Unveiling Vulnerability of Self-Attention

Khai Jiet Liong[1,2]*, Hongqiu Wu[1,2], Hai Zhao[1,2]†, Li Xiaoshan[4]*, Bian Minjie[3,4]
[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
[3] Shanghai Technology Innovation Department, Shanghai Data Group Co., Ltd
[4] Shanghai Data Group Co., Ltd
{liongkj, wuhongqiu}@sjtu.edu.cn
zhaohai@cs.sjtu.edu.cn
lixs@sdata.net.cn
bianmj@sdata.net.cn

Abstract

Pre-trained language models (PLMs) are shown to be vulnerable to minor word changes, which poses a big threat to real-world systems. While previous studies directly focus on manipulating word inputs, they are limited by their means of generating adversarial samples, lacking generalization to versatile real-world attack. This paper studies the basic structure of transformer-based PLMs, the self-attention (SA) mechanism. (1) We propose a powerful perturbation technique *HackAttend*, which perturbs the attention scores within the SA matrices via meticulously crafted attention masks. We show that state-of-the-art PLMs fall into heavy vulnerability that minor attention perturbations (1%) can produce a very high attack success rate (98%). Our paper expands the conventional text attack of word perturbations to more general structural perturbations. (2) We introduce *S-Attend*, a novel smoothing technique that effectively makes SA robust via structural perturbations. We empirically demonstrate that this simple yet effective technique achieves robust performance on par with adversarial training when facing various text attackers. Code is publicly available at `github.com/liongkj/HackAttend`.

Keywords: Explainability, Neural language representation models, Semantics

1. Introduction

Pre-trained language models (PLMs), e.g. BERT (Devlin et al., 2019), GPT (Radford et al., 2018), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020), and DeBERTa (He et al., 2021) have demonstrated human-level performances on a series of challenging natural language processing (NLP) tasks, e.g. reading comprehension (Sun et al., 2019) and logical reasoning (Zhou et al., 2020; Yu et al., 2020; Wu et al., 2023a). Yet, despite their impressive capability, studies unveil that such deep neural networks can be easily misled by minor word perturbations, which poses a significant challenge in deploying robust NLP systems.

Augmenting training data with adversarial samples generated by text attack has been proven to be an effective technique to create robust language models. Existing attackers generate the adversarial samples by manipulating the input text, e.g. by word substitution, swapping or insertion (Jin et al., 2020). Nevertheless, these defense methods are limited by the means of the attack and have limitations in their effectiveness against attacks in more general situations, such as word substitutions beyond synonyms or semantics. Moreover, incorporating such adversarial samples into training always results in a large degradation of the performance.

Our key vision is that the vulnerability of a language model derives from its architecture and inner mechanism. By examining the vulnerability of the inner mechanism, it is possible to gain insights into why language models are susceptible to input perturbations. In this paper, we specifically investigate the self-attention (SA) mechanism in the context of model robustness, an area that has received limited attention despite its fundamental role in PLMs.

To examine the vulnerability of SA in PLMs, We first propose a novel perturbation strategy *HackAttend*. Unlike previous attack methods which focus on the **input words**, our algorithm perturbs the SA weights within the **SA matrices** to trigger the model to a wrong prediction.

Figure 1 illustrates how *HackAttend* generates adversarial samples in the form of custom attention mask which successfully disrupt SA mechanism's ability to capture contextual information effectively. Empirical experiments were conducted on a wide range of tasks, including reading comprehension, logical reasoning, and sentiment analysis, and natural language inference. Our results demonstrate that state-of-the-art language models are heavily vulnerable to *HackAttend*, achieving high attack

* Equal Contribution

† Corresponding author; This paper was partially supported by Joint Research Project of Yangtze River Delta Science and Technology Innovation Community (No. 2022CSJGG1400).

LREC-COLING 2024, pages 17225–17236

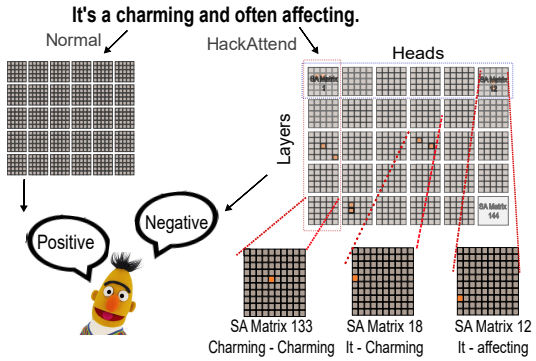


Figure 1: Illustration of the process of *HackAttend* manipulating the SA mechanism. The perturbation of SA units (highlighted in orange) demonstrates how the algorithm induces misclassification in sentiment analysis by flipping the activation states. The example perturbation shows the transition from a positive to a negative sentiment interpretation, providing a visual representation of *HackAttend*'s effect on the model's decision-making process for the SST-2 dataset.

success rate with only minor SA perturbations.

Building upon our findings, we then propose a novel smoothing technique *S-Attend*, which smooths the attention scores during training. Since our technique perturbs the model structure, resulting in no bias to input distribution, it unlocks impressive robustness outcome with minor performance compromise.

