Improving BERT with Self-Supervised Attention

Xiaoyu Kou[1,*,†], **Yaming Yang**[2,*], **Yujing Wang**[1,2,*], **Ce Zhang**[3,†]
Yiren Chen[1,†], **Yunhai Tong**[1], **Yan Zhang**[1], **Jing Bai**[2]
[1]Key Laboratory of Machine Perception (MOE)
Department of Machine Intelligence, Peking University
[2]Microsoft Research Asia [3]ETH Zürich
{kouxiaoyu, yrchen92, yhtong, zhyzhy001}@pku.edu.cn
{yayaming, yujwang, jbai}@microsoft.com
ce.zhang@inf.ethz.ch

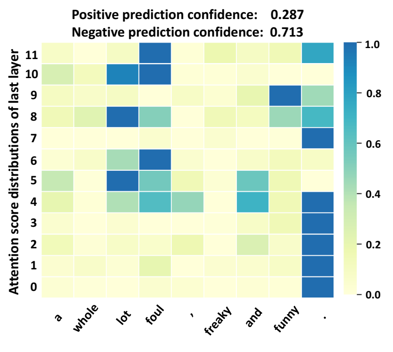
Abstract

One of the most popular paradigms of applying large, pre-trained NLP models such as BERT is to fine-tune it on a smaller dataset. However, one challenge remains as the fine-tuned model often overfits on smaller datasets. A symptom of this phenomenon is that irrelevant words in the sentences, even when they are obvious to humans, can substantially degrade the performance of these fine-tuned BERT models. In this paper, we propose a novel technique, called *Self-Supervised Attention* (SSA) to help facilitate this generalization challenge. Specifically, SSA automatically generates weak, token-level attention labels iteratively by "probing" the fine-tuned model *from the previous iteration*. We investigate two different ways of integrating SSA into BERT and propose a hybrid approach to combine their benefits. Empirically, on a variety of public datasets, we illustrate significant performance improvement using our SSA-enhanced BERT model.

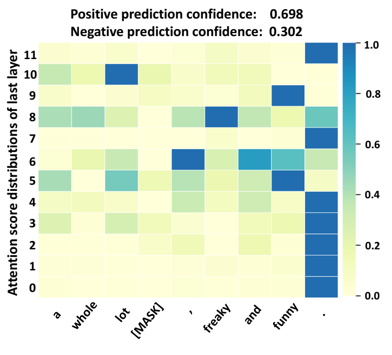
1 Introduction

Models based on self-attention such as Transformer (Vaswani et al., 2017) have shown their effectiveness for a variety of NLP tasks. One popular use case is to leverage a single pre-trained language model, e.g., BERT (Devlin et al., 2018), and transfer it to a specific downstream task. However, opportunities remain to further improve these fine-tuned models as many of them often overfit, especially on smaller datasets.

Motivating Example: One Symptom of Overfitting. This paper is motivated by the observation that many fine-tuned models are very sensitive to the irrelevant words to a sentence. For example, consider the sentence "a whole lot foul, freaky and funny." from SST (Stanford Sentiment Treebank) dataset. In vanilla BERT, this sentence is predicted as negative while its actual label is positive. As illustrated in Figure 1(a), one reason behind this is that the word "foul", which is often associated with negative predictions, was given probably too predominant attention scores by the BERT model. Instead, by masking the irrelevant word "foul" (see Figure 1(b)), the model is able to focus on more relevant word "funny", and the final prediction flips to the correct label.



(a) Original sentence



(b) Sentence after masking a word

Figure 1: The multi-head attention scores of each word on the last layer, obtained by BERT on SST dataset. The ground-truth of sentence "a whole lot foul, freaky and funny." is positive.

Self-Supervised Attention (SSA) In this paper, we hope to alleviate the above overfitting problem

[*]Equal Contribution.
[†]The work was done when the authors visited MSRA

by introducing auxiliary knowledge to the attention layer of BERT. An interesting aspect of our approach is that we do not need any additional information (such as those in MT-DNN (Liu et al., 2019a)) for this auxiliary task. Instead, we propose a novel mechanism called *self-supervised attention* (SSA), which utilizes self-supervised information as the auxiliary loss.

