Language Model Pre-training on True Negatives

Zhuosheng Zhang[1,2], **Hai Zhao**[1,2,*], **Masao Utiyama**[3], **Eiichiro Sumita**[3]

[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University, Shanghai, China
[3]National Institute of Information and Communications Technology (NICT), Kyoto, Japan
zhangzs@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn, {mutiyama,eiichiro.sumita}@nict.go.jp

Abstract

Discriminative pre-trained language models (PLMs) learn to predict original texts from intentionally corrupted ones. Taking the former text as positive and the latter as negative samples, the PLM can be trained effectively for contextualized representation. However, the training of such a type of PLMs highly relies on the quality of the automatically constructed samples. Existing PLMs simply treat all corrupted texts as equal negative without any examination, which actually lets the resulting model inevitably suffer from the false negative issue where training is carried out on pseudo-negative data and leads to less efficiency and less robustness in the resulting PLMs. In this work, on the basis of defining the false negative issue in discriminative PLMs that has been ignored for a long time, we design enhanced pre-training methods to counteract false negative predictions and encourage pre-training language models on true negatives by correcting the harmful gradient updates subject to false negative predictions. Experimental results on GLUE and SQuAD benchmarks show that our counter-false-negative pre-training methods indeed bring about better performance together with stronger robustness.

1 Introduction

Large-scale pre-trained language (PLM) models are playing an important role in a wide variety of NLP tasks with their impressive empirical performance (Radford et al. 2018; Peters et al. 2018; Devlin et al. 2019; Yang et al. 2019; Lan et al. 2020; Clark et al. 2020). So far, there comes two major categories of PLMs with regards to the output style, the generative like GPT (Radford et al. 2018), which employ a decoder for learning to predict a full sequence, and the discriminative like BERT style of PLMs which learn to reconstruct the original uncorrupted text from the intentionally corrupted ones (Raffel et al. 2020; Lewis et al. 2020). In this work, we focus on the latter category of PLMs, typically with denoising objectives (also known as masked language modeling, MLM) (Liu et al. 2019; Joshi et al. 2020; Sun et al. 2019). In a denoising objective, a certain percentage of tokens in the input sentence are masked out, and the model should predict those corrupted tokens during the pre-training (Peters et al. 2018; Sun et al. 2019; Levine et al. 2021; Li and Zhao 2021).[1]

Although existing studies have made progress in designing effective masking strategies (Sun et al. 2019; Joshi et al. 2020; Levine et al. 2021) and auxiliary objectives (Lan et al. 2020; Wang et al. 2020) for language model pre-training, there is still a lack of attention on the quality of training data. Discriminative PLM can be regarded as a kind of auto denoising encoder on automatically corrupted texts. Thus, it is critical to ensure the auto-constructed data is true enough. Intuitively, a discriminative PLM learns to distinguish two types of samples, positive (already existing original ones) and negative (the corrupted ones from the auto constructing). Taking MLM as an example, a proportion of tokens in sentences are corrupted, e.g., replaced with mask symbols, which would affect the sentence structures, leading to the loss of semantics and increasing the uncertainty of predictions. In extreme cases, such corrupted texts may be linguistically correct. However, the current PLMs simply consider all corrupted texts as negative samples, so that the resulting PLM has to be trained on such pseudo-negative data with less efficiency and less robustness – suffers from the wasted training time on meaningless data and the trained PLM may be vulnerable to adversarial attacks like diversity distraction and synonym substitution (Wang et al. 2021).

For each training instance, MLM only calculates label-wise matching between the prediction and the gold tokens in the training process, thus inevitably suffering from the issue of false negatives where the prediction is meaningful but regarded as wrong cases, as examples shown in Table 1. We obverse that such cases appear in more than 7% of the training examples (more details in Section 2). The issue is also observed in sequence generation tasks, which is tied to the standard training criterion of maximum likelihood estimation (MLE) that treats all incorrect predictions as being equally incorrect (Wieting et al. 2019; Li et al. 2020).

*Corresponding author. This paper was partially supported by Key Projects of National Natural Science Foundation of China (U1836222 and 61733011).

[1]There are different classification standards for PLMs, i.e., output style and model architecture. For simplicity, our taxonomy follows Wang, Liu, and Zhang (2021), which is based on the output style. Note that PLMs can also be classified into three types based on the model architecture: encoder-only, decoder-only and encoder-decoder.

Example	Ground-truth	Prediction	MLM	Correction
i am trying to copy [MASK] onto my ipod good	you	happy	✗	-
an adaptive immune system whose [MASK] function ...	primary	main	✗	✓

Table 1: Examples of true negative (the first line) and false negative (the second line). The standard MLM will treat all the predictions as incorrect ones. However, the last false negative predictions can be corrected to ensure more accurate pre-training. More examples are in Figure 3.

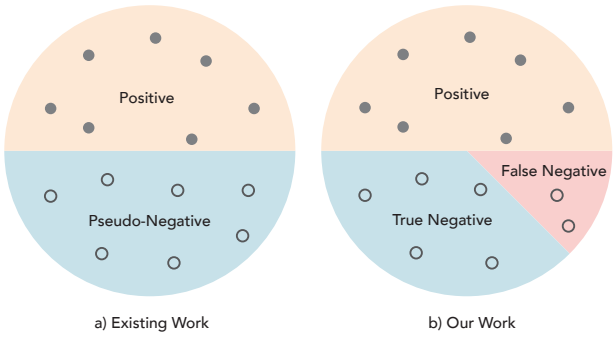

Figure 1: Overview of our study. Existing PLMs were trained by distinguishing positive from pseudo-negative data. In contrast, our work aims to encourage pre-training language models on true negatives by detecting and counteracting false negative predictions.