Our contributions are summarized below:

- We are the first to discuss the perturbations and smoothing technique orienting SA.
- We analyze the impact of different perturbation levels in the SA matrices on downstream tasks, providing insights into the role of SA in capturing task-specific signals.
- We propose a new smoothing technique to defend against general attacks.

2. Related Work

Our work introduces a novel perturbation strategy on the SA component. Similar to the end goal of text adversarial attack algorithms, the objective of *HackAttend* perturbation is to induce misclassifications in language models. Examples of text adversarial attacks includes character level attacks (e.g. TextBugger (Gao et al., 2018), DeepWordBug (Li et al., 2019), HotFlip (Ebrahimi et al., 2018)) and the word level (e.g. TextFooler (Jin et al., 2020), BERT-Attack (Li et al., 2020), SemAttack (Wang et al., 2022), PWWS (Ren et al., 2019), and BBA (Lee et al., 2022). While these methods pay attention on the scope of input text and word embedding, our research, in contrast, places emphasis on the architecture of the model itself. Our goal is not to propose a new algorithm for real-world adversarial attacks, but rather to gain a deeper understanding of this unexplored area.

Our work focuses on the fundamental architecture, self-attention (SA) mechanism (Vaswani et al., 2017) of PLMs such as `RoBERTa` (Liu et al., 2019), `ALBERT` (Lan et al., 2020), and `DeBERTa` (He et al., 2021). SA has been extensively studied in the literature (Shi et al., 2021; You et al., 2020; Zhang et al., 2020b). Wu and Zhao (2022) initially discuss the robustness of SA, highlighting the model's tendency to overemphasize certain spurious keywords while ignoring overall semantics. Drawing upon this observation, we leverage this insight to develop our perturbation strategy.

In previous literature, various metrics have been employed to impose constraints on perturbation levels in order to minimize perceptibility between original and adversarial samples. Metrics such as Jaccard similarity, cosine similarity, Earth Mover's Distance (Yu and Herman, 2005), MoverScore (Zhao et al., 2019) and BERTScore (Zhang et al., 2020a) have proven effective in evaluating semantic similarity of the input. However, the Hamming Distance provides an alternative perspective in our evaluation, allowing us to control perturbation levels at a structural level.

It is worth noting that adversarial training is usually adopted as the means to enhance the robustness of models (Goodfellow et al., 2015; Zhu et al., 2020; Wu et al., 2023c). In contrast to prior research, we distinguish ourselves by removing the adversarial nature and presenting an efficient smoothing technique in improving model robustness in our study. Our method employs random masking of attention units, contrasting with conventional techniques that mask whole unimportant attention heads during training (Budhraja et al., 2020) and inference (Cao and Wang, 2021), or adding an extra layer of complexity (Fan et al., 2021). Through empirical evidence, we demonstrate the substantial advantages of incorporating structural perturbations to achieve comprehensive and resilient robustness against diverse attacks.

3. Introduction to HackAttend

This section elaborates on our proposed *HackAttend* perturbation design. *HackAttend* distinguishes itself from conventional text attacks as our main purpose is for interpretability of the robustness of language models.

3.1. Notations

We first define the notations used in the methodology:

α	Masking percentage

N_L / N_H	Number of layers / heads in the victim model
l_{max} / h_{max}	Maximum number of layers / heads perturbed
$SA_{i,j,k,l}$	The k^{th}-row l^{th}-column attention unit in the SA matrix of the j^{th} head and i^{th} layer
M / M'	Original / adversarial attention mask
S_L / S_H	Importance score of a layer or head

3.2. Overview of Hackattend

A PLM consists of multiple SA layers, each with varying numbers of attention heads (also referred as SA matrices). Each cell within these matrices represents a SA unit. For example, a `BERT-base` model typically includes 12 (layers) x 12 (heads) attention matrices. Our proposed method, *HackAttend* efficiently identifies adversarial samples across various model configurations by employing a greedy algorithm approach.

The overall objective of the *HackAttend* algorithm is to rank and assess the significance of every SA matrices in the model based on their eventual impact the final model prediction (discussed in Section 3.3. We employed a novel gradient-based technique (details in Section 3.4) to prioritize the most significant SA units within the candidate SA matrices. We will then apply a masking operation on the prioritized SA units within a constraint requirement defined in the Section 3.5 to ensure minimal perturbation on the attention mask while effectively exploring the search space.

Unlike traditional adversarial samples, *HackAttend* generates an adversarial structure (in the form of custom attention mask) which perturbs the underlying attention calculation.

3.3. Layers and Head Selection

Layers Selection In our approach, we assess each layer's importance in the model by iteratively masking all attention heads within that layer. If masking a specific layer, denoted as the i-layer, results in an incorrect model prediction, this layer is deemed highly important and assigned a top importance score. For cases where the model still predicts correctly, layer importance is determined by the reduction in the model's output probability. A greater decrease in this probability signifies a higher importance during prediction.

Heads Selection Similar to the process for layer selection, we use a comparable strategy to evaluate the importance of individual heads, denoted as H_j, where each H_j corresponds to a specific head in the model's i^{th} layer. We systematically mask the heads in the i^{th} layer, one at a time, and calculate the model's output probability, resulting in the head importance score $S_H[i, j]$. The scores were ranked in descending order, prioritizing the more vulnerable head as the target for the perturbation.