The idea behind SSA is simple — Given a sentence of n tokens $S = (t_1, ...t_i..., t_n)$ predicted by the model with label y, we change the sentence into S' by, for example, masking the token t_i, and generate the predicted label as y'. If y' is different from y, we set the SSA score of t_i as 1, otherwise 0. Intuitively, this means that if the token t_i plays a prominent role in the current model (i.e., the label prediction will be flipped if this token is masked), it will correspond to higher importance with a larger SSA score. Therefore, we can leverage a SSA layer to improve the accuracy and generalization ability of original model by correctly predicting each token's SSA score.

At first glance, it might be mysterious how SSA score helps the performance of a deep neural network model — all the information of SSA comes from the model itself anyway. The intuition behind our approach is to impose constraints over the model to obtain several good properties. (1) If a model can predict SSA scores correctly, its decision surface is relatively smooth and less noisy. In other words, if we randomly mask some irrelevant words, the decision surface should be more robust. (2) By forcing the learned feature representation to predict the auxiliary task of SSA, the learned features are able to encode more information about the keywords in a sentence and obtain better capability of generalization .

Summary of Technical Contributions Our first contribution is a co-training framework that precisely encodes the above motivation by a joint loss of target task and SSA. This itself already improves the performance of BERT significantly on some tasks. We then explore the idea of directly integrating SSA as an attention layer into the model. As our second contribution, we propose a hybrid framework that combines both co-training and an additional self-supervised attention layer. This further improves the performance of our model on a variety of public datasets. We also demonstrate that the token-level SSA scores learned by the model agree well with human common-sense.

2 Related Work

One category of methods to address the overfitting problem is to increases the volume of training data. STILTs (Phang et al., 2018) introduces an intermediate-training phase rather than directly fine-tuning on the target task, for example, training on MNLI dataset before fine-tuning on the target RTE dataset. RoBERTa (Liu et al., 2019b) focuses on training a better language model, which obtains a more powerful pre-trained checkpoint and improves the performance of downstream tasks significantly. Specifically, it collects a larger corpus, employs better hyper-parameters, and trains more iterations than BERT. Wu et al. 2018 proposes a data augmentation method, named conditional BERT contextual augmentation, to generate synthetic data samples to be additional training data.

Another line of research focuses on auxiliary tasks. For instance, Sun et al. 2019 leverages powerful BERT to tackle Aspect-Based Sentiment Analysis (ABSA) task. By constructing an auxiliary sentence from the aspect, authors convert ABSA to a sentence-pair classification task, such as question answering (QA) and natural language inference (NLI). In this way, the original ABSA task is adapted to BERT, and a new state-of-the-art result is established. MT-DNN (Liu et al., 2019a) presents a multi-task learning framework that leverages large amounts of cross-task data to generate more general representations. AUTOSEM (Guo et al., 2019) proposes a two-stage multi-task learning pipeline, where the most useful auxiliary tasks are selected in the first stage and the training mixing ratio of the selected auxiliary tasks is learned in the second stage.

Our work is also related to attention mechanism, which has been extensively studied for modeling token relationships in natural language sentences. Previous works mainly focus on how to enhance sentence representation. For instance, Yang et al. 2016 proposes hierarchical attention network to progressively build a document vector for document classification; Chen et al. 2018 utilizes a vector-based multi-head attention pooling method to enhance sentence embedding. Tan et al. 2017 is the most relevant method to our work, which utilizes a graph-based attention mechanism in the sequence-to-sequence framework for abstractive document summarization.

In addition, some methods for interpretability calculation are relevant to the SSA score generation strategy in our solution. For example, LIME (Ribeiro et al., 2016) focuses on explaining model's predictions by sampling perturbations of the input and treating the decision changing as a classification boundary. Besides, the gradient-based technique GradCAM (Selvaraju et al., 2017) is another method for modeling interpretability in the computer vision literature. It uses the gradients of any target concept and a convolutional layer to produce a coarse localization map highlighting the important regions in the image. The major focus of this paper is to addresses the usefulness of token importance to improve model accuracy rather than model interpretability. Our SSA score generation strategy can be improved by LIME or GradCAM potentially, which is left to future work.

The model proposed in this paper is distinct from previous works in the following perspective. (1) Our solution does not require any augmented dataset explicitly. Instead, we generate some auxiliary training data for the SSA task and utilize it to improve the generalization capability of the model. (2) We propose a novel auxiliary task, Self-Supervised Attention (SSA), which differentiates the importance of each token with respect to the optimization target. (3) We apply an additional self-supervised attention layer to the vanilla BERT model for amplifying the effect of relevant tokens and diminishing the impact of irrelevant tokens. With the self-supervised attention layer, the model becomes more easier for optimization and more resistant to overfitting.

