Training NLI Models Through Universal Adversarial Attack

Jieyu Lin, Wei Liu, Jiajie Zou, Nai Ding *
Key Laboratory for Biomedical Engineering of Ministry of Education,
College of Biomedical Engineering and Instrument Sciences, Zhejiang University, Hangzhou, China
{jieyu_lin,liuweizju,jiajiezou,ding_nai}@zju.edu.cn

Abstract

Pre-trained language models are sensitive to adversarial attacks, and recent works have demonstrated universal adversarial attacks that can apply input-agnostic perturbations to mislead models. Here, we demonstrate that universal adversarial attacks can also be used to harden NLP models. Based on NLI task, we propose a simple universal adversarial attack that can mislead models to produce the same output for all premises by replacing the original hypothesis with an irrelevant string of words. To defend against this attack, we propose Training with UNiversal Adversarial Samples (TUNAS), which iteratively generates universal adversarial samples and utilizes them for fine-tuning. The method is tested on two datasets, i.e., MNLI and SNLI. It is demonstrated that, TUNAS can reduce the mean success rate of the universal adversarial attack from above 79% to below 5%, while maintaining similar performance on the original datasets. Furthermore, TUNAS models are also more robust to the attack targeting at individual samples: When search for hypotheses that are best entailed by a premise, the hypotheses found by TUNAS models are more compatible with the premise than those found by baseline models. In sum, we use universal adversarial attack to yield more robust models.

1 Introduction

Pre-trained models have achieved impressive performance among natural language processing (NLP) tasks, including natural language inference (NLI) and machine reading comprehension (MRC) (Liu et al., 2019; He et al., 2020). Nevertheless, these models are vulnerable under adversarial attacks (Behjati et al., 2019). For most adversarial attack methods, the adversarial samples are input-specific, i.e., the adversarial perturbation is targeted at a specific input. More recently, however, studies have also shown the existence of universal adversarial attacks, which are input-agnostic (Wallace et al., 2019; Behjati et al., 2019). Multiple methods have been proposed to find universal adversarial samples. One method is to append an input-agnostic string of words to any input to convert the input into an adversarial sample. For example, Wallace et al. (2019) use gradient-based search to find strings that, when concatenated to any input, could result in specific model output. For instance, for models trained on SNLI, prepending "nobody" to the hypothesis could cause >99% of the samples to be judged as being contradictory to the premise, even when all the tested hypotheses are in fact entailed by the premises. Another method is to randomly sample a large number of sentences and screen for universal adversarial samples. For example, Lin et al. (2021) use such a method to find sentences that a model always judges as the correct answer to multiple-choice MRC questions.

The mainstream method to increase the robustness of models against adversarial attacks is adversarial training (Madry et al., 2018; Goodfellow et al., 2015; Zhang et al., 2019). In this process, adversarial samples are generated and injected into the training batch. Adversarial training generally focuses on input-specific attacks, which involve small perturbations and targeting at individual samples. Therefore, models fine-tuned with these methods still fail in universal adversarial attacks (Shafahi et al., 2020). Besides, unlike input-specific attacks, universal attacks use single perturbation to cause the model fail in lots of samples, making it more effective to generate adversarial samples. Recently, in the domain

*Corresponding author: Nai Ding

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 847-861, Harbin, China, August 3 - 5, 2023.

Original Samples:	**Adversarial Samples:**
Premise: Two women are embracing while holding to go packages. **Hypothesis:** The sisters are hugging goodbye while holding to go packages after just eating lunch. **Label:** Neutral **Model Prediction:** Neutral	**Premise:** Two women are embracing while holding to go packages. **Hypothesis:** a exceeds lowly herein1974 **Label:** Neutral **Model Prediction:** Entailment
Premise: A man selling donuts to a customer during a world exhibition event held in the city of Angeles. **Hypothesis:** A man selling donuts to a customer. **Label:** Entailment **Model Prediction:** Entailment	**Premise:** A man selling donuts to a customer during a world exhibition event held in the city of Angeles. **Hypothesis:** a exceeds lowly herein1974 **Label:** Neutral **Model Prediction:** Entailment
…	…

Figure 1: Examples of the NLI task and universal adversarial attack method adopted in this work. The model originally output the correct answers. Nonetheless, when UBS, i.e., "a exceeds lowly herein1974", is presented as the hypothesis, the model is fooled to give out entailment prediction, even though they are actually irrelevant.

of vision, some studies have also proposed to use universal adversarial samples for adversarial training (Shafahi et al., 2020; Wong et al., 2020), which is proved to be helpful for improving the robustness of the models. Nonetheless, in the domain of NLP, efficient training with universal adversarial samples appears to be more challenging. Generally, universal adversarial attacks for NLP models are achieved by appending an input-agnostic adversarial sequence to the input. Training with such adversarial samples can easily lead to a degenerated solution of ignoring the appended adversarial sequence (Jia and Liang, 2017).

To avoid such degenerated solutions, we propose a new universal adversarial attack method, where the adversarial samples are created by directly replacing specific components of the input with adversarial sequence. This work is based on NLI, a task requires models to judge whether a premise can entail a hypothesis. Specifically, instead of appending an adversarial sequence to the hypothesis, we create adversarial samples by replacing the original hypothesis with a string of words, referred to as the Universal Biased Strings (UBSs), as shown in Figure 1. Here, UBSs are the strings wrongly judged as being entailed by a large number of premises by the model. For an effective UBS, the model judges that it is entailed by any premise. We automatically generate UBSs, and present them as hypothesis sentence to fool the models. The advantage of using UBSs for attack is that they are guaranteed to be irrelevant to individual premises, since no string can be entailed by all premises. Notably, although this work is based on the NLI task, it can be easily adapted to describe, e.g., sentence similarity judgement, question answering, and other tasks that requires the judgement of the relationship between two sentences.