Instead of measuring negative diversity via diversity scores between the different incorrect model outputs, our method is dedicated to mediating the training process by detecting the alternative predictions as opposed to the gold one, to steer model training on true negatives, which benefits the resulting language modeling in general. The comparison with existing work is illustrated in Figure 1.

Though the false negatives may potentially hurt the pre-training in both efficiency and robustness to a great extent, it is surprising that this problem is kept out of the research scope of PLMs until this work to our best knowledge. To address the issue of misconceived false negative predictions and encourage pre-training language models on true negatives or more true negatives, we present an enhanced pre-training approach to counteract misconceived negatives. In detail, we investigate two enhanced pre-training objectives: 1) hard correction to shield the gradient propagation of the false negative samples to avoid training with false negative predictions; 2) soft regularization by minimizing the semantic distances between the prediction and the original one to smooth the rough cross-entropy. Experimental results on widely-used down-streaming benchmark tasks, including GLUE (Wang et al. 2019) and SQuAD (Rajpurkar et al. 2016), show that our approach boosts the baseline performance by a large margin, which verifies the effectiveness of our proposed methods and the importance of training on true negatives. Case studies show that our method keeps simplicity and also improves robustness.

Base Model			*Large Model*		
Check.	Iter.	Pred.	Check.	Iter.	Pred.
6.25	6.90	1.31	6.25	7.46	1.50
12.5	6.96	1.34	12.5	7.58	1.55
25.0	6.97	1.36	25.0	7.31	1.49
50.0	7.05	1.36	50.0	7.46	1.56
80.0	7.06	1.40	80.0	7.38	1.57
100.0	7.07	1.41	100.0	7.44	1.60

Table 2: Statistics (%) of the hard corrections under base and large settings on the wikitext-2-raw-v1 corpus. Checkpoint means the checkpoint saved at the specific training steps (%).

2 Preliminaries: The *False Negative* Issue

Definition Our concerned false negatives in MLM are the reasonable predictions but discriminated as wrong predictions because such predictions do not match the single gold token for each training case. For example, many tokens are reasonable but written in different forms or are synonyms of the expected gold token.

Severity For simplicity, we focus on the subset of false negatives from WordNet (Miller 1992) – the predictions which are the synonyms of the ground-truth tokens. To have an intuition about the severity of false negative predictions during pre-training, we collect the statistics from two perspectives: 1) prediction-level: the proportion of corrected predictions when they mismatch the gold labels; 2) iteration-level: the proportion of iterations (sequences) when the correction happens.[2] We use the wikitext-2-raw-v1 corpus (Merity et al. 2017) for validation. We use the pre-trained checkpoints of the BERT-base and BERT-large models described in Section 4.1 for the analysis.[3]

According to Table 2, we observe that the ratio of detected false negatives is around 6.0%-7.0% in iteration-level and 1.0%-2.0% in token-level.[4] As training goes on, the correction ratio increases, indicating that our method gradually

[2]The rationale is that training on false negatives tends to learn incorrect semantics of the whole sequence.

[3]The MLM process is the same as our experiments on BERT models in the subsequent sections.

[4]It is hard to collect the statistics of false negatives automatically. For simplicity, we only calculate the subset related to synonyms. Therefore, the issue is expected to occur more frequently than counted.

plays a more important role as the training proceeds, which supports our hypothesis.

Influence Pre-training on false negatives would possibly bring harm in terms of training efficiency, model effectiveness, and robustness against adversarial attacks (detailed discussions in Section 5). As the saying goes, "the rotten apple injures its neighbors", training on random examples would bring training bias from meaningless data, so it needs to be corrected with more data and results in more cost of resources and time. In addition, the inaccurate pre-training may affect the model robustness, as the PLM may fail to capture the similarity of tokens or sentences in different expressions.

3 Methodology

3.1 Masked LM

Masked LM (MLM) is a denoising language model technique used by BERT (Devlin et al. 2019) to take advantage of both the left and right contexts. Given a sentence $\mathbf{s} = \{w_1, w_2, \ldots, w_n\}$, where a certain proportion of tokens are randomly replaced with a special mask symbol. The input is fed into the multi-head attention layer to obtain the contextual representations, which is defined as $H = \text{FFN}(\text{MultiHead}(K, Q, V))$, where K, Q, V are packed from the input sequence representation $\mathbf{s}$. Then, the model is trained to predict the masked token based on the context.

Denote $\mathcal{Y}$ as the set of masked positions using the mask symbol, and the masked tokens are represented as $w_k, k \in \mathcal{Y}$. The objective of MLM is to maximize the following objective:

$$\mathcal{L}_{mlm}(w_k, \mathbf{s}) = \mathbb{E}\left(-\sum_{k\in\mathcal{Y}} \log p_\theta(w_k \mid \mathbf{s})\right). \quad (1)$$

3.2 Pre-training on True Negatives

A natural solution to encourage the language model pre-training on true negatives is to identify and counteract the false negative issue in language model pre-training. To this end, it is possible to correct or prune the harmful gradient update after detecting the false negative predictions. In detail, we investigate two enhanced pre-training objectives, including 1) hard correction (HC), which shields the gradient propagation of the false negative samples to avoid training with false negative predictions; 2) soft regularization (SR), which measures the distribution similarity between the predicted token and the original one, to smooth the tough cross-entropy by minimizing the semantic distances. Figure 2 illustrates our pre-training scheme.