3.4. SA Units Selection

To determine the SA units that disrupt the model's attention mechanism, we utilize a gradient-based algorithm for ranking the SA units, inspired by the work of Wu and Zhao (2022) where the underlying premise is SA units with larger gradients has more significant impact on model predictions. By masking these SA units in the worse-case direction, we maximizes the empirical training risk. Our process

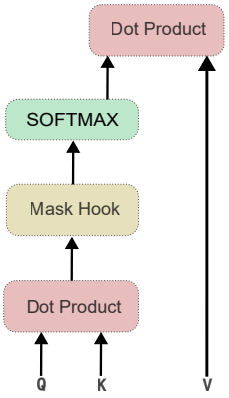


Figure 2: Placement of mask hook

involves the use of original attention mask (M), where no attention units are initially masked at the start of each forward step. We then compute the gradients of the loss function with respect to the M, and rank the SA units based on their gradients in the backward step.

To obtain the gradients of SA units during the backward pass, we add a custom backward hook in the encoder layer, illustrated in Figure 2, to capture the gradients flowing through M. These modifications are applied to the original SA layer, specifically just before the softmax operation, while keeping the remaining components unchanged.

Nevertheless, it has come to our attention that specific attention heads may generate very small gradients in certain samples such that ranking with gradients is impractical, as these gradients may essentially become zero across all units. As a fallback strategy, the attention scores obtained during the forward pass will serve as a proxy score as proposed by Michel et al. (2019). Furthermore, an ablation study comparing both ranking strategies can be found in Subsection 6.2.

3.5. Constraints

In the final step of the algorithm, the attention mask M with the highest importance rankings within the candidate SA matrices is perturbed, preventing the model from attending to those units during computations. In this process, masking involves expanding the original attention mask, which has dimensions of `[seq_len, seq_len]`, to create a modified mask with dimensions of `[num_layers, num_head, seq_len, seq_len]`, and selectively deactivating specific SA units.

To ensure the learning process remains intact, a constraint is imposed on the percentage of tokens to be perturbed. This constraint guarantees the high similarity between M and M', thereby preserving the overall model's functionality while maintaining an optimal adversarial approach.

To implement the token masking constraint, the parameter masking percentage, denoted as α, is introduced. It is assigned a value of 0.01, representing 1.0% of the non-padding tokens. For example, let's consider a SA matrix with 225 units (a 15×15 matrix), and the goal is to mask 1% of the units. In this case, the top 2 units with the highest gradients would be selected and denoted as SA_{i,j,k_1,l_1} and SA_{i,j,k_2,l_2}, where (k_1, l_1) and (k_2, l_2) represent the indices of the top 2 sorted SA units. M' serves as a replacement for the original attention mask before it was fed into the victim model.

3.6. Algorithm

HackAttend employs a greedy approach to generate adversarial attention mask by ranking all SA matrices. It iteratively selects the most vulnerable layers and heads based on their importance scores and searches for important SA units based on gradients. The resulting adversarial structure, denoted as M', is then assessed whether it could induce misclassification of the victim model. The algorithm, with its greedy decision-making characteristic, is presented in Algorithm 1.

Algorithm 1 *HackAttend*

Input: Maximum number of layers perturbed (l_{max}), Maximum number of heads perturbed (h_{max}), Victim model (f), Masking percentage (α)

Select a sample $\{x, y\}$
Go forward step and obtain the gradients
$S_L \leftarrow$ Rank all layers based on importance scores
for $l_i \leftarrow 1$ **to** l_{max} **do**
  $S_H \leftarrow$ Rank the heads in the $S_L[l_i]^{\text{th}}$ layer based on their importance scores
  for $h_j \leftarrow 1$ **to** h_{max} **do**
    $M' \leftarrow$ Mask the units in $S_H[h_j]^{\text{th}}$ head with top α largest gradients
    $\hat{y} \leftarrow f(x, M')$
    if $\hat{y} \neq y$ **then**
      return SUCCESS
    end if
  end for
end for
return FAIL

4. Empirical Experiment

This section reports our empirical results.

4.1. Evaluation Settings

We evaluate the effectiveness of the perturbations and its impact using the following metrics:

Hamming Distance This metric quantifies the extent of perturbations to the SA matrix. Since the attention matrix is a binary matrix, Hamming distance measures the number of bits that differ between M and M'. We average the Hamming distance across each perturbed SA matrix as follows:

$$d_H(M, M') = \frac{\sum_{i=1}^{N_{SA}} (M_i \oplus M'_i)}{N_{SA}}$$

where $\oplus$ represents the XOR (exclusive or) operation, N_{SA} is the number of SA matrices perturbed.

Clean Accuracy This metric is used to measure the accuracy score on the clean set.

Robust Accuracy This metric is used to measure the accuracy score under attack/perturbation.