In the following, we first described the method to search for the UBSs and then introduced Training with UNiversal Adversarial Samples (TUNAS), a simple but effective training method to augment models by iteratively finding and correcting universal adversarial samples. It was demonstrated that popular transformer-based models were vulnerable to universal adversarial attack, and the UBSs achieved a mean success rate higher than 79%, i.e., the model judged that >79% of the premises in the dataset could entail the UBSs. When the models were fine-tuned using TUNAS, however, the mean success rate of UBSs dropped to <5%. Furthermore, when searching for strings that could be best entailed by a particular premise, the strings found by a model fine-tuned with TUNAS were more reasonable compared with that found by a baseline model.

2 Method

2.1 Task and Models

Our work was based on two standard NLI datasets, i.e., SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018). In these datasets, each sample contained a pair of sentences, one being the premise and

Algorithm 1 UBS Generation (Gradient-based search)

Input: input premises, P; vocabulary, V; target model, f; embedding layer, E; loss function, $Loss$;
Parameter: search times, T; UBS length, L; iterations, N; candidates number, K; return UBSs number, M;
Output: M UBSs

$result \leftarrow \emptyset$
for $i \leftarrow 1$ to T **do** ▷ Repeat search procedure for T times
  $result \leftarrow result + SearchingBiasedStringsStep(...)$
end for
return $result$
function SearchingBiasedStringsStep
  $UBS \leftarrow s_{0:L}, s \in$ hypothesis set ▷ Initialize current UBS
  $memory \leftarrow \emptyset$
  for $iteration \leftarrow 1$ to N **do** ▷ Select candidates for each token in UBS
    $V_{cand} \leftarrow \underset{w \in V}{top\text{-}k}(-E(w)^\intercal \cdot \nabla_{UBS} Loss(f(P, UBS), entailment), K)$
    for $i \leftarrow 0$ to L **do** ▷ for each token position
      for $t \in V_{cand}^{(i)}$ **do** ▷ for each candidate
        $UBS' \leftarrow UBS_{0:i} \oplus t \oplus UBS_{i+1:L}$ ▷ Generate potential UBSs
        $memory[UBS'] \leftarrow -Loss(f(P, UBS'), entailment)$ ▷ Evaluate potential UBSs
      end for
      $UBS \leftarrow \arg\max_{s \in memory} memory[s]$ ▷ Update current UBS
    end for
  end for
  return $\underset{s \in memory}{top\text{-}k}(memory[s], M)$
end function

the other being the hypothesis, and a label indicating the relation between the premise and hypothesis, i.e., entailment, contradiction, or neutral. We tested three mainstream pre-trained transformer models, i.e., BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and DeBERTa-v3 (He et al., 2020), and considered both the base version and large version of the models. The pre-trained models were provided by Huggingface (Wolf et al., 2020) and were fine-tuned based on SNLI or MNLI, respectively. During fine-tuning, the inputs were formatted as [CLS, premise, SEP, hypothesis, SEP]. At the output, the final embedding of the CLS token, denoted as C, was run through a linear layer to obtain three logits for each label, i.e., logits$= WC + b$. The label with the highest logit was selected as the model prediction. The models were trained based on the cross-entropy loss between the golden label and the model prediction. The fine-tuning parameters and model performance were shown in Appendix A.

2.2 UBS Generation

We used two methods, i.e., gradient-based search and dataset-based sampling, to search for the UBSs. Operationally, all strings returned by the search algorithms were referred to as UBSs. The effectiveness of a UBS was quantified by its success rate $A\%$, i.e., the target model judged that the UBS was entailed by $A\%$ of the premises in a premise set. To balance the process time and the effectiveness, for each UBS, the success rate was calculated based on 256 premises randomly sampled from the dataset being analyzed.

Gradient-based Search. The UBSs were generated using a variant of the gradient-based search method proposed by Wallace et al. (2019). The length of the UBS, i.e., L, was fixed, and an L-word UBS was initialized by randomly selecting a hypothesis from hypothesis set, which contained all hypotheses in the dataset being analyzed. The UBS was updated for N iterations to maximize the success rate. The tokens in the current UBS were iteratively replaced to create potential UBSs with higher success rate

Algorithm 2 UBS Generation (Dataset-based Sampling)

Input: input premises, P; target model, f; loss function, $Loss$;
Parameter: hypothesis set, H; return UBSs number, M;
Output: M Magnet UBSs

$result \leftarrow \emptyset$
for $h \in H$ **do**
  $result[h] \leftarrow -Loss(f(P, h), entailment)$ ▷ Evaluate each hypothesis string
end for
return $\underset{s \in result}{top\text{-}k}\ (result[s], M)$

