Using Deep Mixture-of-Experts to Detect Word Meaning Shift for TempoWiC

Ze Chen, Kangxu Wang, Zijian Cai, Jiewen Zheng Jiarong He, Max Gao Jason Zhang
Interactive Entertainment Group of Netease Inc., Guangzhou, China
{jackchen,wangkangxu,caizijian01,zhengjiewen,gzhejiarong,jgao,
fyzhang}@corp.netease.com

Abstract

This paper mainly describes the *dma* submission to the TempoWiC task, which achieves a macro-F1 score of 77.05% and attains the first place in this task. We first explore the impact of different pre-trained language models. Then we adopt data cleaning, data augmentation, and adversarial training strategies to enhance the model generalization and robustness. For further improvement, we integrate POS information and word semantic representation using a Mixture-of-Experts (MoE) approach. The experimental results show that MoE can overcome the feature overuse issue and combine the context, POS, and word semantic features well. Additionally, we use a model ensemble method for the final prediction, which has been proven effective by many research works.

1 Introduction

Lexical Semantic Change (LSC) Detection has drawn increasing attention in the past years(Liu et al., 2021; Laicher et al., 2021). Existed research works (Liu et al., 2021) have shown that contextual word embeddings such as those produced by BERT (Devlin et al., 2018) have great advantages over non-contextual embeddings for inferring semantic shift when there is limited data. Meanwhile, many datasets are released to accelerate research in this direction. Pilehvar and Camacho-Collados (2018) proposed Word-in-Context (WiC) dataset as an benchmark for generic evaluation of context-sensitive representations. Raganato et al. (2020) extended WiC to XL-WiC dataset with multilingual extensions. In contrast to these, TempoWiC (Loureiro et al., 2022b) is crucially designed around the time-sensitive meaning shift and instances of word usage tied to Twitter trending topics. Our main work is to build a system that can detect semantic changes of target words in tweet pairs during different time periods for TempoWiC. It is framed as a binary classification task that addresses whether two instances of a target word have the same meaning. And pre-trained language models are adopted to produce contextual embeddings.

2 Background

2.1 Task Description

TempoWiC (Loureiro et al., 2022b) is a new benchmark especially aimed at detecting a meaning shift in social media. Given a pair of sentences and a target word, the task is framed as a simple binary classification problem in deciding whether the meaning corresponding to the first target word in context is the same as the second one or not.

The dataset of TempoWiC consists of 3297 annotated instances, which are divided into train/dev/test sets of size 1,428/396/1,473 instances, respectively. The target words involved in this task do not overlap between sets. For each sample, tweet pairs containing the target word were collected from the Twitter API at different time periods. The prior date is exactly one year before the peak date to avoid seasonal confound factors. The label True indicates that the word has the same meaning in the two tweets, while the label False indicates that the meaning is different.

2.2 Pre-trained Language Models

Recently, pre-trained language models (LM) have achieved remarkable achievement on natural language processing tasks, becoming one of the most effective methods for engineers and scholars. Transformers-based Pre-trained language models such as BERT(Devlin et al., 2018), RoBERTa(Liu et al., 2019), DeBERTa(He et al., 2020), DeBERTaV3(He et al., 2021) is designed to pre-trained deep representation from unlabeled text, which can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

Proceedings of the The First Workshop on Ever Evolving NLP (EvoNLP), pages 7 - 11

By training language models using Twitter corpora from different time periods, (Loureiro et al., 2022a) showed that language undergoes semantic transformations over time, proving that training a language model with outdated corpora leads to a decline in performance.

2.3 Mixture-of-Experts

MoE (Arnaud et al., 2019) is an approach for conditionally computing a representation. Given several expert inputs, the output of MoE is a weighted combination of the experts. Recently, MoE achieves significant improvements on several natural language processing tasks, such as named entity recognition (Meng et al., 2021), recommendation(Zhu et al., 2020) and machine translation(Shazeer et al., 2017).

3 System Overview

In this section, we first present the framework details for the models adopted in our work. Then we introduce several strategies for improving the models' robustness. Finally, we talk about the design of the model ensemble method.

3.1 Models

Our model framework can be divided into three layers:*encoding*, *matching* and *prediction*. The *encoding* layer is meant for sequence modeling to capture contextual semantic representation. The *matching* layer focuses on finding out the interrelation and differences between the target words in two different tweets. And the *prediction* layer is implemented as a classifier that decides whether the meaning of the target word is the same or not.