Attack Success Rate (ASR) This metric is used to measure the perturbation's success rate when the model makes an incorrect prediction after the perturbation. The ASR is computed as follows:

$$ASR = \frac{\text{\# successful perturbation}}{\text{\# correct predictions}}$$

An effective perturbation algorithm has the following properties: 1) a low Hamming distance, indicating high degree of similarity and the perturbation is cunning yet minimally affects the attention structure; 2) a high ASR and/or low robust accuracy is desired, indicating the effectiveness successfully inducing misclassifications;

4.2. Dataset

For our experiments, we choose four representative NLP tasks:

• **Sentiment Analysis**: Stanford Sentiment Treebank (SST-2) (Socher et al., 2013);

• **Natural Language Inference (NLI)**: HELLASWAG (Zellers et al., 2019), a multiple-choice common-sense reasoning dataset;

• **Dialogue Comprehension**: Dialogue-based machine reading comprehension (DREAM) (Sun et al., 2019), in a multiple-choice format;

• **Logical Reasoning**: ReClor (Yu et al., 2020), a machine reading comprehension in a multiple-choice format.

Dataset	Mean	Max	Min
DREAM	75.3	128.0	24.0
HELLASWAG	86.7	128.0	19.0
RECLOR	125.1	128.0	66.8
SST-2	25.2	55.0	4.0

Table 1: Sequence length of the Test/Dev split.

4.3. Setup

Victim Models For our experiments, we selected `BERT-base` as the victim model. We initially fine-tuned this pretrained model on a target dataset (training details can be found in Appendix), followed by perturbation experiments using the same dataset on a single NVIDIA TITAN RTX 24G.

In the *HackAttend* implementation, we employ two configurations to limit the number of layers/heads that the algorithm can target. The full-scale setting uses parameters h_{max} and l_{max} set to 12, while the half-scale is set to 6, thereby reducing the search space. Note that these settings do not imply that all layers/heads are perturbed simultaneously. The attention mask is related to real sequence length of the input text i.e. the non-pad tokens. With the same mask percentage α, longer sequences results in more SA units being masked. Hence, the statistics of sequence length is shown in Table 1.

4.4. Results

Main Results Table 2 demonstrates the overall results of *HackAttend* across various tasks. The introduced perturbations demonstrate its effectiveness, achieving an ASR of at least 98.9% in 3 out of 4 tasks and reducing the clean accuracy on DREAM ($\sim$ 64%), RECLOR ($\sim$ 52%), and HELLASWAG ($\sim 39\%$) under the full-scale setting. *HackAttend* is sub-optimal in the case of SST-2 in both configurations, yielding ASR of 27.4% and robust accuracy drop ($\sim 26\%$) and 10.2% ASR, robust accuracy drop ($\sim 10\%$), respectively.

We hypothesize that this discrepancy can be attributed to the varying degrees of the reliance on SA layer across different tasks. For example, in simple tasks like sentiment analysis, the model depends on a combination of linguistic features and local keywords, making them less sensitive to perturbations. In contrast, tasks involving complex question answering and story comprehension heavily rely on attention mechanisms, as demonstrated empirically in Subsection 4.4.

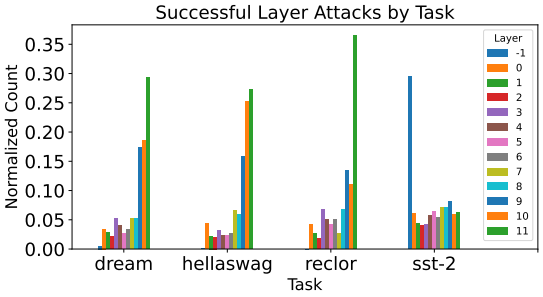


Figure 3: Normalized count of successful perturbation grouped by layers chosen and task. `-1` indicates a failed perturbation.

Results on Perturbation Strength Notably, in the half-scale setting, the gradient-based algorithm remains significant in most of the datasets. On DREAM, the drop in ASR% was only marginal, measuring at 7.7%. On HELLASWAG, the drop was even smaller, at 3.2%. The RECLOR dataset exhibited minimal decrease in ASR, with only 0.4% reduction. These findings indicate that half-scale setting is effective, but perturbations under the full-scale setting yield more promising results.

Results on Different Components Table 3 provides a summary of baseline experiments, encompassing both the performance of the *HackAttend* algorithm and the impact of each component. For our baseline, we employ random selection for layers, heads, and SA units. The results show that incorporating importance scores indeed enhances the perturbation efficiency (lesser number of queries). However, empirically, we find that gradient alone is sufficient to identify vulnerable SA units (Wu and Zhao, 2022).

Vulnerability of Different Layers Figure 3 offers valuable insight into the vulnerability of various layers. It shows the frequency of each layer was selected for perturbation. The counts are normalized to provide relative comparison of the frequencies of successful perturbations across different layers and tasks. Higher layers, i.e. the 12$^{\text{th}}$ layer, is notably more sensitive to *HackAttend*, given its responsibility in managing long-range dependencies and global context.

This consistency applies to all tasks except SST-2, where all layers exhibit similar performance. This is because sentiment analysis tasks rely heavily on local context, like keywords features, which is often sufficient to yield accurate predictions.