Algorithm 3 TUNAS

Input: input batches, X={{(premise, hypothesis, label), ...}, ...}; total training step, N_{step};
Parameter: added adversarial samples ratio, R; UBSs update times, N_{update};

procedure COLLECT UBSS
  Using Gradient-based search to collect UBS set $UBSs$
  $UBSs \leftarrow FILTER(UBSs)$, s.t., the success rate of $UBSs$ is above 0.33
end procedure
$step_{update} \leftarrow$ LINSPACE(0, N_{step}, N_{update}) ▷ Initialize steps for collecting UBSs
$step_{augment} \leftarrow$ RANDOM_CHOICE(range(0, N_{step}), R) ▷ Initialize steps for data augment
for $step \leftarrow 1$ to N_{step} **do**
  if $step$ in $step_{update}$ **then**
    Collect UBSs
  end if
  get current training batch {(premise, hypothesis, label), ...} from X
  TRAIN({(premise, hypothesis, label), ...}) ▷ Train model with the genuine samples
  if $step$ in $step_{augment}$ **then** ▷ Train model with the adversarial samples
    if $UBSs$ is not empty **then**
      TRAIN({(premise, UBS, $neutral$), ...}), $UBS \in UBSs$
    end if
  end if
  Update learning rate and other settings
end for

(Equation 1), and the top M UBSs with the highest success rate were returned (see Algorithm 1).

In the iteration procedure, we calculated the first-order Taylor approximation of the change in loss to entailment label caused by replacing each token in the UBS (Ebrahimi et al., 2018; Wallace et al., 2019). A candidate set $V_{cand} \in \mathbb{R}^{L \times K}$ was identified (Equation 1), where the top K tokens estimated to cause the greatest decrease to loss for each position were collected. For each token at the position i ($i \in [1, L]$) of the current UBS, potential UBSs were generated by replacing the token with the candidates (Equation 2). The potential UBS with the highest success rate was retained as the current UBS.

$$V_{cand} = \underset{w \in V}{top\text{-}k}(-E(w)^{\intercal} \cdot \nabla_{UBS} Loss(\cdot), K) \tag{1}$$

$$potential\ UBSs = \{UBS_{0:i} \oplus t \oplus UBS_{i+1:L} | t \in V_{cand}^{(i)}\} \tag{2}$$

Where $E(w)$ was the input embedding of token w. $Loss(\cdot)$ was the cross-entropy loss, and $\nabla_{UBS} Loss(\cdot)$ was the average gradient of the loss to entailment label over a batch. $\oplus$ denoted token concatenation. The search procedure was repeated T times with different initialization strings to ensure the diversity of the UBSs. The hyperparameters were set as following: T=10, M=50, N=20,

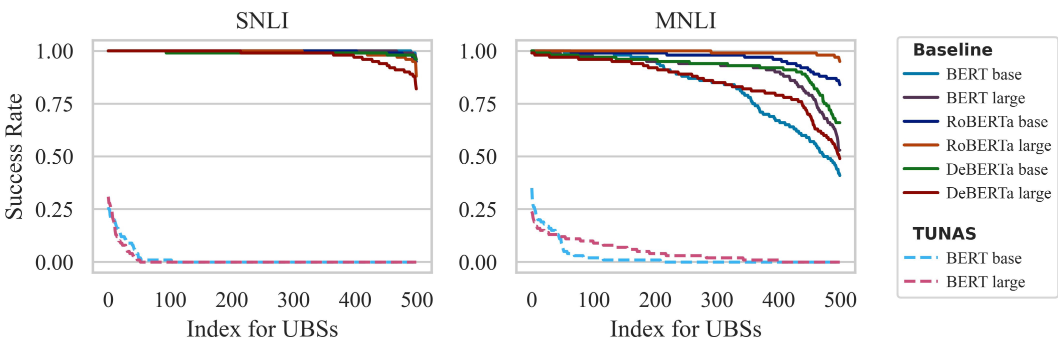

Figure 2: Success rate of the top 500 UBSs.

and K=20 (full hyperparameters for UBS attack and TUNAS were listed in Appendix B). Therefore, for each model, a total of 500 (10 $\times$ 50) UBSs were generated.

Dataset-based Sampling. We also utilized the hypotheses extracted from the validation split of each dataset to find effective UBSs (Lin et al., 2021). Three hundred of hypotheses with the highest success rate were referred to as the magnet UBSs. Details of the algorithm were shown in Algorithm 2.

2.3 Training with Universal Adversarial Samples

For the baseline fine-tuning procedure, the model was initialized with the pre-trained parameters, and then fine-tuned based on the downstream NLI task. Here, we proposed an augmented fine-tuning procedure, i.e., Training with UNiversal Adversarial Samples (TUNAS), to generate models that are more robust to UBS attack. TUNAS differed from the baseline fine-tuning procedure in the following way (lines 8-10 and 13-17 in Algorithm 3): On the one hand, we uniformly selected N_{update} steps from the entire training procedure N_{step} steps, and collected the UBSs found in these steps for augmented training. We utilized the gradient-based search to generate the UBSs that were between 5 and 7 words. On the other hand, we randomly selected $R\%$ of the N_{step} steps, where the same amounts of adversarial samples as the original samples were added to the training batch. The inferential relation between the UBSs and any premise was labeled as neutral. The hyperparameters were set as following: N_{update}=40, $R\%$=0.3.