A. Base Model Figure 1 shows the details of our base model. Two tweets are concatenated together and fed into a pre-trained LM, and the contextual embeddings(e.g. E_1, E_2 [1]) corresponding to the target word on each tweet of the pair can be achieved. Then E_1 and E_2 are processed by the *matching* layer to find the difference in these two tweets. The procedure can be summarized as follows:

$$E_{match} = [E_1; E_2; E_1 - E_2; E_1 * E_2; E_{CLS}]$$

$$y_o = softmax(MLP(E_{match}))$$

$$loss = CrossEntropy(y_o, y_{true})$$

where E_{CLS} is the embedding of the first token, E_{match} is the output of the *matching* layer, MLP is a multi-layer perceptron, y_{true} is the gold label and y_o is the output by the base model. $E_1 * E_2$ means the Hadamard product of these two vectors, and $E_1 - E_2$ represents the elementwise subtraction.

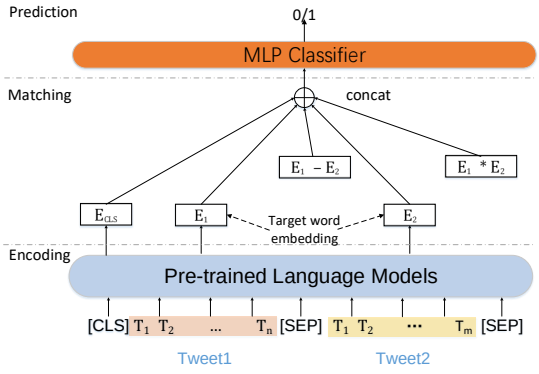

Figure 1: Base Model Architecture

B. MoE Models Figure 2 gives a glimpse of our MoE-based model architecture. We extend the base model with two separate BiLSTM to integrate the POS information and the word semantic representation. For a pair of tweets, we first extract the contextual embeddings for the target word from pre-trained LM, and then we use two separate BiLSTM to get POS encoding and word semantic encoding. At last, an MoE module is adopted to merge these three encodings for the target word. The generated embeddings(e.g. $E_1^{'}$, $E_2^{'}$) for the target word are then processed by the *matching* layer and *prediction* layer as described above. Here we denote the POS encoding for the target word in the pair of tweets as E_1^P, E_2^P, and denote the GloVe-initialized word semantic encoding as E_1^G, E_2^G respectively.

The details of an MoE module for this task are given in Figure 3, which consists of a gating network and three experts. The procedure can be summarized as follows:

$$w_C, w_P, w_G = Gate(E_1, E_1^P, E_1^G)$$

$$E_1^{'} = w_C * E_1 + w_P * E_1^P + w_G * E_1^G$$

where w_C, w_P, w_G are the weights for contextual expert, POS expert, and word semantic expert respectively, $Gate$ stands for the gating network, and

[1] We experimented with different target word representations: the first token in the word span, the mean value of all tokens in the span, the concatenation of the first token and last token in the span. And we found that adopting the concatenation of the first token and last token in the span can perform better than others. Please refer to Appendix A for more details.

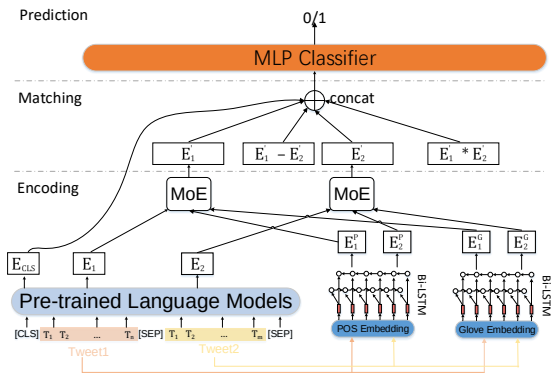

Figure 2: MoE-based Model Architecture

$E_1^{'}$ is the output of MoE module for the target word in the first tweet. We can get $E_2^{'}$ for the target word in the second tweet by the same approach. And two gating networks are implemented here.

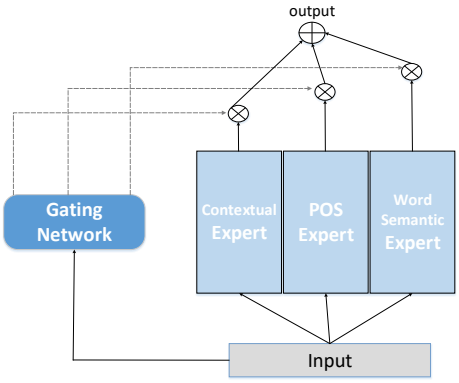

Figure 3: Illustration of an MoE module

- **Separate Gating Network(S-Gate):** The weight for each expert is calculated separately. We define a task-specific vector V_t, the weight for expert i can be calculated as: $w_i = \sigma(\theta[V_t, E^i])$, where θ are trainable parameters, $[,]$ is the concatenation and σ is the Sigmoid activation, E^i is the encoding of i-th expert.
- **Joint Gating Network(J-Gate):** The weights for all experts are calculated together. We define the weight vector for all experts as W, which is a three-dimension vector and can be calculated as: $W = softmax(\theta[E^1, E^2, E^3])$, where θ are trainable parameters.