Dataset	Max N	ASR%	*clean*%	*robust*%	# Query	Hamming
DREAM	12	98.9	64.7	0.7	18.6	611.4
	6	91.2		5.7	11.2	618.2
HellaSWAG	12	99.9	39.6	0.0	8.8	1222.2
	6	96.7		1.3	7.1	1232.8
ReClor	12	100.0	51.8	0.0	7.3	2151.3
	6	99.6		0.2	6.5	2153.7
SST-2	12	27.4	93.9	67.8	123.6	9.3
	6	10.2		83.8	34.1	9.5

Table 2: Max N indicates the maximum number of candidate heads/layers and α= 1.0% and the results for DREAM, HELLASWAG, RECLOR, and SST-2 are reported on the dev set.

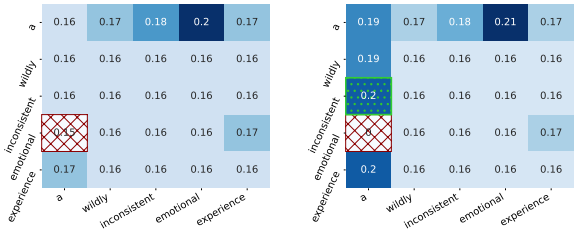
Setting	ASR%	# Query	Hamming
DREAM			
Baseline	$61.7_{\pm 1.31}$	$76.4_{\pm 1.25}$	$565.3_{\pm 79.20}$
Imp. w/o Grad	$64.6_{\pm 0.14}$	$65.8_{\pm 0.07}$	$562.4_{\pm 3.34}$
Grad w/o Imp.	$98.9_{\pm 0.00}$	$20.0_{\pm 0.09}$	$611.9_{\pm 0.33}$
HackAttend	98.9	18.6	611.4
HellaSwag			
Baseline	$88.4_{\pm 0.06}$	$39.9_{\pm 0.18}$	$1191.6_{\pm 2.17}$
Imp. w/o Grad	$88.8_{\pm 0.02}$	$33.5_{\pm 0.01}$	$1195.6_{\pm 0.38}$
Grad w/o Imp.	$99.9_{\pm 0.00}$	$8.3_{\pm 0.01}$	$1221.9_{\pm 0.13}$
HackAttend	99.9	8.8	1222.2
ReClor			
Baseline	$77.9_{\pm 0.10}$	$61.2_{\pm 1.45}$	$2123.4_{\pm 203.71}$
Imp. w/o Grad	$77.6_{\pm 0.70}$	$47.4_{\pm 0.55}$	$2109.3_{\pm 21.85}$
Grad w/o Imp.	$100.0_{\pm 0.00}$	$5.5_{\pm 0.00}$	$2151.3_{\pm 0.00}$
HackAttend	100.0	7.3	2151.3
SST-2			
Baseline	$0.6_{\pm 0.01}$	$143.4_{\pm 0.00}$	$9.9_{\pm 2.29}$
Imp. w/o Grad	$0.6_{\pm 0.00}$	$143.6_{\pm 0.00}$	$8.5_{\pm 0.33}$
Grad w/o Imp.	$27.3_{\pm 0.00}$	$124.8_{\pm 0.01}$	$9.3_{\pm 0.00}$
HackAttend	27.4	123.6	9.3

Table 3: Experimental results for various tasks (mean $\pm$ variance over three seeds). "Baseline" indicates random layer, head and SA units selection, while "*Imp.*" represent layer and head selection using Importance Score. "*Grad*" represent SA unit selection using gradient.

4.5. Case Study

The functionality of *HackAttend* is illustrated through a successful example from the SST-2 dataset, where the ground truth label is Negative (Figure 4a). *HackAttend* masks the SA unit corresponding to the word pair "*a-emotional*" based on gradients (labeled in red). This redirects the model's attention from the more relevant word pair "*a-emotional*" to "*a-inconsistent*" (labeled in green), effectively deceiving the model into focusing on a less informative feature (Figure 4b).

While "*a-inconsistent*" contributes to understanding of global semantics, the word pair "*a-emotional*" contains local semantics in the form of **emotional cues**. *HackAttend* effectively misleads the model by selectively suppressing these cues through attention scores manipulation.



(a) Original (Negative) (b) *HackAttend* (Positive)

Figure 4: Comparison of attention maps on the 2nd layer, 10th head of `BERT-base` before and after perturbation. The sample is "*A wildly inconsistent emotional experience.*" from SST-2. The sentence is classified as positive after the perturbation.

5. Efficiency Analysis

We choose two representative text attackers BERT-Attack (BA) (Li et al., 2020) and TextFooler (TF) (Jin et al., 2020) from TextAttack[1] on DREAM, RECLOR and SST-2 where the benchmark results on the first 100 successful perturbations are summarized in Table 4.

For complex tasks, *HackAttend* outperforms its counterparts in generating adversarial samples with higher efficiency. Unlike the other two algorithms, which require more time as input sequences grow longer, *HackAttend* maintains relatively consistent efficiency across different sequence lengths, with the exception of SST-2.

6. Ablation Study

All experiments use the same hyperparameter as in Section 4 unless otherwise specified.