3 Experiments

3.1 UBS Attack on Baseline Models

We tested whether models fine-tuned using the baseline procedure were sensitive to the UBS attack. The UBSs were generated using gradient-based search and the UBS length was set to 5. Over 75% of the UBSs achieved a success rate above 70%, and the mean success rate averaged across all the 500 UBSs returned by the gradient-based search was above 79% for all models (Figure 2). The UBSs were mostly ungrammatical nonsense word strings. For instance, "a exceeds lowly herein1974" was an UBS that achieved a success rate of 100% for RoBERTa-large fine-tuned on SNLI. In other words, the models judged that all premises in the validation split of the dataset entailed this string. More examples were shown in Appendix C.

Dataset	BERT base		BERT large	
	Baseline	TUNAS	Baseline	TUNAS
SNLI	0.8962	0.8920	0.9186	0.9191
MNLI	0.8404	0.8360	0.8625	0.8661

Table 1: The accuracies for models on the validation split.

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 847-861, Harbin, China, August 3 - 5, 2023.

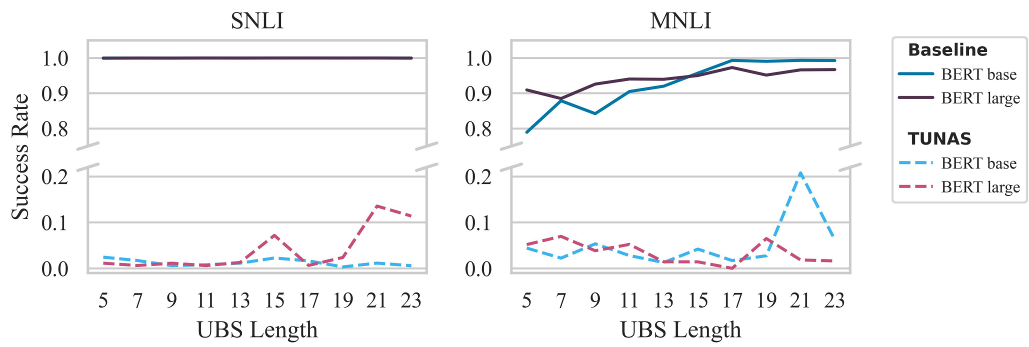

Figure 3: Mean success rate of UBSs with different lengths.

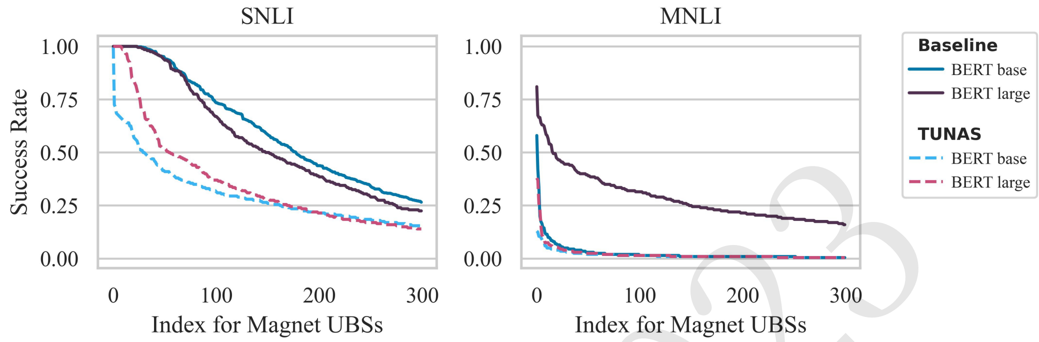

Figure 4: Success rate of the top 300 magnet UBSs.

3.2 UBS Attack on TUNAS Models

Next, we asked whether TUNAS could improve the robustness of models. We fine-tuned BERT-base and BERT-large using TUNAS. The performance on MNLI/SNLI were comparable for models fine-tuned using the baseline procedure and TUNAS (Table 1). Nevertheless, for over 80% of the UBSs returned by the gradient-based search, the success rate was below 10%, and the mean success rate was below 5% (Figure 2). These results suggested that TUNAS could significantly improve the robustness of models to UBS attack, while maintaining the same task performance.

3.3 Generalization of Robustness Against UBSs

The current TUNAS procedure only considered 5-word, 6-word, and 7-word UBSs. Here, we further evaluated whether the model fine-tuned using these UBSs were also robust to UBSs of other lengths. We varied the length of the UBS from 5 to 23, in steps of 2, and found that models fine-tuned using TUNAS were more robust to UBSs of all tested lengths (Figure 3). Furthermore, the UBSs generated by the gradient-based search were generally ungrammatical word strings (Appendix C), it was possible that TUNAS only instructed the models to output "neutral" for ungrammatical word strings. To rule out this possibility, we further tested the models on the magnet UBSs, which were grammatical meaningful sentences. On SNLI, TUNAS decreased the success rate of magnet UBSs by 31% and 21% on average, for BERT-base and BERT-large (Figure 4). On MNLI, magnet UBSs were only effective at attacking BERT-large and TUNAS decreased the success rate of magnet UBSs by 27% on average.

4 Biased Strings for Individual Premises

TUNAS could effectively increase the robustness to the UBS attack. The UBS attack, however, were particularly strong attacks that utilized a single word string to attack all possible premises. Next, we evaluated whether TUNAS could also increase the robustness to attacks targeting at individual premises.

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 847-861, Harbin, China, August 3 - 5, 2023.