3 Model

This section presents an overview of the proposed *Self-Supervised Attention* (SSA) model. First, in Section 3.1, we introduce the auxiliary task of SSA and describe the training data generation procedure for this task. Second, in Section 3.2, we introduce the co-training framework to optimize the target task and auxiliary task simultaneously. As last, the hybrid model with self-supervised attention layer is proposed in Section 1.

3.1 The auxiliary task of SSA

Based on a sentence S, we generate another sentence S' by randomly masking several tokens in the original sentence, which constructs a noisy neighbor of S. Specifically, if S' does not affect the predicted label, the modified tokens can be de-emphasized for improving the generalization capability of the original model. Otherwise, we want to emphasize the modified tokens to address their task-oriented importance. Motivated by this, we propose a novel auxiliary task: *Self-Supervised Attention* (SSA), which learns a task-oriented weighting score for each token. If we correctly predict the importance of each token for a specific task, the decision surface between S and S' can be smoother and more generalized.

Given a sentence $S = (t_1, ...t_i..., t_n)$, the SSA task outputs a binary output vector $Y = (y_1^t, ..., y_i^t..., y_n^t)$, where y_i^t indicates how important the token t_i is to the target task. The loss function can be formulated as follows:

$$\mathcal{L}_{SSA} = -\sum_i \ell_{SSA}(y_i, y_i^t),$$
$$y_i^t = \sigma\big(w_{SSA}^i M(t_i)\big)$$

where $M(t_i)$ denotes the model from the previous epoch, w_{SSA}^i is the fully connected layer of the i-th token for SSA task, σ is a softmax operation, y_i denotes the SSA label of token t_i, and ℓ_{SSA} is the cross entropy loss.

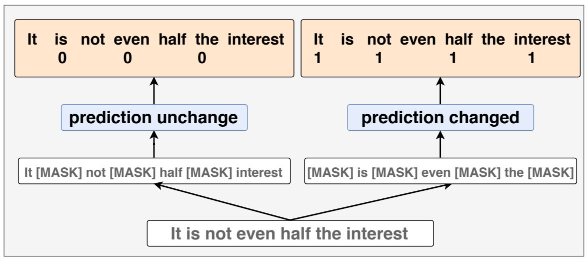

It is not even half the interest
0 0 0
It is not even half the interest
1 1 1 1
prediction unchange
prediction changed
It [MASK] not [MASK] half [MASK] interest
[MASK] is [MASK] even [MASK] the [MASK]
It is not even half the interest

Figure 2: Label generation procedure

The training data generation procedure for SSA task is described as follows. Given a sentence S, we first mask several tokens randomly and get a new sentence S'. Then, we inference the predicted label of two sentences using the same model, $M(t_i)$. If both predicted labels are the same, we set the importance labels of masked tokens as 0, as they have little impact on the target task. Otherwise, these tokens should be emphasized and the corresponding importance labels are set to be 1. We only process the sentences S that the prediction of $M(t_i)$ is correct to get rid of noises. The benefit of this operation is that with the increase of epochs, the label given by the of the $M(t_i)$ becomes more and more accurate. Besides, the number of tokens being masked is proportional to the length of the sentence, which is set to 0.3 empirically. We use a generation ratio γ to control the amount of generated sentences, and larger γ means more sentences will be generated.

We train different samples generated by the the same sentence in different iterations without slowing down the training procedure. Figure 2 shows two training samples generated by a sentence in multiple iterations. Suppose that we have an input sentence S ="It is not even half the interest", we generate the masked sentence S_1' ="It [MASK] not [MASK] half [MASK] interest" or S_2' ="[MASK] is [MASK] even [MASK] the [MASK]". S_1' does not change the prediction label, so the masked tokens are labeled by 0, that is, $Y_1 = (\_, 0, \_, 0, \_, 0, \_)$, where "_" does not contribute to the loss function. On the other hand, S_2' flips the original prediction of S, so we have $Y_2 = (1, \_, 1, \_, 1, \_, 1)$.