[1] https://github.com/QData/TextAttack

Dataset	Perturbation/Attack	Avg Time (s)	# Query
DREAM	*HackAttend* (ours)	**0.7**	**13.1**
	TextFooler	0.9	90.7
	BERT-Attack	0.8	64.3
ReClor	*HackAttend* (ours)	**0.6**	**7.3**
	TextFooler	1.9	170.4
	BERT-Attack	2.0	103.8
SST-2	*HackAttend* (ours)	23.8	67.2
	TextFooler	**0.4**	92.5
	BERT-Attack	1.0	**40.0**

Table 4: Efficiency Analysis - Average Time and Query Count for success samples. Our method outperforms TextFooler and BERT-Attack in terms of execution time and query efficiency. Candidate size (k) for BERT-Attack is set to 10.

6.1. Effect of Masking Percentage

We evaluate the effect of mask percentage (α) by comparing α={1%, 0.1%, 0.01%}. Table 5 demonstrates that ASR maintains reasonable high, DREAM ($\sim 72\%$), HELLASWAG ($\sim 92.5\%$) and RECLOR ($\sim 84.9\%$) even with α as low as 0.01%. This suggests that selection of SA unit using gradients is effective in generating adversarial samples across various datasets and level of masking.

Dataset	Mask%	ASR%	Hamming	# Query
DREAM	1.00	98.9	611.4	18.6
	0.10	91.2	62.4	36.3
	0.01	72.7	5.7	58.9
HellaSWAG	1.00	99.9	1221.2	8.8
	0.10	98.9	121.6	17.6
	0.01	92.5	11.5	29.7
SST-2	1.00	27.4	9.3	123.6
	0.10	6.4	1.1	139.0
	0.01	6.4	1.0	139.1
ReClor	1.00	100.0	2151.3	7.3
	0.10	100.0	213.3	15.9
	0.01	84.9	19.4	40.7

Table 5: Impact of masking percentage on performance metrics across different tasks.

6.2. Effect of Ranking for SA unit

Table 6 shows both ranking by score and by negative gradient can achieve 100% ASR, with some trade-off on number of queries and per-query time. Indeed, the gradients pointing in the steepest descent direction have significant impact on the loss function, are efficient in identifying vulnerable SA units. Masking them will more likely contribute to errors in the model's predictions.

Alternatively, the attention score, assigned as a normalized value, shows the relevance of each SA unit to the model's predictions. However, since the attention score evaluates each unit individually, it may require perturbing more SA matrices to disrupt the necessary path for accurate predictions.

Ranking	ASR%	Hamming	# Query	Avg Time (s)
Gradient	100	2151.3	**7.3**	0.11
Score	100	2151.3	22.8	**0.08**

Table 6: Performance metrics reported on RECLOR for different Ranking Strategies ($\alpha = 1.0\%$). Score represent Attention Score.

7. HackAttend Inspired Smoothing

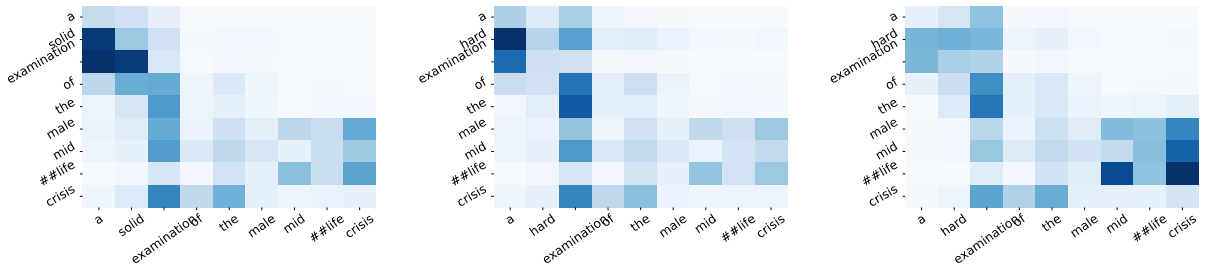
In this section, we explore *S-Attend*, a smoothing technique to improve model robustness performance, which is particularly effective in situations where attacker diversity or characteristics are not fully known.

Adversarial training (AT) (Goodfellow et al., 2015) is proven to be an effective defense technique by augmenting the training data with adversarial samples. However, storing such adversarial structures (e.g. for multiple attention heads/layers) slows down training or requires large amount of storage. Rather, we propose an efficient technique - randomly smoothing the attention weights during training - *S-Attend*. Concretely, we randomly apply masking on attention weights following a parametric Bernoulli distribution with α = {0.1,0.2,0.5} for all heads.

Table 7 presents the performance of baseline BERT model, *S-Attend*, adversarial training techniques on BERT Model (CreAT (Wu et al., 2023b) & FreeLB (Zhu et al., 2020)) and ADA (adversarial data augmentation) technique against BERT-Attack (BA) and TextFooler (TF) adversarial attacks. ADA models refer to BERT models trained on datasets augmented by specific attack algorithms, subsequently evaluated for their defensive effectiveness against those attacks. The column "*clean%*" and "*robust%*" represent evaluation result on clean dataset and adversarial augmented test sets using respective attackers, respectively.

While this smoothing technique is relatively inexpensive and straightforward, empirical experiments demonstrates that *S-Attend* trained model has shown comparable or even superior robustness performance in specific tasks when compared to the regular adversarial training method, often with little to no compromise to clean performance.