Hard Correction The criteria of hard correction is to prune the gradient when the model suffers from confusion about whether the prediction is correct or not. For each prediction, we check if the predicted token r_k is highly related to the ground-truth token w_k based on a short lookup table $\mathcal{V}$ in which each w_k is mapped to a list of synonyms $\mathcal{V}[w_k]$. The training objective is:

$$\mathcal{L}_{hc} = \mathbb{E}\left(-\sum_{k\in\mathcal{Y}, r_k\notin\mathcal{V}[w_k]} \log p_\theta(w_k \mid \mathbf{s})\right). \quad (2)$$

In our implementation, the lookup table is built by retrieving the synonym alternatives for each word in the model vocabulary, e.g., from WordNet (Miller 1992) or Word2Vec embedding (Mikolov et al. 2013). Therefore, there will be no extra computation overhead for the construction of lookup table during training and the cost of retrieving synonyms is imperceptible. For the synonym source, we use WordNet synonyms by default (Section 5 will compare retrieving synonyms from WordNet and Word2Vec embedding). For each training iteration, if the predicted token is found in the synonym list for the gold token, then the correction is activated and the loss calculation for the k-th token will be neglected.[5] Such a prediction will be judged as correct by HC in cross-entropy — the correction can be applied by simply ignoring this prediction before feeding to the cross-entropy loss function. As a post-processing technique, the hard correction technique will not bring any false positives.

Soft Regularization The hard correction method above relies on external tools, which may affect the coverage of corrections due to the restricted size of the lookup table. In pursuit of more general usage, we are interested in finding a softer way to minimize the harm of false negatives. A natural way is to leverage semantic distance between the original and predicted tokens as regularization.

For w_k and r_k, we fetch their token representations from the model's embedding layer, denoted as e_k and e'_k, respectively. We leverage cosine similarity as the regularization based on the intuition that the semantic distance between the prediction and gold tokens should be minimized:

$$\mathcal{L}_{sr} = \frac{1}{N_m}\sum_{k=1}^{N_m}(1 - \frac{e_k \cdot e'_k}{\|e_k\| \cdot \|e'_k\|}), \quad (3)$$

where N_m is the number of masked tokens to predict.

SR is based on the hypothesis that the predicted tokens should have a semantic relationship with the gold ones in the same embedding space to some extent, which is supported by various existing studies (Bordes et al. 2013; Zhang and Zhao 2021; Chen et al. 2021; Li et al. 2020).[6] We choose to apply SR to the embedding layer because the embedding

[5] Words might be tokenized into several pieces before feeding PLMs, in which cases the correction will not be applied because those cases do not violate our criteria. We found that there are 62.51% tokens in the model vocabulary that have synonyms found in WordNet after removing all the stopwords.

[6] A possible concern of SR is that it may encourage the model to minimize the cosine similarity between unrelated tokens in early stages when the model is not well-trained. However, since MLM dominates the training, especially in the early stages (e.g., the loss from around 30 to 10 until convergence in ELECTRA-small). In contrast, the value of SR is usually 0-1. As such, SR would not harm the training in the early stages. As a regularization method, it further enhances the model performance beyond MLM when the training proceeds.

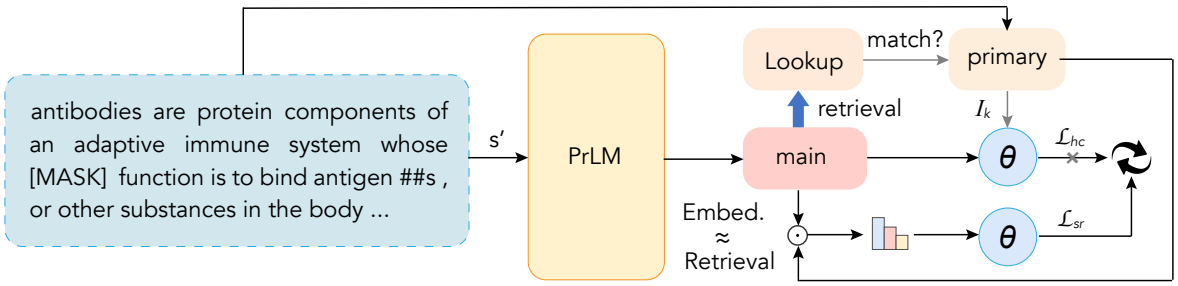

Figure 2: Illustration of our pre-training scheme.

layer is the most fundamental and stable layer as it is far from the output layer to reflect on the returned gradients during training. Optimizing the embedding layer would possibly lead to a more severe influence on the model training and help the model learn semantics between words better, as indicated by (Jiang et al. 2020). Just calculating token-wise distance neglects the context of the whole sequence. In Section 5, we will discuss the pros and cons of token-level and sentence-level SR variants.

4 Experiments

4.1 Setup

Pre-training In this part, we will introduce the model architecture, hyper-parameter setting, and corpus for pre-training our models. Our methods are applicable to general MLM-style language models. Considering the training efficiency, we employ ELECTRA small and base as our default backbone models and implement our pre-training objectives on top of them. We follow the model configurations in (Clark et al. 2020) for fair comparisons. For hyper-parameters, the batch size is 128 for the base models in our work instead of 256 as in the original setting due to limited resources. The mask ratio is 15%. We set a maximum number of tokens as 128 for small models and 512 for base models.[7] The small models are pre-trained from scratch for $1000k$ steps. To save computation, like previous studies (Dong et al. 2019), we continue training base models for $200k$ steps using the pre-trained weights as initialization. The learning rates for small and base models are 5e-4, and 5e-5, respectively. We use OpenWebText (Radford et al. 2019) to train small models, and Wikipedia and BooksCorpus (Zhu et al. 2015) for training base models following (Clark et al. 2020). The baselines and our models are trained to the same steps for a fair comparison.