3.2 Data Cleaning and Augmentation

Given that the dataset is somewhat small, and there are some flaws in the labeled data, we adopt simple cleaning and augmentation strategies. We simply remove HTML tags and emojis in tweets, and replace the symbol *@username* with a generic placeholder. Moreover, we directly remove the wrongly labeled samples of the target word position. There are many different data augmentation strategies: token shuffling, cutoff, back-translation, and so on. We just introduce the WiC dataset(Pilehvar and Camacho-Collados, 2018) for data augmentation in this paper.

3.3 Adversarial Training

Adversarial attack has been well applied in both computer vision and natural language processing to improve the model's robustness. We implement this strategy with Fast Gradient Method(Goodfellow et al., 2014), which directly uses the gradient to compute the perturbation and augments the input with this perturbation to maximizes the adversarial loss. The training procedure can be summarized as follows:

$$\min_{\theta} E_{(x,y)\sim\mathcal{D}}[\max_{\Delta x\in\Omega} L(x+\Delta x, y; \theta)]$$

where x is input, y is the gold label, $\mathcal{D}$ is the dataset, θ is the model parameters, $L(x+\Delta x, y; \theta)$ is the loss function and Δx is the perturbation.

3.4 Model Ensemble

For the final prediction, we implement a model ensemble method. In detail, we use one base model and the other two MoE models mentioned above to get the prediction scores and then average these output scores as the final result.

4 Experiments

4.1 Experimental Setup

Our implementation is based on the Transformers library by HuggingFace(Wolf et al., 2019) for the pre-trained models and corresponding tokenizers. During training, the data is processed by batches of size 8, the maximum length of each sample is set to 256, and the learning rate is set to 1e-6 with a warmup ratio over 10%. By default, we set ϵ to 1.0 in FGM and set the MLP to two layers with a hidden size of 256.

When MoE models are employed, the hidden size of BiLSTM is set to 1024, and the pre-trained Twitter GloVe word vectors [2] are used for word embedding initialization. Moreover, we use nltk

[2] https://nlp.stanford.edu/projects/glove/

toolkit [3] to extract POS tags, and the POS embeddings are randomly initialized. Our system jointly optimizes over different experts, but their model architectures differ. We adopt differential learning rates to tackle this problem. The learning rate for the transformer-based model is set to 1e-6, and the learning rate for BiLSTM is set to 1e-4.

4.2 Results and Analysis

In this section, we first present experimental results on the base model. Then we experiment with MoE models using the effective strategies validated on the base model. At last, the results of the model ensemble are reported.

We explore the impact of different pre-trained LMs adopted as the contextual encoder. Results given in Table 1 show that DeBERTa-large can perform well on this task. And TimeLMs(Loureiro et al., 2022a) can perform better than generic RoBERTa since they are implemented and adapted to the Twitter domain. Moreover, TimeLMs-2020-09 can achieve almost the best results among TimeLMs, largely because the dev dataset is distributed over this time period. From the last two rows in Table 1, we can find that data cleaning and augmentation can increase the macro-F1 score by 2.83 percentage points, and FGM training can increase this indicator by 2.57%. Additionally, the ablation study results on *matching* layer are presented in Appendix A, we can find that the first token $[CLS]$ embedding can help improve the performance of this task. The subtraction and Hadamard product operations can also help find the difference between target words in two tweets.

When we experiment with MoE models, the data cleaning and augmentation, and FGM training are adopted by default. And the pre-trained DeBERTa-large is used for the contextual encoder. Table 2 shows the performance of different MoE models. We can find that when integrating POS information and word semantic representation by using an MoE architecture, the performance can improve a lot. More specifically, the MoE model with S-Gate and J-Gate can achieve macro-F1 scores of 79.25% and 79.19% respectively, both of which increase the base by more than 2%. For further analysis, ablation studies are done here. We experiment with POS information and GloVe separately and find that using an MoE model to integrate POS information can improve the performance by 1%, while using an MoE model to combine word semantic representation can increase the macro-F1 score by about 2%.