In DREAM, *S-Attend*† outperformed BA(ADA) with a 1.2% increase in robust accuracy and a marginal 0.3% drop in clean accuracy compared to TF(ADA) while maintaining competitive clean ac-



(a) Normal model (clean) (b) Normal model (augmented) (c) S-Attend model (augmented)

Figure 5: Comparison of attention maps for the text *a solid examination of the male midlife crisis.* from SST-2 on the 10th layer, 4th head of `BERT-base` between normal model (on clean and BERT-Attack augmented dataset), and *S-Attend* model (on augmented dataset). After the adversarial attack, the normal model misclassifies as negative, while *S-Attend* model correctly maintains as positive.

Dataset	Defense/Smoothing	*clean*%	*robust*% TF	BA
ReClor	Baseline	51.8	0.8	2.0
	CreAT	49.0	46.6	48.0
	FreeLB	50.4	50.2	49.6
	TF(ADA)	47.8	47.4	47.8
	BA(ADA)	47.4	47.0	46.6
	S-Attend ($\alpha = 0.1$)	48.6	47.4	47.8
	S-Attend ($\alpha = 0.2$)	51.0	50.0	49.6
	S-Attend† ($\alpha = 0.5$)	**52.8**	**51.4**	**51.2**
DREAM	Baseline	64.7	19.3	3.8
	CreAT	65.0	55.1	**55.2**
	FreeLB	**65.1**	**56.2**	55.1
	TF(ADA)	57.4	55.6	54.1
	BA(ADA)	55.7	51.9	52.5
	S-Attend ($\alpha = 0.1$)	63.2	54.2	52.8
	S-Attend† ($\alpha = 0.2$)	64.4	54.6	53.7
	S-Attend ($\alpha = 0.5$)	63.0	53.0	52.8

Table 7: Robust Performance Evaluation: Baseline model vs. Adversarial Training models vs. ADA models. Baseline represents regular fine-tuned BERT base. † denotes the best *S-Attend* Model.

curacy. In ReClor, the *S-Attend*† model stands out prominently with remarkable performance. It boosts clean accuracy by 1.0% compared to the baseline, along with reducing robust accuracy by 0.4% and 0.6% in both ADA test sets.

The effectiveness of the smoothing mechanism employed by *S-Attend* can be attributed to two key factors. 1)During training, *S-Attend* promotes the activation of various components, which in turn aids in reducing sensitivity to noisy input data. 2) Attention weight perturbation during training helps the model to learn a more generalized representation, capturing global semantics and intricate word relationships. Fundamentally, this significantly enhances the model's robustness (Wu et al., 2020).

7.1. S-Attend Against HackAttend

To further validate *S-Attend*'s effectiveness, we compared various methods against *HackAttend* perturbations, as seen in Table 8. These results highlight our model's robustness against a broad spectrum of attacks, be it conventional adversarial attacks or perturbations like *HackAttend*. Notably, the smoothing technique can either reduce ASR% or, at the very least, increase query counts.

Both CreAT and FreeLB exhibits limited effectiveness against *HackAttend*, specifically on ReClor. In the other hand, our simple yet straightforward smoothing technique effectively mitigates this limitation. However, it's worth noting that the ASR% of *HackAttend* remains high after smoothing, indicating the SA component still presents vulnerabilities.

Dataset	Method	Mask%	ASR%
ReClor	Baseline	1.00	100.0
	S-Attend		100.0
	CreAT		100.0
	FreeLB		100.0
	Baseline	0.10	99.6
	S-Attend		**87.1**
	CreAT		100.0
	FreeLB		100.0
DREAM	Baseline	1.00	98.9
	S-Attend		**97.5**
	CreAT		100.0
	FreeLB		98.9
	Baseline	0.10	91.2
	S-Attend		**85.1**
	CreAT		90.0
	FreeLB		88.3

Table 8: Robustness evaluation against *HackAttend* perturbations. Adversarial Training vs. *S-Attend* smoothing. Baseline represents regular fine-tuned model.

7.2. Case Study

The aftereffects of applying *S-Attend* could be demonstrated through a successful scenario where our model correctly maintains its predictions even when subjected to adversarial attacks. As an example from SST-2 (Figure 5), the attention heatmap indicates that the words "*solid*" and "*examination*" are pivotal in the model's positive sentiment prediction. However, a synonym word swap ("*solid*"→"*hard*") by BERT-Attack causes misclassification due to spurious patterns, which suggest that models tend to heavily rely on word matching (Hao et al., 2021).

In contrast, the *S-Attend* model showcases its remarkable capability and adaptability in effectively focusing on critical cues like "*midlife*" and "*crisis*". It demonstrates its ability to shift its attention, when confronted with potentially misleading word substitutions, such as synonyms and semantically related terms, while maintaining predictive accuracy in challenging linguistic scenarios.

8. Conclusion

In this paper, we presented *HackAttend*, a method for perturbing SA-based PLMs. Our experiments demonstrated its effectiveness in generating adversarial structures, particularly on complex tasks. *HackAttend* achieves high success rates with minimal perturbation. Additionally, we proposed *S-Attend*, a smoothing technique that enhances the model SA structure with minimal impact on training time, while exhibiting competitive performance against other adversarial training.