To verify the generality of our methods on other PLMs, we also implemented them on $\text{BERT}_{\text{base}}$ and $\text{BERT}_{\text{large}}$ backbones (Devlin et al. 2019) according to the same implementation for $\text{ELECTRA}_{\text{base}}$. Specifically, we pre-train our methods based on $\text{BERT}_{\text{base}}$ and $\text{BERT}_{\text{large}}$ checkpoints for $200k$ steps on the Wikipedia and BooksCorpus. For a fair comparison, we also train the baseline models to the same steps. Please note that it is inadequate to pursue absolute gains for large models by using single-machine NVIDIA V100 GPUs (e.g., slower convergence speed with much smaller batch sizes), compared with TPUs for training large models in public releases (Devlin et al. 2019). Therefore, we focus on the relevant improvements between our methods and the baselines under the same training steps.

Fine-tuning For evaluation, we fine-tune the pre-trained models on GLUE (General Language Understanding Evaluation) (Wang et al. 2019) and SQuAD v1.1 (Rajpurkar et al. 2016) to evaluate the performance of the pre-trained models. GLUE include two single-sentence tasks (CoLA (Warstadt, Singh, and Bowman 2019), SST-2 (Socher et al. 2013)), three similarity and paraphrase tasks (MRPC (Dolan and Brockett 2005), STS-B (Cer et al. 2017), QQP (Chen et al. 2018)), three inference tasks (MNLI (Nangia et al. 2017), QNLI (Rajpurkar et al. 2016), RTE (Bentivogli et al. 2009). We follow ELECTRA hyper-parameters for single-task fine-tuning. We did not use any training strategies like starting from MNLI, to avoid extra distractors and focus on the fair comparison in the single-model and single-task settings.

4.2 Main Results

We evaluate the performance of our pre-training enhancement compared with the baselines in small and base sizes on GLUE and SQuAD benchmarks in Tables 3-4. From the results, we have the following observations:

1) The models with our enhanced pre-training objectives outperform the BERT and ELECTRA baselines in all the subtasks. In particular, with the same configuration and pre-training data, for both the small-size and the base-size, our methods outperform the strong ELECTRA baselines by +1.5(dev)/+1.4(test) and +0.7(dev)/+1.3(test) on average, respectively. The results demonstrate that our proposed methods improve the pre-training of ELECTRA substantially and disclose that mediating the training with true negatives is quite beneficial for improving language model pre-training.

2) Our methods outperform the baselines on both the base and large models,[8] which indicates that the false negative issue may be independent of the model size, and the training remains insufficient in training language models on different scales.

[7] For evaluation of the reading comprehension tasks, we also pre-train the variants with the length of sentences in each batch as up to 512 tokens.

[8] Since larger models obtain better baseline results, we also calculate error-reduction ratio (ERR) for comparison, e.g., the ERR of $\text{BERT}^{\text{HC}}_{\text{base}}$ and $\text{BERT}^{\text{HC}}_{\text{large}}$ is 3.6% and 2.3%, respectively. The statistics also indicate consistent strength in varied model sizes.

Model	CoLA	SST	MRPC	STS	QQP	MNLI	QNLI	RTE	Average	Δ
$\text{BERT}_{\text{base}}$	61.1	93.0	86.8	87.1	90.8	84.7	91.4	67.9	82.9	-
$\text{BERT}^{\text{HC}}_{\text{base}}$	62.9	93.2	87.5	87.4	90.9	84.9	91.5	69.3	83.5	↑0.7
$\text{BERT}^{\text{SR}}_{\text{base}}$	61.2	93.5	89.0	87.5	90.9	84.8	91.6	68.6	83.4	↑0.6
$\text{BERT}_{\text{Large}}$	61.7	93.7	88.5	90.1	91.3	86.7	92.4	72.9	84.7	-
$\text{BERT}^{\text{HC}}_{\text{large}}$	62.3	93.4	89.0	90.5	91.5	87.0	93.0	73.7	85.1	↑0.4
$\text{BERT}^{\text{SR}}_{\text{large}}$	62.3	94.2	89.2	90.1	91.4	87.0	92.8	74.0	85.1	↑0.4
$\text{ELECTRA}_{\text{small}}$	56.8	88.3	87.4	86.8	88.3	78.9	87.9	68.5	80.4	-
$\text{ELECTRA}^{\text{HC}}_{\text{small}}$	**62.0**	89.8	87.0	86.7	89.0	80.4	88.0	67.9	81.4	↑1.0
$\text{ELECTRA}^{\text{SR}}_{\text{small}}$	61.1	**90.1**	**89.5**	**87.0**	**89.4**	**80.8**	**88.8**	**68.6**	**81.9**	↑1.5
$\text{ELECTRA}_{\text{base}}$	68.3	95.3	90.9	**91.3**	91.7	88.5	93.0	82.3	87.7	-
$\text{ELECTRA}^{\text{HC}}_{\text{base}}$	**70.9**	**95.6**	**91.2**	**91.3**	**92.0**	88.7	**93.6**	83.8	**88.4**	↑0.7
$\text{ELECTRA}^{\text{SR}}_{\text{base}}$	70.4	95.4	90.4	91.2	91.9	**89.1**	93.4	**84.8**	88.3	↑0.6

Table 3: Comparisons between our proposed methods and the baseline pre-trained models on the dev set of GLUE tasks. STS is reported by Spearman correlation, CoLA is reported by Matthew's correlation, and other tasks are reported by accuracy.

Model	EM	ΔEM	F1	ΔF1
$\text{ELECTRA}_{\text{small}}$	75.8	-	83.9	-
$\text{ELECTRA}^{\text{HC}}_{\text{small}}$	**77.7**	↑1.9	**85.6**	↑1.7
$\text{ELECTRA}^{\text{SR}}_{\text{small}}$	76.0	↑0.2	84.2	↑0.3
$\text{ELECTRA}_{\text{base}}$	85.1	-	91.6	-
$\text{ELECTRA}^{\text{HC}}_{\text{base}}$	**85.7**	↑0.6	**92.1**	↑0.5
$\text{ELECTRA}^{\text{SR}}_{\text{base}}$	85.6	↑0.5	92.0	↑0.4

Table 4: Results on the SQuAD dev set. EM and F1 are short for the exact match and F1 scores (Rajpurkar et al. 2016).