3.2 Co-training Framework

A straight-forward way to leverage the SSA task is training the target task and SSA jointly. Specifically, the overall loss can be defined as a linear combination of two parts:

$$\mathcal{L} = \alpha \mathcal{L}_{target} + (1 - \alpha)\mathcal{L}_{SSA}$$

where:

$$\mathcal{L}_{target} = \sum_{i=1} \ell_{target}(y_i, y_i^s),$$

where ℓ_{target} is the loss function of the target task, for example, negative log-likelihood loss for sentiment classification and mean-squared loss for regression task; y_i and y_i^s denote the actual label of sentence s_i and the predicted label for the target task respectively; $\mathcal{L}_{target}$ denotes the loss function of the target task while $\mathcal{L}_{SSA}$ represents for that of SSA task; α is a linear combination ratio which controls the relative importance of two losses.

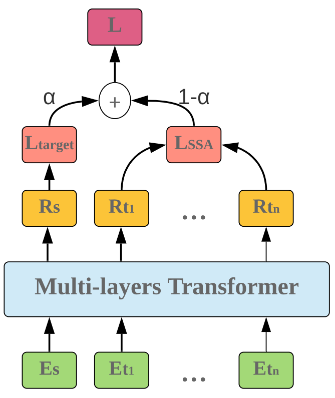


Figure 3: The model architecture for co-training

The model architecture for co-training is illustrated in Figure 3. Each token in the input sentence is mapped to an representation, and then fed to an encoder of BERT or multi-layers Transformer. The output of the encoder consists of a sentence representation vector R_s and token-level representation vectors $[R_{t_1}, R_{t_2}, ..., R_{t_n}]$. The sentence representation is used for target task prediction and the token representation vectors are leveraged in the SSA task. The co-training framework optimizes these two tasks by the target loss and auxiliary loss of SSA simultaneously.

3.3 Hybrid Model with SSA layer

The limitation of co-training framework is that the SSA task can not impact target prediction explicitly. It only acts as a regularizer on the loss function to force the learned embedding vectors of tokens to encode their relevant importance. Intuitively, if an irrelevant token exists in an training sentence, we can mask the token explicitly and guide the model to capture more important information for alleviating the overfitting problem. Therefore, we add an additional *self-supervised attention* (SSA) layer on the vanilla BERT model, which re-weights each token explicitly according to their relative importance to the target task.

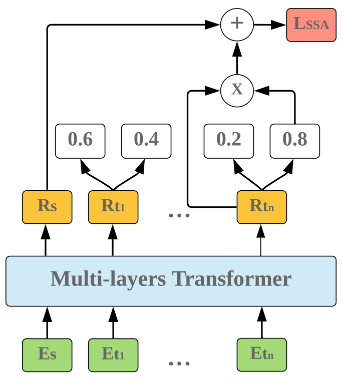


Figure 4: Hybrid Model with SSA layer

$$R_o = \beta \cdot R_{[CLS]} + (1 - \beta) \sum_i \left(\sigma(y_i^t) \cdot R_i\right) \quad (1)$$

The hybrid model with SSA layer is illustrated in Figure 4 and Equation 1 is a mathematical formulation. It yields an extra sentence representation by summing up all the token embedding vectors R_i weighted by the SSA prediction scores y_i^t after the softmax operation σ. Afterwards, the extra sentence embedding output by the SSA layer and the original sentence embedding vector $R_{[CLS]}$ are linearly combined as the final sentence representation R_o. β is a hyper-parameter to control the relative weight of this linear combination.