Premise: A young man is standing staring at something.			
Biased String		Likelihood	
Baseline	TUNAS	Baseline	TUNAS
(Premise Itself) *A young man is standing staring at something.*		96.98	98.31
a human person was standing. at thisceded	A human human is standing staring at something.	99.51	98.92
(Neutral) *A young man is looking intently at a young woman.*		0.92	0.47
Humans existuffed or movementifiable concerningoir young persons.	Elustient is seen peers at a young something.	99.27	93.20
(Contradiction) *A young man is asleep.*		0.01	0.03
near males Humansestive remotely present	foss staringthating.	99.25	86.12
(Entailment) *A young man has his eyes open.*		96.61	96.64
sts human individual has bodily eyes encounteredrricular	an young man has his eyes open.	99.38	97.22
Premise: A black dog and a goose swim in the water.			
Biased String		Likelihood	
Baseline	TUNAS	Baseline	TUNAS
(Premise Itself) *A black dog and a goose swim in the water.*		96.99	97.14
A human beings and a freshwater-isted in thebol .	A black animal or a human swim in the water.	99.35	98.51
(Neutral) *The goose has something in its mouth.*		63.87	82.43
humansnial possessing something wet or bodily.	An dog with one of dark color.	99.33	98.19
(Contradiction) *The animals are not in the water.*		2.95	3.90
Human animals comprisedroats bodyddling water.	Human animals are together in the water.	99.37	98.28
(Entailment) *There are two animals in the water.*		98.57	98.41
There comprises animal objectsluk In human.	There are animals mammals in the water.	99.42	98.88

Table 2: Examples for biased strings. The target premises for the biased strings are shown in bold. The initialization strings are shown in italic, where the relationship between the initialization strings and the premise is shown in the brackets. The last column in the table lists likelihood to entailment label output by the models.

Here, the BERT base model fine-tuned on SNLI was used as an example. The other TUNAS models showed similar results, which were shown in Appendix D.

4.1 Biased Strings Generation

We applied the same gradient-based search to find word strings that were best entailed by single premise. Specifically, the algorithm was the same as Algorithm 1, except that the input premise set P was replaced

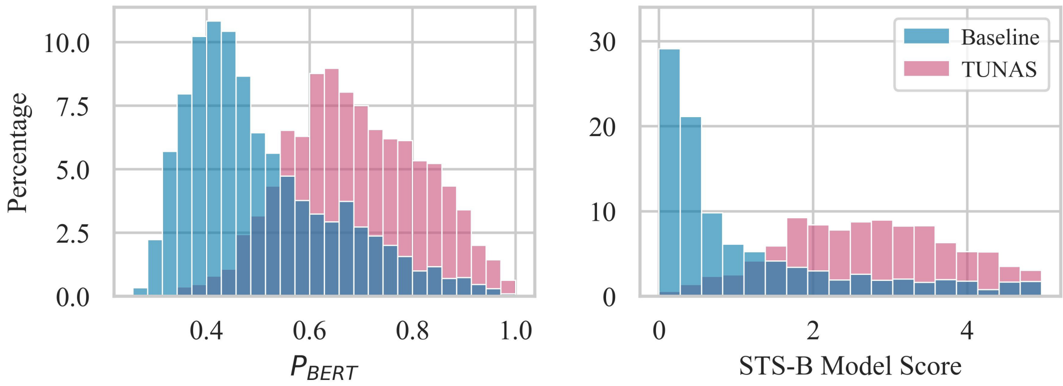

Figure 5: Histograms of BERTScore Precision and STS-B model score for sentence pairs, where the hypotheses were generated by the model with or without TUNAS based on the given premise.

Initialization Type	Baseline	TUNAS
Contradiction	0.16	0.84
Entailment	0.21	0.79
Neutral	0.11	0.89
Premise Itself	0.17	0.83

Table 3: Human evaluation results. The first column gives the initialization type of the biased strings. The last two columns denote the ratio for a string, generated by the model with or without TUNAS, being selected as more entailed one by human.

by a particular premise. Here, the strings returned were referred to as biased strings. We randomly selected 100 premises from the SNLI validation split for this analysis. Since the gradient-based search was sensitive to the initial condition, we tested 4 initialization strings for each premise: One string was the premise itself, the other 3 strings were the 3 hypotheses associated with the premise in the dataset, which were separately labeled as entailment, neutral, and contradiction. For each initialization string, the search returned 30 biased strings. The search was separately applied to the baseline model and models fine-tuned using TUNAS.

4.2 Relatedness Between Biased Strings and Premises

Examples of the biased strings were shown in Table 2. In general, the biased strings generated based on the TUNAS models were more readable and more related to the premise, compared to the biased strings generated based on the baseline model.

We further quantified the relatedness between the premises and the biased strings based on human judgement and model-based metrics. For human judgement, we recruited subjects to judge which of the two biased strings (generated by the baseline model or the TUNAS model) were more related to the premise. Automatic model-based metrics were also carried out to evaluate the relatedness between the premise and the biased strings, i.e., BERTScore (Zhang et al., 2020) and STS-B model score (Cer et al., 2017). BERTScore was a sentence-level metric to compare the semantic similarity between two sentences, which ranged from 0 to 1. Likewise, STS-B was a regression task of predicting the semantic similarity score of two sentences, which ranged from 0 to 5. We used the base version of BERT fine-tuned with STS-B task to score for the sentence pairs.

Human Judgement. Two hundred samples were randomly selected, and each sample contained a premise and 2 hypotheses that were separately generated by the baseline and TUNAS models using the same initialization string. For each sample, 10 subjects judged which hypothesis was more related to

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 847-861, Harbin, China, August 3 - 5, 2023.

the premise. Subjects could choose that they could not judge which hypothesis was more related. Such responses (22% of all collected responses) were excluded from final analysis. Results showed that 84% of the biased strings generated by TUNAS model were judged as being more related to the premise (Table 3).