3) Both SR and HC pre-training strategies help the resulting model surpass the baselines. Note that our proposed method is model-agnostic so that the convenient usability of its backbone precursor can be kept without architecture modifications. In comparison, SR is more generalizable as it does not require extra resources, while HC has the advantage of interpretation via explicit correction.

4) Our enhanced pre-training objectives show considerable performance improvements on linguistics-related tasks such as CoLA and MRPC. These tasks are about linguistic acceptability and paraphrase/semantic equivalence relationship. Besides, our methods also achieve obvious gains in tasks requiring complex semantic understanding and reasoning, such as MNLI and SQuAD, showing that they may help capture semantics to some extent.

5 Analysis

Robustness Evaluation Intuitively, our method would be helpful for improving the robustness of PLMs because the approaches may indicate lexical semantics and representation diversity during the correction or regularization operations. To verify the hypothesis, we use a robustness evaluation platform TextFlint (Wang et al. 2021) on SQuAD, from which two standard transformation methods are adapted: 1) *AddSentenceDiverse* generates distractors with altered questions and fake answers; 2) *SwapSynWordNet* transforms an input by replacing its words with synonyms provided by WordNet.

Table 5 shows the robustness evaluation results. We observe that both kinds of attacks induce a significant performance drop of the baseline system, by 54.95% and 6.0% on the EM metrics, respectively, indicating that the system is sensitive to distractors with similar meanings. In contrast, both of our models can effectively resist those attacks with less performance degradation. Specifically, the HC method works stably in the *SwapSynWordNet* attack. We speculate the reason is that the hard correction strategy captures the synonym information during pre-training, which would take advantage of lexical semantics. The other variant, the soft regularization objective, achieves much better performance in the *AddSentenceDiverse*. The most plausible reason might be the advantage of acquiring semantic diversity by regularizing the semantic distance in the SR objective.

Lookup Table from WordNet vs. Word2Vec For the hard correction approach, the candidate synonyms for detecting false negative predictions can be derived from WordNet (Miller 1992) or Word2Vec embedding space (Mikolov et al. 2013) as described in Section 3.2.[9] To verify the impact of different sources, we compare the results as shown in the second block of Table 6. We see that $\text{ELECTRA}^{\text{HC}}_{\text{WordNet}}$ outperforms $\text{ELECTRA}^{\text{HC}}_{\text{Embedding}}$ by a large margin. The most plausible reason would be that the retrieved list of synonyms from $\text{ELECTRA}^{\text{HC}}_{\text{WordNet}}$ would have higher quality than that from $\text{ELECTRA}^{\text{HC}}_{\text{Embedding}}$. Although the embedding-based method may benefit from semantic matching, but would also bring noise as it is hard to set the threshold to ensure the top-ranked words are accurate synonyms. Therefore, $\text{ELECTRA}^{\text{HC}}_{\text{WordNet}}$ turns out to be better suitable for

[9]We use the public `GloVe.6B.50d` vectors for embedding retrieval. Since the embedding method returns a ranked list by calculating the similarity score with the whole vocabulary, we only take the top 10 most similar words for each retrieval.

Model	*AddSentenceDiverse* (Ori.→Trans.)				*SwapSynWordNet* (Ori.→Trans.)			
	Exact Match	ΔEM	F1 Score	ΔF1	Exact Match	ΔEM	F1 Score	ΔF1
$\text{ELECTRA}_{\text{small}}$	80.55→25.60	↓54.95	85.10→26.43	↓58.67	80.67→74.67	↓6.00	85.38→80.43	↓4.95
$\text{ELECTRA}^{\text{HC}}_{\text{small}}$	82.59→34.13	↓48.46	86.78→36.60	↓50.18	82.33→**79.67**	↓**2.66**	86.68→**83.65**	↓3.03
$\text{ELECTRA}^{\text{SR}}_{\text{small}}$	78.84→**37.20**	↓**41.64**	80.84→**38.29**	↓**42.55**	78.67→75.67	↓3.00	80.88→78.51	↓**2.37**

Table 5: Robustness evaluation on the SQuAD dataset. Ori. represents the results of original dataset for robustness evaluation derived from the SQuAD 1.1 dev set by TextFlint (Wang et al. 2021) while Trans. indicates the transformed one. The assessed models are the small models from Table 4.

Model	CoLA	SST	MRPC	STS	QQP	MNLI	QNLI	RTE	Average	Δ
$\text{ELECTRA}_{\text{small}}$	56.8	88.3	87.4	86.8	88.3	78.9	87.9	68.5	80.4	-
$\text{ELECTRA}^{\text{HC}}_{\text{WordNet}}$	62.0	89.8	87.0	86.7	89.0	80.4	88.0	67.9	81.4	↑1.0
$\text{ELECTRA}^{\text{HC}}_{\text{Embedding}}$	59.0	88.5	87.0	86.4	88.8	79.6	87.9	67.1	80.6	↑0.2
$\text{ELECTRA}^{\text{SR}}_{\text{Word}}$	61.1	90.1	89.5	87.0	89.4	80.8	88.8	68.6	81.9	↑1.5
$\text{ELECTRA}^{\text{SR}}_{\text{Sent}}$	59.5	89.6	90.0	86.7	89.1	80.4	90.0	68.2	81.6	↑1.2

Table 6: Comparative studies of variants on GLUE dev sets on small models. The first block shows our baseline. The second block presents the results of HC methods based on WordNet and Word2Vec embedding. The third block compares the word-level regularization and sentence-level regularization.