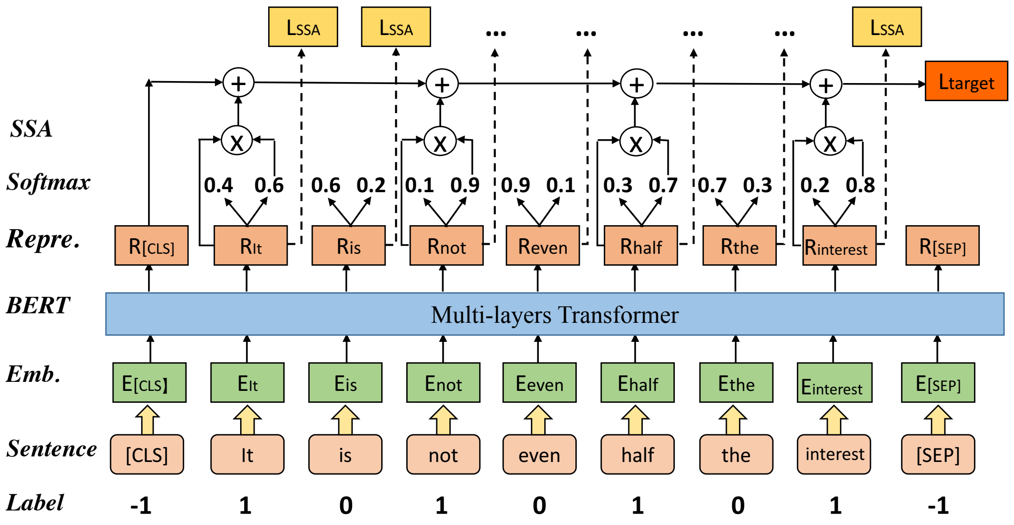


Figure 5: Illustration of an example based on the Hybrid model

Figure 5 further demonstrates a concrete example of hybrid model. Take "It is not even half the interest" as input, the discrete tokens are mapped to embedding representations first, then the embedding vectors are transformed by a multi-layer transformer encoder (e.g., BERT), to generate sentence embedding and token representations. As a result, the SSA layer identifies that 'It', 'not', 'half', and 'interest' are important for the target sentiment classification task, and then constructs an extra sentence embedding by weighted summation of token-level embedding vectors based on the corresponding SSA scores. The final sentence representation is a linear combination of original sentence embedding and extra sentence embedding. The loss function of primary task and SSA task are jointly optimized in a co-training framework.

4 Experiments

In this section, we describe the experimental details about the proposed models **SSA-Co** and **SSA-Hybrid** on the GLUE benchmark.

4.1 Tasks

The General Language Understanding Evaluation (GLUE) benchmark is a collection of datasets for training and evaluating neural language understanding models. The statistics of datasets are summarized in Table 1, where **CoLA** (Warstadt et al., 2018) and **SST-2** (Socher et al., 2013) are datasets for single-sentence classification tasks, **STS-B** (Cer et al., 2017) is for text similarity task, **MRPC** (Dolan and Brockett, 2005) and **QQP** are binary classification tasks, and the rests are natural language inference tasks including **MNLI** (Williams et al., 2017), **QNLI** (Rajpurkar et al., 2016), **RTE** (Dagan et al., 2005) and **WNLI** (Levesque et al., 2012). We follow the default evaluation metrics for tasks in GLUE benchmark.

Task	#Train	#Dev.	#Test	#C.
CoLA	8,551	1,042	1,064	2
SST-2	67,350	873	1,822	2
MRPC	3,669	409	1,726	2
STS-B	5,750	1,501	1,380	*
QQP	363,871	40,432	390,965	2
MNLI	392,703	9,816/9,833	9,797/9,848	3
QNLI	104,744	5,464	5,464	2
RTE	2,491	278	3,001	2
WNLI	636	72	147	2

Table 1: Statistics of all datasets. The numbers on the left and right side of character "/" for task MNLI represent MNLI-m and MNLI-mm correspondingly; "#C." is number of categories in the task; "*" denotes regression task.

For each task, we use the default train/dev/test split. The model is trained on the training set, while the hyper-parameters are chosen based on the development set. We submit the prediction results of test set to the GLUE evaluation service[2] and report the final evaluation scores.

4.2 Algorithms

The pre-training and fine-tuning frameworks have been achieved rapid evolution since BERT was proposed. As introduced in Related Work section, there are many existing works on data augmentation and leveraging axillary tasks. Our proposed algorithm is orthogonal to these techniques and can be applied to more advanced models. In this paper,

[2]https://gluebenchmark.com

GLUE test set results of BERT-base										
Model	Avg	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	WNLI
BERT	78.3	52.1	93.5	88.9/84.8	87.1/85.8	71.2/89.2	84.6/83.4	90.5	66.4	**65.1**
BERT-r	78.4	51.8	94.0	87.1/82.8	86.9/86.0	71.0/89.1	84.2/83.2	90.5	69.1	**65.1**
BERT-m	78.3	53.1	93.7	88.1/83.9	86.6/85.4	69.8/88.6	84.2/83.1	90.6	67.2	**65.1**
BERT-SSA-Co	79.0	52.8	94.1	89.0/85.2	87.2/86.1	71.1/89.1	84.6/83.7	90.8	69.8	**65.1**
BERT-SSA-H	**79.3**	**54.1***	**94.4**	**89.1/85.4***	**87.2/86.3**	**71.5/89.3****	**84.7/84.1***	**91.3***	**70.3****	**65.1**