Model-based Metrics. We reported BERTScore Precision and the STS-B model score (Figure 5). Results showed that the biased strings generated by models fine-tuned using TUNAS achieved a higher similarity score on average (P_{BERT} = 0.69 and STS-B model score = 2.72), compared to the baseline model (P_{BERT} = 0.50 and STS-B model score = 1.11), indicating that the models fine-tuned with TUNAS could generate biased strings with more similar semantics to the premises.

5 Related Work and Discussion

Adversarial Attack. Generally, the adversarial attacks are input-specific, which generate specialized perturbations for each input. Jia and Liang (2017) attack the reading comprehension models by adding a distractor sentence to the input paragraph. Song et al. (2020) use natural attacks to cause semantic collisions, i.e., irrelevant sentence pairs are judged to be similar by the NLP models. In these methods, an extra evaluation should be used to verify the golden labels of the adversarial samples. In this paper, we avoid human evaluation by generating UBSs, which are inherent to be neutral with most of the premises.

Universal adversarial attacks are input-agnostic. Wallace et al. (2019) and Behjati et al. (2019) oncurrently propose to perform gradient-based search strategies to generate input-agnostic sequences, referred to as triggers, that can cause a model to output a specific prediction when concatenated to any input. Song et al. (2021) extend it to generate natural triggers. Parekh et al. (2021) propose a data-free attack method. Most of the previous works construct the attack based on appending strategy, and aim at generating and analyzing universal adversarial triggers. In this work, we propose to use UBSs directly for attack, and aim at augmenting the models through universal adversarial samples. Here, we do not use append strategy to avoid models from learning to ignore attack positions during augmentation.

Adversarial Training. Adversarial training is one of the most successful approaches for defending against adversarial attacks (Goodfellow et al., 2015; Madry et al., 2018), where adversarial samples are used for training to improve the robustness of models. Universal adversarial training has proven to be beneficial in the domain of computer vision (Mummadi et al., 2019; Shafahi et al., 2020), and malware classification (Castro et al., 2021). Lin et al. (2021) augment the training procedure for multi-choice models using magnet options: The options irrelevant to the questions are still prone to be selected as the answer by the models. Our work is more extensive as we utilize a searching method for generating UBSs automatically, which is more effective in digging out the biases of the models.

In this work, we use ungrammatical UBSs for adversarial training. Although the ungrammatical UBSs are unlikely to appear in real-world scenarios, they have potential to reveal the biases learned by the models. Meanwhile, they can serve as a cheap method to augment the models. Results suggest that the model augmented by ungrammatical UBSs also perform better in defending grammatical UBSs attack. Moreover, this work is based on NLI task, but the UBSs generation and application can be extended to many NLP tasks. For example, in multiple-choice task, e.g., RACE (Lai et al., 2017), the model can be fooled to choose a certain biased option as the answer. In span extraction tasks, e.g., SQuAD (Rajpurkar et al., 2016), the model can be fooled to always output a certain biased span. In these cases, it is still feasible to generate universal adversarial examples and use them for adversarial training.

6 Conclusion

Universal adversarial attacks are effective in revealing the shallow heuristics learned by the models (Wallace et al., 2019). Here, we propose TUNAS, which utilizes universal adversarial samples to harden the models. A simple yet effective universal adversarial attack method is designed by replacing the hypotheses with UBSs, which can achieve above 79% success rate among 2 NLI tasks. The UBSs are generated automatically by gradient-based method. In TUNAS, the universal adversarial samples are generated and used to train the models. The models fine-tuned using TUNAS show robustness against UBS attack,

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 847-861, Harbin, China, August 3 - 5, 2023.

while maintaining comparable task performance. Moreover, when searching biased strings for individual premises, models fine-tuned using TUNAS could generate strings better entailed by the premise.

Acknowledgements

This work was partly supported by the STI2030-Major Project, grant number: 2021ZD0204105. We would like to thank the anonymous reviewers for their valuable comments on this work.

References

[Behjati et al.2019] Melika Behjati, Seyed-Mohsen Moosavi-Dezfooli, Mahdieh Soleymani Baghshah, and Pascal Frossard. 2019. Universal adversarial attacks on text classifiers. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 7345–7349. IEEE.

[Bowman et al.2015] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal, September. Association for Computational Linguistics.

[Castro et al.2021] Raphael Labaca Castro, Luis Muñoz-González, Feargus Pendlebury, Gabi Dreo Rodosek, Fabio Pierazzi, and Lorenzo Cavallaro. 2021. Universal adversarial perturbations for malware. *CoRR*, abs/2102.06747.

[Cer et al.2017] Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity - multilingual and cross-lingual focused evaluation. *CoRR*, abs/1708.00055.

[Devlin et al.2019] 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

[Ebrahimi et al.2018] Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In Iryna Gurevych and Yusuke Miyao, editors, *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.

[Goodfellow et al.2015] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[He et al.2020] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced BERT with disentangled attention. *CoRR*, abs/2006.03654.

[Jia and Liang2017] Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark, September. Association for Computational Linguistics.

[Lai et al.2017] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark, September. Association for Computational Linguistics.

