Guillaume and Conneau, Alexis. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*, 2019.

Liu, Weijie, Zhou, Peng, Zhao, Zhe, Wang, Zhiruo, Deng, Haotang, and Ju, Qi. Fastbert: a self-distilling bert with adaptive inference time. *arXiv preprint arXiv:2004.02178*, 2020.

Liu, Xin, Chen, Qingcai, Deng, Chong, Zeng, Huajun, Chen, Jing, Li, Dongfang, and Tang, Buzhou. LCQMC:a large-scale Chinese question matching corpus. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1952–1962, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1166.

Liu, Yinhan, Ott, Myle, Goyal, Naman, Du, Jingfei, Joshi, Mandar, Chen, Danqi, Levy, Omer, Lewis, Mike, Zettlemoyer, Luke, and Stoyanov, Veselin. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Miyato, Takeru, Dai, Andrew M, and Goodfellow, Ian. Adversarial training methods for semi-supervised text classification. *ICLR*, 2017.

Peters, Matthew E, Neumann, Mark, Iyyer, Mohit, Gardner, Matt, Clark, Christopher, Lee, Kenton, and Zettlemoyer, Luke. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

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

Raffel, Colin, Shazeer, Noam, Roberts, Adam, Lee, Katherine, Narang, Sharan, Matena, Michael, Zhou, Yanqi, Li, Wei, and Liu, Peter J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

Reimers, Nils and Gurevych, Iryna. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

Robertson, Stephen and Zaragoza, Hugo. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc, 2009.

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

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

Xu, Yige, Qiu, Xipeng, Zhou, Ligao, and Huang, Xuanjing. Improving bert fine-tuning via self-ensemble and self-distillation. *arXiv preprint arXiv:2002.10345*, 2020.

Yang, An, Wang, Quan, Liu, Jing, Liu, Kai, Lyu, Yajuan, Wu, Hua, She, Qiaoqiao, and Li, Sujian. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2346–2357, 2019a.

Yang, Zhilin, Dai, Zihang, Yang, Yiming, Carbonell, Jaime, Salakhutdinov, Ruslan, and Le, Quoc V. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019b.