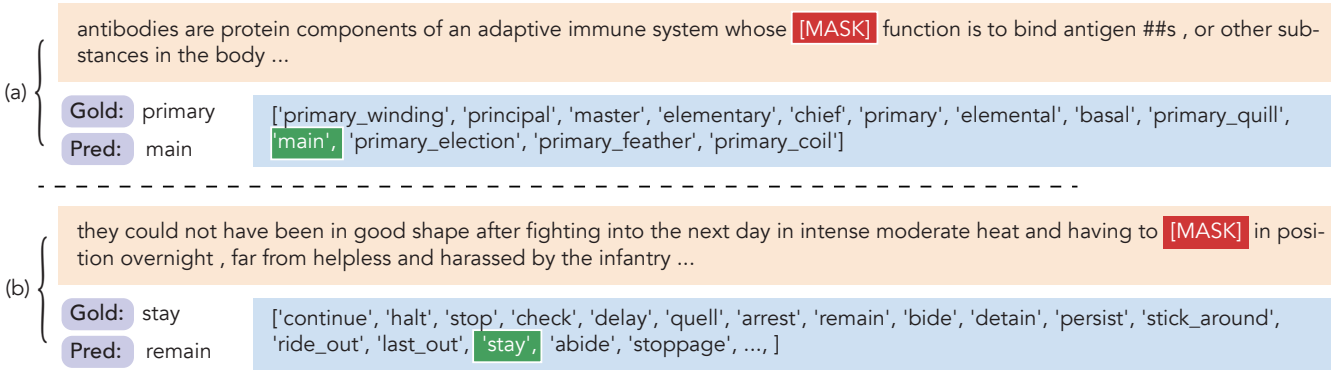

Figure 3: Interpretation of the hard correction process. The orange box contain the input sentence, the purple buttons indicate the gold and predicted tokens, and the blue box shows the WordNet synonyms for the gold token.

our task.[10]

From Word-level to Sentence-level Regularization The soft regularization approach measures the semantic distance between the predicted one and the ground truth, which may neglect the sentence-level context. We are interested in whether measuring the sentence-level similarity would achieve better results. To verify the hypothesis, we fill the masked sentence $\mathbf{s}$ with the predicted tokens r_k to have the predicted sentence s_p. Then, s_p and s are fed to the Transformer encoder to have the contextualized representation H_p and H_s, respectively. To guide the probability distribution of model predictions H_p to match the expected probability distribution H_s, we adopt Kullback–Leibler (KL) divergence: $\mathcal{L}_{kl} = \text{KL}(H_p \parallel H_s)$, where $\mathcal{L}_{kl}$ is applied as the degree of sentence-level semantic mismatch. In detail, we first apply softmax on the two hidden representations to obtain two distributions, and then the KL divergence is calculated between those two distributions. The loss function is then written as: $\mathcal{L}' = \mathcal{L}_{dlm} + \mathcal{L}_{kl}$. For clarity, we denote the original $\text{ELECTRA}^{\text{SR}}_{\text{small}}$ method described in Eq. 3 as $\text{ELECTRA}^{\text{SR}}_{\text{Word}}$ and the sentence-level variant as $\text{ELECTRA}^{\text{SR}}_{\text{Sent}}$.

The comparative results are reported in the third block of Table 6, which indicates that using sentence-level regularization ($\text{ELECTRA}^{\text{SR}}_{\text{Sent}}$) also outperforms the baseline and nearly reaches the performance of word-level one ($\text{ELECTRA}^{\text{SR}}_{\text{Word}}$) on average, with slightly better results on MRPC and MNLI. Although $\text{ELECTRA}^{\text{SR}}_{\text{Sent}}$ still keeps the same parameter size with baseline, it leads to more computation resources because it requires the extra calculation for the predicted sequence H_p. Therefore, considering the bal-

[10]Since the HC method relies on synonym retrieval from external sources, it is possible to bring false positive corrections theoretically. However, we seldom detect such cases in our preliminary experiments, so we leave the open question for interested readers to avoid deviating from the focus of this paper.

ance between effectiveness and efficiency, ELECTRA$^{\text{SR}}_{\text{Word}}$ can serve as the first preferred choice for practical applications, and ELECTRA$^{\text{SR}}_{\text{Sent}}$ can be employed when computation resources are sufficient.[11]

Case Studies To interpret how our method works, we randomly select some hard correction examples as shown in Figure 3 by taking the ELECTRA$_{\text{small}}$ as the baseline model. We find that the baseline model produces reasonable predictions such as *main* and *remain*, as opposed to the golds ones, *primary* and *stay*. Those predictions will be determined as wrong and possibly harm pre-training. Fortunately, such cases can be easily solved by our proposed method. Though the synonym list may contain irrelevant words, our correction will not bring false positives because it only cares about whether the predicted word is in the shortlist or not. Being a detected synonym is a sufficient condition, though it is not a necessary condition as those predictions make up the subset of false negatives. Therefore, inappropriate options in the list would not bring side effects.

6 Related Work

Self-supervised learning is one of the major topics in training pre-trained models (Peters et al. 2018; Radford et al. 2018; Devlin et al. 2019; Zhu et al. 2022), which decides how the model captures knowledge from large-scale unlabeled data. Recent studies have investigated denoising patterns (Raffel et al. 2020; Lewis et al. 2020), MLM alternatives (Yang et al. 2019), and auxiliary objectives (Lan et al. 2020; Joshi et al. 2020) to improve the power of pre-training. However, studies show that the current models still suffer from under-fitting issues, and it remains challenging to find efficient training strategies (Rogers, Kovaleva, and Rumshisky 2020).