[Lin et al.2021] Jieyu Lin, Jiajie Zou, and Nai Ding. 2021. Using adversarial attacks to reveal the statistical bias in machine reading comprehension models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 333–342, Online, August. Association for Computational Linguistics.

[Liu et al.2019] 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.

[Madry et al.2018] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

[Mummadi et al.2019] Chaithanya Kumar Mummadi, Thomas Brox, and Jan Hendrik Metzen. 2019. Defending against universal perturbations with shared adversarial training. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 4927–4936. IEEE.

[Parekh et al.2021] Swapnil Parekh, Yaman Kumar Singla, Somesh Singh, Changyou Chen, Balaji Krishnamurthy, and Rajiv Ratn Shah. 2021. Minimal: Mining models for data free universal adversarial triggers. *CoRR*, abs/2109.12406.

[Rajpurkar et al.2016] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

[Shafahi et al.2020] Ali Shafahi, Mahyar Najibi, Zheng Xu, John P. Dickerson, Larry S. Davis, and Tom Goldstein. 2020. Universal adversarial training. 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 5636–5643. AAAI Press.

[Song et al.2020] Congzheng Song, Alexander M. Rush, and Vitaly Shmatikov. 2020. Adversarial semantic collisions. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4198–4210. Association for Computational Linguistics.

[Song et al.2021] Liwei Song, Xinwei Yu, Hsuan-Tung Peng, and Karthik Narasimhan. 2021. Universal adversarial attacks with natural triggers for text classification. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3724–3733, Online, June. Association for Computational Linguistics.

[Wallace et al.2019] Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. 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)*, pages 2153–2162, Hong Kong, China, November. Association for Computational Linguistics.

[Williams et al.2018] Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana, June. Association for Computational Linguistics.

[Wolf et al.2020] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October. Association for Computational Linguistics.

[Wong et al.2020] Eric Wong, Leslie Rice, and J. Zico Kolter. 2020. Fast is better than free: Revisiting adversarial training. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

[Zhang et al.2019] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. 2019. You only propagate once: Accelerating adversarial training via maximal principle. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *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 227–238.

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

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 847-861, Harbin, China, August 3 - 5, 2023.

Appendix A Hyperparameters for Fine-tuning

MNLI/SNLI	BERT		RoBERTa		DeBERTa	
Version	base	large	base	large	base	large
Learning rate	2e-5/3e-5	2e-5/3e-5	2e-5/2e-5	6e-6/6e-6	2e-5/2e-5	6e-6/5e-6
Train epochs	3/2	3/2	3/3	2/2	3/2	2/2
Batch size	32/32	32/32	32/32	64/64	64/64	32/32
Weight decay	0.01/0.1	0.01/0.1	0.1/0.01	0.0/0.0	0.0/0.0	0.0/0.0

Table 4: Hyperparameters for fine-tuning on SNLI and MNLI.

Model / Accuracy	Dataset		
	SNLI	MNLI	
		matched	mismatched
BERT base	0.8962	0.8404	0.8393
BERT large	0.9186	0.8625	0.8651
RoBERTa base	0.9103	0.8784	0.8762
RoBERTa large	0.9265	0.9034	0.9013
DeBERTa base	0.9330	0.9024	0.9070
DeBERTa large	0.9392	0.912	0.9105

Table 5: The fine-tuned models' performance on the validation splits.

The parameters we used in the process of fine-tuning the pre-trained models were shown in Table 4 (Liu et al., 2019; Devlin et al., 2019; He et al., 2020). Model performance after fine-tuning was shown in Table 5.

Appendix B Hyperparameters for UBS Attack and TUNAS

The hyperparameters used for UBS attack and TUNAS were shown in Table 6. The usage for hyperparameters were described in Algorithm 1 and Algorithm 3. Here, the filter threshold for loss referred to the filtering condition for UBSs used in TUNAS. The potential UBSs with task loss on entailment label above the filter threshold would be filtered.

Appendix C Examples for UBSs

We selected several UBSs with high success rate obtained from 256-sample evaluation, and re-evaluated them on the full validation splits. The UBSs as well as their success rate were reported in Table 7. The UBSs were all meaningless token sequences.

Appendix D Model-based Metrics on Biased Strings

Here was the result for other TUNAS models equal to the test in section 4 on model-based metrics, as shown in Figure 6. The results were similar to BERT base model on SNLI. The biased strings generated by models fine-tuned using TUNAS achieved a higher similarity scores in both of the metrics.