[3] https://www.nltk.org/

Model	Accuracy	macro-F1
TimeLMs-2019-12	67.17%	63.34%
TimeLMs-2020-03	68.18%	65.20%
TimeLMs-2020-09	68.43%	65.42%
TimeLMs-2020-12	68.18%	65.20%
TimeLMs-2021-03	66.67%	63.88%
TimeLMs-2022-03	68.18%	65.12%
RoBERTa-base	61.62%	60.30%
DeBERTa-base	69.90%	65.60%
DeBERTa-large	71.72%	71.68%
+ Data Aug	74.63%	74.51%
+ Data Aug + FGM(Base)	**77.53%**	**77.08%**

Table 1: Results of base model on dev dataset

Model	Accuracy	macro-F1
Base	77.53%	77.08%
S-Gate + POS + GloVe	**79.29%**	**79.25%**
S-Gate + POS	78.26%	77.20%
S-Gate + GloVe	78.19%	77.18%
J-Gate + POS + GloVe	**79.29%**	79.19%
J-Gate + POS	78.62%	78.31%
J-Gate + GloVe	77.55%	77.12%

Table 2: Results of MoE-based models on dev dataset

Table 3 gives the results of our model ensemble method. By averaging the prediction scores of one base model and the other two MoE models(S-Gate + POS + GloVe, J-Gate + POS + GloVe), the macro-F1 score can increase by more than 1% on the dev dataset. And our model ensemble method achieves a macro-F1 score of 77.05% on the test dataset, which attains the first place in this task.

Dataset	Accuracy	macro-F1
Dev	80.81%	80.5%
Test	**78.34%**	**77.05%**

Table 3: Ensemble results on both Dev and Test dataset

5 Conclusion

In this work, we provide an overview of the combined approach to detect the meaning shift in social media. We investigate the impact of adopting different pre-trained LMs, finding that DeBERTa performs best for this task. Experimental results show that strategies such as data augmentation and adversarial training can enhance the model's robustness. In particular, incorporating POS information and word-level semantic representation with MoE models can significantly improve performance. For future work, we will investigate how to incorporate different TimeLMs with MoE models for this task.

References

Estephe Arnaud, Arnaud Dapogny, and Kevin Bailly. 2019. Tree-gated deep mixture-of-experts for pose-robust face alignment.

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

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Severin Laicher, Sinan Kurtyigit, Dominik Schlechtweg, Jonas Kuhn, and Sabine Schulte im Walde. 2021. Explaining and improving bert performance on lexical semantic change detection. *arXiv preprint arXiv:2103.07259*.

Yang Liu, Alan Medlar, and Dorota Glowacka. 2021. Statistically significant detection of semantic shifts using contextual word embeddings. In *Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems*, pages 104–113, Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-collados. 2022a. TimeLMs: Diachronic language models from Twitter. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 251–260, Dublin, Ireland. Association for Computational Linguistics.

Daniel Loureiro, Aminette D'Souza, Areej Nasser Muhajab, Isabella A. White, Gabriel Wong, Luis Espinosa Anke, Leonardo Neves, Francesco Barbieri, and Jose Camacho-Collados. 2022b. Tempowic: An evaluation benchmark for detecting meaning shift in social media.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gemnet: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2018. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*.

Alessandro Raganato, Tommaso Pasini, Jose Camacho-Collados, and Mohammad Taher Pilehvar. 2020. XL-WiC: A multilingual benchmark for evaluating semantic contextualization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7193–7206, Online. Association for Computational Linguistics.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Ziwei Zhu, Shahin Sefati, Parsa Saadatpanah, and James Caverlee. 2020. Recommendation for new users and new items via randomized training and mixture-of-experts transformation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1121–1130.

A Additional Experiments on the base model

In this part, we present several additional experimental results on the base model.

We tried different target word representation methods for contextual embedding. The results on dev dataset are listed in Table 4.

Target word	Accuracy	macro-F1
$First$	75.6%	74.49%
$Mean$	74.94%	74.46%
$First$ + $Last$	**77.53%**	**77.08%**

Table 4: Results of different target word representation methods

To make further analysis, we conducted ablation studies to investigate the contribution of different components of *matching* layer. Results are shown in Table 5.

***Matching* layer**	**Accuracy**	**macro-F1**
E_1 + E_2	75.92%	74.74%
+ E_{CLS}	77.15%	76.45%
+ E_{CLS} +[E_1-E_2] + [E_1*E_2]	**77.53%**	**77.08%**

Table 5: Results of different components of *matching* layer