Denoising Patterns MLM has been widely used for pre-training (Devlin et al. 2019; Lan et al. 2020; Clark et al. 2020; Song et al. 2020), in which the fundamental part is how to construct high-quality masked examples (Raffel et al. 2020). The current studies commonly define specific patterns for mask corruption. For example, some are motivated from the language modeling units, such as subword masking (Devlin et al. 2019), span masking (Joshi et al. 2020), and n-gram masking (Levine et al. 2021; Li and Zhao 2021). Some employ edit operations like insertion, deletion, replacement, and retrieval (Lewis et al. 2020; Guu et al. 2020). Others seek for external knowledge annotations, such as named entities (Sun et al. 2019), semantics (Zhou et al. 2020; Zhang et al. 2020b), and syntax (Zhang et al. 2020c; Xu et al. 2021). To provide more diversity of mask tokens, RoBERTa applied dynamic masks in different training iterations (Liu et al. 2019). These prior studies either employ pre-defined mask construction patterns or improve the diversity of mask tokens to help capture knowledge from pre-training.

[11]Besides the methods we discussed in this work, there are alternative ways to achieve the regularization effects, e.g., using a softmax temperature with the standard loss.

MLM Alternatives To alleviate the task mismatch between the pre-training and the fine-tuning tasks, XLNet (Yang et al. 2019) proposed an autoregressive objective for language modeling through token permutation, which further adopts a more complex model architecture. Instead of corrupting sentences with the mask symbol that never appears in the fine-tuning stage, MacBERT (Cui et al. 2020) proposes to use similar words for the masking purpose. Yamaguchi et al. (2021) also investigates simple pre-training objectives based on token-level classification tasks as replacements of MLM, which are often computationally cheaper and result in comparable performance to MLM. In addition, training sequence-to-sequence (Seq2Seq) language models has also aroused continuous interests (Dong et al. 2019; Lewis et al. 2020; Raffel et al. 2020).

Auxiliary Objectives Another research line is auxiliary objectives in conjunction with MLM, such as next sentence prediction (Devlin et al. 2019), span-boundary objective (Joshi et al. 2020), and sentence-order prediction (Lan et al. 2020). Such line of research emerges as hot topics, especially in domain-specific pre-training, such as dialogue-oriented language models (Zhang et al. 2020a; Wu et al. 2020; Zhang and Zhao 2021).

As the major difference from the existing studies, our work devotes itself to mediating misconceived negatives as the essential drawback of MLM during the MLE estimation and aiming to guide language models to learn from true negatives through our newly proposed regularization and correction methods. Besides the heuristic pre-trained patterns like masking strategies during data construction, we stress that there are potential post-processing strategies to guide the MLM training: correction and pruning. Those strategies are considered to deal with the false negative issue during MLM training, where the model would yield reasonable predictions but discriminated as wrong predictions because such predictions do not match the single gold token for each training case. For example, many tokens are reasonable but written in different forms or are the synonyms of the expected gold token. We could directly drop the uncertain predictions or correct the training with soft regularization. Promoting our view to sentence level, the similarity between the predicted sentence and the original sentence can also be taken into account to measure the sentence-level confidence that indicates how hard the task is, which would be beneficial to provide more fine-grained signals and thus improve the training quality.

7 Conclusions

The work identifies the false negative issue in language model pre-training and proposes methods to counteract it. Though discriminative PLMs may quite straightforwardly suffer from the false negative issue according to our exploration in this work, it has been completely ignored for a long time, and it is a bit surprising that maybe this work is the first one that formally considers such a big pre-training leak. Our work indicates that mediating false negatives is so important that counter-false-negative pre-training can synchronously improve the effectiveness and robustness of PLMs.

References

Bentivogli, L.; Clark, P.; Dagan, I.; and Giampiccolo, D. 2009. The Fifth PASCAL Recognizing Textual Entailment Challenge. In *ACL-PASCAL*.

Bordes, A.; Usunier, N.; García-Durán, A.; Weston, J.; and Yakhnenko, O. 2013. Translating Embeddings for Modeling Multi-relational Data. In Burges, C. J. C.; Bottou, L.; Ghahramani, Z.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems*, 2787–2795.

Cer, D.; Diab, M.; Agirre, E.; Lopez-Gazpio, I.; and Specia, L. 2017. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 1–14. Vancouver, Canada.

Chen, W.; Li, P.; Chan, H. P.; and King, I. 2021. Dialogue summarization with supporting utterance flow modelling and fact regularization. *Knowledge-Based Systems*, 229: 107328.

Chen, Z.; Zhang, H.; Zhang, X.; and Zhao, L. 2018. Quora question pairs. `https://www.kaggle.com/c/quora-question-pairs`. Accessed: 2022-06-01.

Clark, K.; Luong, M.; Le, Q. V.; and Manning, C. D. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *8th International Conference on Learning Representations*.

Cui, Y.; Che, W.; Liu, T.; Qin, B.; Wang, S.; and Hu, G. 2020. Revisiting Pre-Trained Models for Chinese Natural Language Processing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 657–668. Online.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*, 4171–4186. Minneapolis, Minnesota.

Dolan, W. B.; and Brockett, C. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; and Hon, H. 2019. Unified Language Model Pre-training for Natural Language Understanding and Generation. In Wallach, H. M.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E. B.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, 13042–13054.

Guu, K.; Lee, K.; Tung, Z.; Pasupat, P.; and Chang, M.-W. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Jiang, H.; He, P.; Chen, W.; Liu, X.; Gao, J.; and Zhao, T. 2020. SMART: Robust and Efficient Fine-Tuning for Pre-trained Natural Language Models through Principled Regularized Optimization. In *ACL*, 2177–2190. Online.