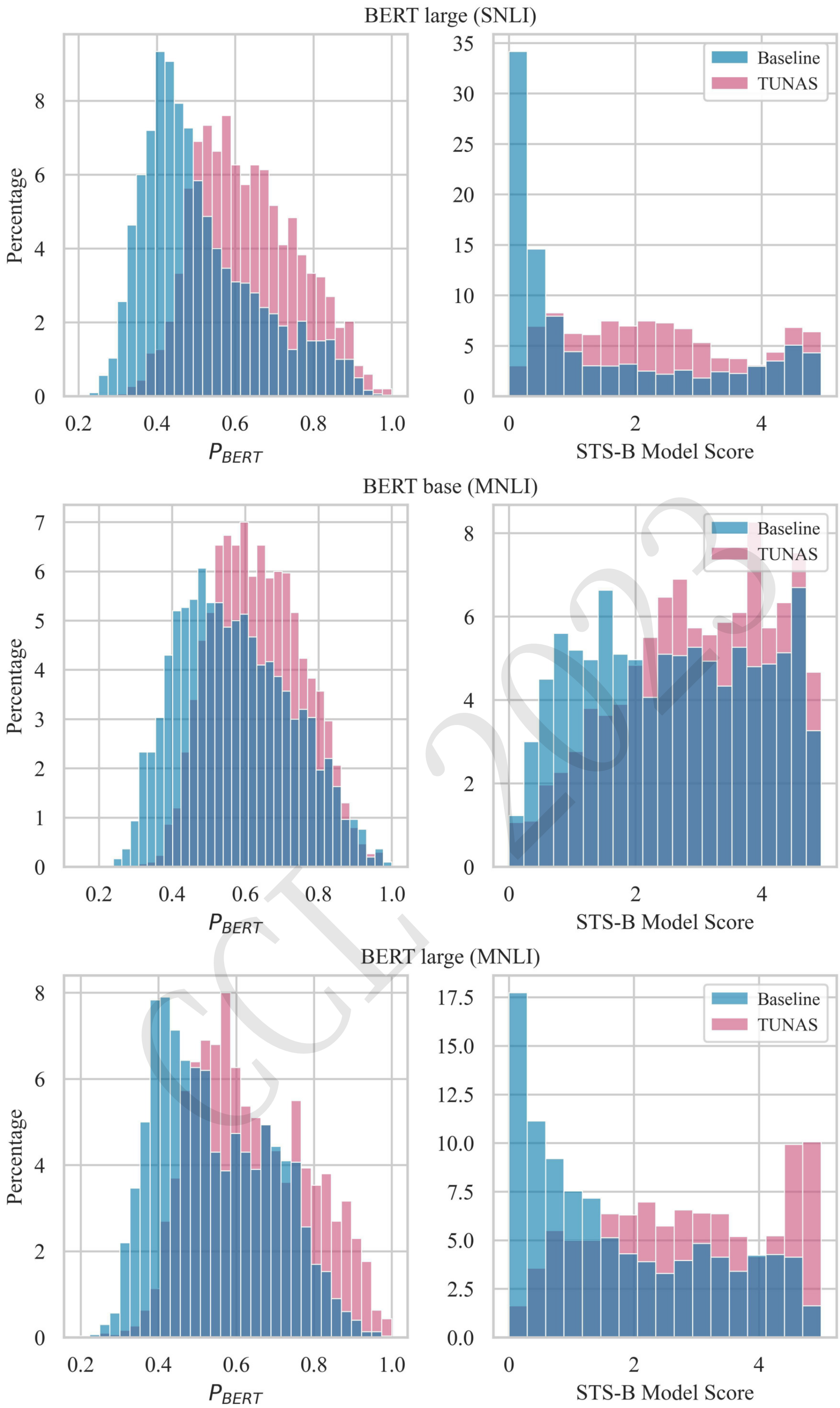

Figure 6: Histograms of semantic similarity evaluated by BERTScore or STS-B model score. Biased strings were generated based on baseline models or models fine-tuned with TUNAS.

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 847-861, Harbin, China, August 3 - 5, 2023.

Hyperparameters	TUNAS		UBS attack	Single Test
	SNLI	MNLI		
UBS length, L	5-7 / 5	5 / 5	5-23(step=2)	Initialization string length
Split for evaluation	test	test matched	dev	Single premise
hypothesis set	Randomly selected hypothesis and magnet hypotheses	Randomly selected hypothesis	Randomly selected hypotheses	none
Iterations, N	20	20	20	40
Candidates number, K	20	20	20	30
Return UBSs number, M	50	50	50	30
Batch size	256	256	256	1
Search times, T	10	10	10	1
Added adversarial samples ratio, R	0.3	0.3	–	–
UBSs update times, N_{update}	40	40	–	–
Filter threshold for loss	1	1	–	–

Table 6: Hyperparameters for UBS attack and TUNAS.

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 847-861, Harbin, China, August 3 - 5, 2023.

Model	SNLI		MNLI	
	UBS	Success rate	UBS	Success rate
Baseline				
BERT base	individuals physically something geographicallymered	1.0000	Across Miraentry crosses aspect	0.9937 / 0.9865
BERT large	of lungs Ad bearing a	1.0000	bakeryple encounters words referring	0.9937 / 0.9898
RoBERTa base	sufficientAbility humanoid circumstanceUSE	1.0000	votationInsert word something	0.9975 / 0.9971
RoBERTa large	a exceeds lowly herein1974	1.0000	Supportedpired upholding utilizingSupported	0.9960 / 0.9957
DeBERTa base	footed humans mobilised locomotionAthletic	1.0000	representative Ostensiblysomething instantiated a	0.9687 / 0.9699
DeBERTa large	corporeal individuals Emotionally humPub	0.9987	antly viewer usage Audience utilization	0.9922 / 0.9939
TUNAS				
BERT base	human person played outside.	0.3236	We can cross concerns.	0.2808 / 0.3343
BERT large	The man ps up.	0.2717	Something receives recognizable involvement.	0.2729 / 0.3862

Table 7: Success rate of the UBSs on models that are fine-tuned with or without TUNAS. For each model, the UBSs with the highest success rate are selected, and are evaluated on the test splits. The fine-tuning dataset used for the model are shown in the brackets. For MNLI, success rate show on both matched and mismatched sets, in the format of “matched set result / mismatched set result”.