Limitations

This paper only studies on encoder-only architecture. Firstly, extending the applicability of *HackAttend* to other model architectures, such as encoder-decoder architectures like T5 (Raffel et al., 2020) or decoder-only architectures like GPT (Radford et al., 2018) are not studied. We did not evaluate *HackAttend* on large language models due to the resource limitations.

9. Bibliographical References

Aakriti Budhraja, Madhura Pande, Preksha Nema, Pratyush Kumar, and Mitesh M. Khapra. 2020. On the weak link between importance and prunability of attention heads. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3230–3235. Association for Computational Linguistics.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Boxin Wang, Chejian Xu, Xiangyu Liu, Yu Cheng, and Bo Li. 2022. Semattack: Natural textual attacks via different semantic spaces. In *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 176–205. Association for Computational Linguistics.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. Freelb: Enhanced adversarial training for natural language understanding. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Deokjae Lee, Seungyong Moon, Junhyeok Lee, and Hyun Oh Song. 2022. Query-efficient and scalable black-box adversarial attacks on discrete sequential data via bayesian optimization. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 12478–12497. PMLR.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press.

Dongxian Wu, Shu-Tao Xia, and Yisen Wang. 2020. Adversarial weight perturbation helps robust generalization. In *Advances in Neural Information*

Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual.

Han Shi, Jiahui Gao, Xiaozhe Ren, Hang Xu, Xiaodan Liang, Zhenguo Li, and James Tin-Yau Kwok. 2021. Sparsebert: Rethinking the importance analysis in self-attention. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 9547–9557. PMLR.

Hongqiu Wu and Hai Zhao. 2022. Adversarial self-attention for language understanding. *CoRR*, abs/2206.12608.

Hongqiu Wu, Linfeng Liu, Hai Zhao, and Min Zhang. 2023a. Empower nested boolean logic via self-supervised curriculum learning. *arXiv preprint arXiv:2310.05450*.

Hongqiu Wu, Yongxiang Liu, Hanwen Shi, Hai Zhao, and Min Zhang. 2023b. Toward adversarial training on contextualized language representation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Hongqiu Wu, Yongxiang Liu, Hanwen Shi, hai zhao, and Min Zhang. 2023c. Toward adversarial training on contextualized language representation. In *The Eleventh International Conference on Learning Representations*.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 31–36. Association for Computational Linguistics.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops, SP Workshops 2018, San Francisco, CA, USA, May 24, 2018*, pages 50–56. IEEE Computer Society.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. Textbugger: Generating adversarial text against real-world applications. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society.

Kai Sun, Dian Yu, Jianshu Chen, Dong Yu, Yejin Choi, and Claire Cardie. 2019. DREAM: A challenge dataset and models for dialogue-based reading comprehension. *Trans. Assoc. Comput. Linguistics*, 7:217–231.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6193–6202. Association for Computational Linguistics.

Ming Zhou, Nan Duan, Shujie Liu, and Heung-Yeung Shum. 2020. Progress in neural nlp: Modeling, learning, and reasoning. *Engineering*, 6(3):275–290.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani B. Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2890–2896. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14014–14024.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In

9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. 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, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4791–4800. Association for Computational Linguistics.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1085–1097. Association for Computational Linguistics.

Shuyang Cao and Lu Wang. 2021. Attention head masking for inference time content selection in abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5008–5016. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020a. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3428–3448. Association for Computational Linguistics.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020. Reclor: A reading comprehension dataset requiring logical reasoning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Weiqiu You, Simeng Sun, and Mohit Iyyer. 2020. Hard-coded gaussian attention for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7689–7700. Association for Computational Linguistics.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 563–578. Association for Computational Linguistics.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2021. Self-attention attribution: Interpreting information interactions inside transformer. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12963–12971. AAAI Press.

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. *CoRR*, abs/1907.11692.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1298–1308. Association for Computational Linguistics.

Zhenghua Yu and Gunawan Herman. 2005. On the earth mover's distance as a histogram similarity metric for image retrieval. In *Proceedings of the 2005 IEEE International Conference on Multimedia and Expo, ICME 2005, July 6-9, 2005, Amsterdam, The Netherlands*, pages 686–689. IEEE Computer Society.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Zhihao Fan, Yeyun Gong, Dayiheng Liu, Zhongyu Wei, Siyuan Wang, Jian Jiao, Nan Duan, Ruofei Zhang, and Xuanjing Huang. 2021. Mask attention networks: Rethinking and strengthen transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1692–1701. Association for Computational Linguistics.

Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. 2020b. Sg-net: Syntax-guided machine reading comprehension. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9636–9643. AAAI Press.

10. Appendices

Fine-tuning Details

	LR	BSZ	EP	WP	MSL
DREAM	$3e-5$	16	8	0.1	128
HellaSWAG	$2e-5$	32	3	0.1	128
ReClor	$2e-5$	24	6	0.06	128
SST-2	$2e-5$	32	3	0.06	128

Table 9: Suggested fine-tuning setting. LR: learning rate; BSZ: batch size; EP: training epochs; WP: warmup proportion; MSL: sequence length;