Joshi, M.; Chen, D.; Liu, Y.; Weld, D. S.; Zettlemoyer, L.; and Levy, O. 2020. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *TACL*, 8: 64–77.

Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; and Soricut, R. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *8th International Conference on Learning Representations*.

Levine, Y.; Lenz, B.; Lieber, O.; Abend, O.; Leyton-Brown, K.; Tennenholtz, M.; and Shoham, Y. 2021. {PMI}-Masking: Principled masking of correlated spans. In *International Conference on Learning Representations*.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *ACL*, 7871–7880. Online.

Li, Y.; and Zhao, H. 2021. Pre-training Universal Language Representation. In *ACL-IJCNLP*, 5122–5133. Online.

Li, Z.; Wang, R.; Chen, K.; Utiyama, M.; Sumita, E.; Zhang, Z.; and Zhao, H. 2020. Data-dependent Gaussian Prior Objective for Language Generation. In *8th International Conference on Learning Representations*.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2017. Pointer Sentinel Mixture Models. In *5th International Conference on Learning Representations*.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed Representations of Words and Phrases and their Compositionality. In Burges, C. J. C.; Bottou, L.; Ghahramani, Z.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems*, 3111–3119.

Miller, G. A. 1992. WordNet: A Lexical Database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

Nangia, N.; Williams, A.; Lazaridou, A.; and Bowman, S. 2017. The RepEval 2017 Shared Task: Multi-Genre Natural Language Inference with Sentence Representations. In *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP*, 1–10. Copenhagen, Denmark.

Peters, M.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep Contextualized Word Representations. In *NAACL-HLT*, 2227–2237. New Orleans, Louisiana.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training. *Technical Re*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21: 1–67.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2383–2392. Austin, Texas.

Rogers, A.; Kovaleva, O.; and Rumshisky, A. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8: 842–866.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A.; and Potts, C. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1631–1642. Seattle, Washington, USA.

Song, K.; Tan, X.; Qin, T.; Lu, J.; and Liu, T.-Y. 2020. Mpnet: Masked and permuted pre-training for language understanding. *arXiv preprint arXiv:2004.09297*.

Sun, Y.; Wang, S.; Li, Y.; Feng, S.; Chen, X.; Zhang, H.; Tian, X.; Zhu, D.; Tian, H.; and Wu, H. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2019. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Wang, C.; Liu, P.; and Zhang, Y. 2021. Can Generative Pre-trained Language Models Serve As Knowledge Bases for Closed-book QA? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 3241–3251.

Wang, W.; Bi, B.; Yan, M.; Wu, C.; Xia, J.; Bao, Z.; Peng, L.; and Si, L. 2020. StructBERT: Incorporating Language Structures into Pre-training for Deep Language Understanding. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Wang, X.; Liu, Q.; Gui, T.; Zhang, Q.; Zou, Y.; Zhou, X.; Ye, J.; Zhang, Y.; Zheng, R.; Pang, Z.; et al. 2021. Textflint: Unified multilingual robustness evaluation toolkit for natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, 347–355.

Warstadt, A.; Singh, A.; and Bowman, S. R. 2019. Neural Network Acceptability Judgments. *Transactions of the Association for Computational Linguistics*, 7: 625–641.

Wieting, J.; Berg-Kirkpatrick, T.; Gimpel, K.; and Neubig, G. 2019. Beyond BLEU:Training Neural Machine Translation with Semantic Similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4344–4355. Florence, Italy.

Wu, C.-S.; Hoi, S. C.; Socher, R.; and Xiong, C. 2020. TOD-BERT: Pre-trained Natural Language Understanding for Task-Oriented Dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 917–929. Online.

Xu, Z.; Guo, D.; Tang, D.; Su, Q.; Shou, L.; Gong, M.; Zhong, W.; Quan, X.; Jiang, D.; and Duan, N. 2021. Syntax-Enhanced Pre-trained Model. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 5412–5422. Online.

Yamaguchi, A.; Chrysostomou, G.; Margatina, K.; and Aletras, N. 2021. Frustratingly Simple Pretraining Alternatives to Masked Language Modeling. *arXiv preprint arXiv:2109.01819*.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J. G.; Salakhutdinov, R.; and Le, Q. V. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In Wallach, H. M.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E. B.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 5754–5764.

Zhang, Y.; Sun, S.; Galley, M.; Chen, Y.-C.; Brockett, C.; Gao, X.; Gao, J.; Liu, J.; and Dolan, B. 2020a. DIALOGPT : Large-Scale Generative Pre-training for Conversational Response Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 270–278. Online.

Zhang, Z.; Wu, Y.; Zhao, H.; Li, Z.; Zhang, S.; Zhou, X.; and Zhou, X. 2020b. Semantics-aware BERT for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 9628–9635.

Zhang, Z.; Wu, Y.; Zhou, J.; Duan, S.; Zhao, H.; and Wang, R. 2020c. SG-Net: Syntax Guided Transformer for Language Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhang, Z.; and Zhao, H. 2021. Structural Pre-training for Dialogue Comprehension. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 5134–5145. Online.

Zhou, J.; Zhang, Z.; Zhao, H.; and Zhang, S. 2020. LIMIT-BERT : Linguistics Informed Multi-Task BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 4450–4461. Online.

Zhu, P.; Cheng, D.; Luo, S.; Xu, R.; Liang, Y.; and Luo, Y. 2022. Leveraging enterprise knowledge graph to infer web events' influences via self-supervised learning. *Journal of Web Semantics*, 100722.

Zhu, Y.; Kiros, R.; Zemel, R. S.; Salakhutdinov, R.; Urtasun, R.; Torralba, A.; and Fidler, S. 2015. Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, 19–27.