Table 2: GLUE test set results scored by GLUE evaluation service. The results of BERT-r are reproduced with the open sourced codes and the reproduction numbers are on-par with the numbers reported in the GLUE leaderboard[1]. "BERT-m" represents our baseline model "BERT-mask"; "SSA-H" denotes our SSA-Hybrid method; The result on the left and right side of character "/" for task MNLI represents MNLI-m and MNLI-mm correspondingly; We also conduct significant test between **SSA-Hybrid** and **BERT**, where * means the improvement over vanilla BERT is significant at the 0.05 significance level; ** means the improvement over vanilla BERT is significant at the 0.01 significance level;

the comparison are mainly on the basis of vanilla BERT while more experiments on other model variants (e.g., MT-DNN) are left to future works.

The configurations of baselines and our solutions are described as below.

BERT (Devlin et al., 2018) is a multi-layer bidirectional transformers. It is first pre-trained on a large corpus containing 3,300M words, and then fine-tuned on downstream tasks. We download the checkpoint of BERT-base[3] and BERT-large[4] from the official website.

BERT-mask is a simple baseline to our algorithm, where the training data is augmented by randomly masking tokens in the original sentences in the same way as SSA. By comparing our solution with this simple baseline, we can clearly examine the superiority of self-supervised attention.

RoBERTa (Liu et al., 2019b) is an improved recipe of BERT. It worth noting that the results reported on the leaderboard are ensembles of single-task models. For a fair comparison, the results of RoBERTs in our experiments are reproduced based on the officially released checkpoint[5].

SSA-Co and **SSA-Hybrid** are the models proposed in this paper, where SSA-Co takes SSA as an auxiliary task and leverages the co-training framework to optimize the two tasks jointly; SSA-Hybrid takes the hybrid model which contains an additional *self-supervised attention* (SSA) layer in the network structure.

[3] `https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip`

[4] `https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-24_H-1024_A-16.zip`

[5] `https://github.com/pytorch/fairseq`

4.3 Detailed Settings

All experiments are based on the publicly available implementation of BERT and RoBERTa (PyTorch)[6], and we follow the fine-tuning regime specified in (Devlin et al., 2018). While many submissions to the GLUE leaderboard depend on multitask fine-tuning or ensembles, **our submission depends only on single-task fine-tuning**.

We employ grid search to find the optimal hyper-parameters according to a pre-defined range: learning rate $lr \in$ {1e-5, 2e-5}, batch size $b \in$ {16, 32}, epochs $T \in$ {2, 3, 5}, loss combination ratio $\alpha \in${0.7, 0.9}, linear combination ratio $\beta \in${0.2, 0.5, 0.9} and auxiliary data generation ratio for SSA and BERT-m $\gamma \in \{0.6, 1.0, 2.0\}$. Note that the maximum sequence length is set to 128 in all of our experiments.

We also found that when fine-tuning the model on some small datasets, the convergence is not stable. To alleviate this problem, we evenly split the validation set into three subset and choose the hyper-parameters achieving high average performance with the smallest variance on the three validation subset.

4.4 Main Results

The experimental results on GLUE benchmark are summarized in Table 2 and Table 3. It is a known problem that the train/dev split for WNLI is correct but somewhat adversarial, and many papers exclude WNLI in their tables, including the original paper of BERT (Devlin et al., 2018). Therefore we do not compare the performance on this dataset (only list for completeness).

As shown in the Table 2, our methods achieve

[6] https://github.com/huggingface/pytorch-pretrained-BERT

GLUE test set results of BERT-large										
Model	Avg	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	WNLI
BERT	80.5	60.5	94.9	89.3/85.4	**87.6**/86.5	72.1/89.3	**86.7**/85.9	92.7	70.1	**65.1**
BERT-r	80.5	60.2	94.9	89.2/85.3	**87.6**/86.5	72.1/89.4	86.5/85.9	92.8	70.1	**65.1**
BERT-SSA-H	**81.2**	**62.7**	**95.8**	**90.2/86.2**	**87.6/86.8**	**72.4/89.5**	86.2/**86.2**	**92.9**	**71.5**	**65.1**

GLUE test set results of RoBERTa-base										
Model	Avg	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	WNLI
RoBERTa-r	81.5	61.1	95.6	91.3/88.5	88.1/87.2	72.2/89.4	87.1/86.3	92.8	73.9	**65.1**
RoBERTa-SSA-H	**82.2**	**62.0**	**96.2**	**92.2/89.3**	**89.6/88.7**	**72.5/89.6**	**87.4/87.0**	**93.1**	**74.9**	**65.1**

GLUE test set results of RoBERTa-large										
Model	Avg	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	WNLI
RoBERTa-r	83.8	63.9	96.3	91.5/88.3	91.6/91.1	72.6/89.7	89.5/**89.6**	94.3	82.3	**65.1**
RoBERTa-bSSA-H	**84.3**	**64.9**	**96.8**	**92.6/90.2**	**91.8/91.2**	**73.5/90.0**	**89.7**/89.5	**94.6**	**82.9**	**65.1**

Table 3: The GLUE results of our model based on BERT-large and RoBERTa.

better results than base BERT consistently. Specifically, **SSA-Co** obtains better results on seven datasets, while keeps on-par performance on the rest datasets. It demonstrates that the auxiliary task of SSA is helpful for model generalization. Moreover, the hybrid model **SSA-Hybrid** performs better on all the datasets and pushes the average score of base BERT (reproduction version) from 78.4 to 79.3. The BERT-mask baseline also shows advantages in several datasets, which can be explained by the effect of data augmentation. However, the decay in other datasets (MRPC, STS-B, QQP and MNLI) indicates that such an improvement is unstable. Our model is obviously more robust, demonstrating the superiority of SSA-based hybrid model over only data augmentation.

We also apply our solutions to more advanced baselines, i.e., BERT-large, RoBERTa-base, and RoBERTa-large. As shown in Table 3, **SSA-Hybrid** consistently outperforms other baselines on different datasets. For BERT-large, it wins on six datasets and leads to a 0.7 absolute increase on the average score. The results further verify the advantages of self-supervised attention layer. Moreover, **SSA-Hybrid** also achieves improvement on RoBERTa, especially on RoBERTa-base. For RoBERTa-large which is one of the strongest single model in the literature, it still leads to a 0.5 absolute increase on the average score. These improvements are impressive under the consideration that no external knowledge is leveraged and only a few extra weights are introduced to the model.

In addition, we discuss gains on different data sizes. In our GLUE classification datasets, RTE, MRPC and CoLA have the smallest training data sizes, and we achieve 1.7%, 2.7% and 3.3% lifts compared to BaseBERT, which is more significant than larger datasets like MNLI. This is reasonable because smaller datasets benefit more from generalization techniques.

4.5 Training Cost Comparison

To further evaluate the extra training cost of SSA paid for the performance improvement, we analyze the training cost on SST-2 dataset. The results are demonstrated in Figure 6. It is understandable that **BERT** and **BERT-m** converge faster than **BERT-SSA-H** in the initial iterations, since SSA-based solution contains more parameters which lead to harder optimization. However, after one epoch, the SSA-based model begins to show its advantage and achieve a lower loss than baseline models with the same training time. Note that when the loss drops each time, the SSA-based model always has a short delay after two baselines, indicating the time cost of label generation. Nevertheless, as shown in the figure, this cost is minor compared with the total training time, and the final performance improvement well deserves the cost.

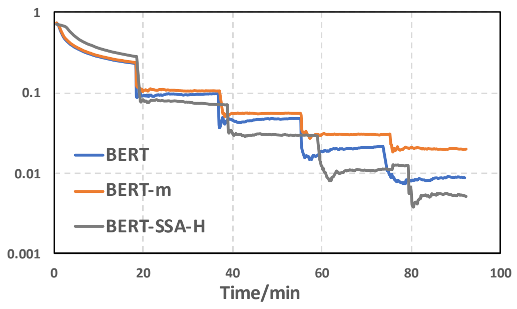


Figure 6: The x-axis represents individual loss value of each epoch scaled using $\log_{10}$ and the y-axis represents training time.

4.6 Sensitivity Study

To investigate the effectiveness of SSA label generation strategy, we conduct experiments on SST-2 under different generation ratio γ. The larger γ means more sentences are generated as training samples. We depict the comparison results in Figure 7. At the beginning, with more sentences generated, all of the three models keep improvement. After γ is greater than 0.8, the performance of BERT-mask degrades dramatically. In contrast, our solution **BERT-SSA-Co** and **BERT-SSA-Hybrid** keep enhancement and converge to better performances. The robustness of our model owes to the self-supervised attention layer which distinguishes relevant words, and a strict label generation strategy that leverages pre-trained knowledge to obtain self-supervised labels.

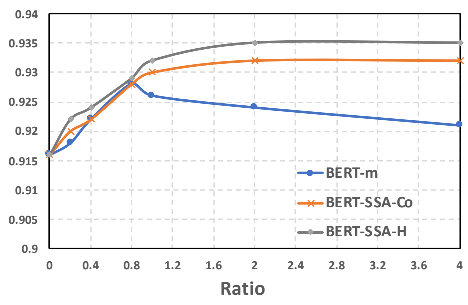
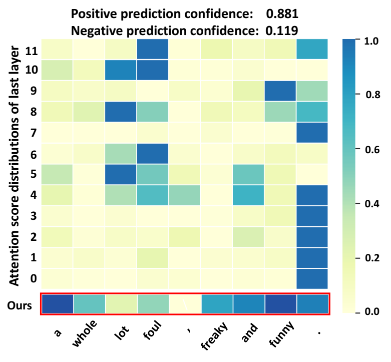


Figure 7: Results of parameter sensitivity on SST-2. The x-axis is the value of γ and the y-axis is the corresponding accuracy.

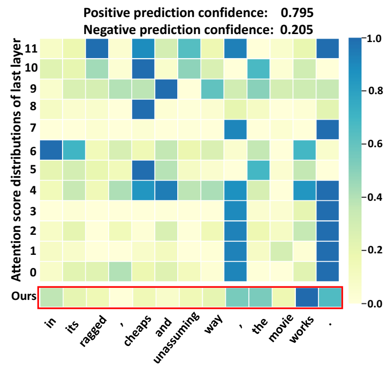
4.7 Case Study

In this section, we visualize two cases to explain why the self-supervised attention layer helps for model improvement, and show the SSA scores learned in the hybrid model are in line with human common-sense. One case is the motivating example, i.e., "a whole lot foul, freaky and funny". As shown in Figure 8(a), the vanilla BERT is misled by the strong negative word 'foul' and gives a wrong prediction. Instead, Our proposed SSA layer identifies 'funny' as the important tokens and put less emphasis on the token 'foul'. The SSA layer correctly capture the relative word importance and generate a better sentence embedding for label prediction. As a result, the sentence can be correctly classified as positive by our SSA-enhanced hybrid model.

Another case is the sentence "in its ragged, cheaps and unassuming way, the movie works.". It is much more difficult than the first case, because there are more negative words which could bias the sentiment prediction result. As shown in Figure 8(b), the vanilla BERT pay much attentions on 'ragged' and 'cheaps' than 'works'. The reason is obvious since 'ragged' and 'cheaps' presents strong



(a) Case 1



(b) Case 2

Figure 8: The multi-head attention scores and SSA score of each word on last layer, obtained by our model fine-tuned on SST.

negative sentiments while 'works' is even not an emotional adjective. However, the self-supervised attention layer is able to identify the actual important tokens, i.e., "works", in such a misleading context. In this way, the final prediction is corrected by the SSA-based solution.

5 Conclusions and Future Work

In this paper, we propose a novel technique called self-supervised attention (SSA) to prevent BERT from overfitting when fine-tuned on small datasets. The hybrid model contains an additional self-supervised attention layer on top of the vanilla BERT model, which can be trained jointly by an auxiliary SSA task without introducing any external knowledge. We conduct extensive experiments on a variety of neural language understanding tasks, and the results demonstrate the effectiveness of the proposed methodology. The case study further shows that the token-level SSA scores learned in the model are in line with human common-sense. In parallel with our research, many recent works focus on data augmentation and multi-task learning. In the next step, we plan to integrate our methodology with more advanced models and evaluate the generality of the proposed solution.

References

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Qian Chen, Zhen-Hua Ling, and Xiaodan Zhu. 2018. Enhancing sentence embedding with generalized pooling. *arXiv preprint arXiv:1806.09828*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

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*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2019. Autosem: Automatic task selection and mixing in multi-task learning. *arXiv preprint arXiv:1904.04153*.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*.

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

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Chi Sun, Luyao Huang, and Xipeng Qiu. 2019. Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence. *arXiv preprint arXiv:1903.09588*.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2018. Conditional bert contextual augmentation. *arXiv preprint arXiv:1812.06705*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.