FiRo: Finite-context Indexing of Restricted Output Space for NLP Models Facing Noisy Input

Minh Nguyen[1] [*] **Nancy F. Chen**[2,3,4]
[1] Cornell University, USA
[2] Institute for Infocomm Research (I2R), A*STAR, Singapore
[3] CNRS@CREATE, Singapore
[4] Centre for Frontier AI Research (CFAR), A*STAR, Singapore

Abstract

NLP models excel on tasks with clean inputs, but are less accurate with noisy inputs. In particular, character-level noise such as human-written typos and adversarially-engineered realistic-looking misspellings often appears in text and can easily trip up NLP models. Prior solutions to address character-level noise often alter the content of the inputs (low fidelity), thus inadvertently lowering model accuracy on clean inputs. We proposed *FiRo*, an approach to boost NLP model performance on noisy inputs without sacrificing performance on clean inputs. *FiRo* sanitizes the input text while preserving its fidelity by inferring the noise-free form for each token in the input. *FiRo* uses finite-context aggregation to obtain contextual embeddings which is then used to find the noise-free form within a restricted output space. The output space is restricted to a small cluster of probable candidates in order to predict the noise-free tokens more accurately. Although the clusters are small, *FiRo*'s effective vocabulary (union of all clusters) can be scaled up to better preserve the input content. Experimental results show NLP models that use *FiRo* outperforming baselines on six classification tasks and one sequence labeling task at various degrees of noise [1].

1 Introduction

Extensive use of pretrained language models (Radford et al., 2018; Devlin et al., 2019; Liu et al., 2019) has led to impressive performance on clean text. However, these models are not robust to natural noise (e.g. irregular capitalization, misspellings, creative mix of characters and digits) and adversarial noise (Pruthi et al., 2019). Thus, they often underperform when facing noisy inputs (e.g. social media text) during deployment (Rosenthal et al., 2017; Belinkov and Bisk, 2018).

Deployed models which analyze user inputs need to do well on both clean and noisy inputs.

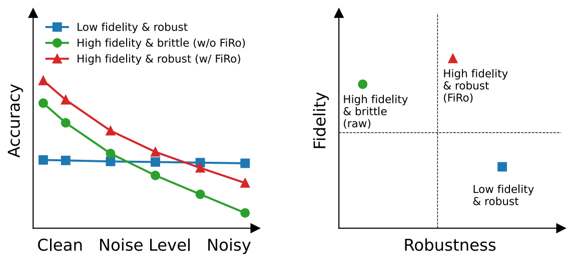

Figure 1: *FiRo* has better better fidelity-robustness trade-off pre-processing text than other approaches do (including not doing pre-processing, i.e. *raw*). Thus, NLP models when using *FiRo* pre-processed text do better on both clean and noisy inputs than when using text pre-processed by other approaches. Text pre-processed by low-fidelity approaches may lead to poor performance on clean inputs while not doing pre-processing may lead to poor performance on noisy inputs. Also see Section 5.2 for actual fidelity-robustness estimation.

Thus, models need to balance between the trade-offs of (1) sensitivity to semantic differences and (2) robustness to noise. A model sensitive to even minor input changes can differentiate semantic changes but is also not very robust. A model that always output the same prediction regardless of the inputs is extremely robust (Jones et al., 2020) yet is of little use because it can only make trivial predictions. Models can be robustified by training with an additional denoising objective (e.g. BART (Lewis et al., 2020)). However, they may need to be retrained to cope with additional types of input noise (e.g. noise faced when adapting to text input in new domains) and retraining could be costly because of the large number of parameters in these models. In contrast, lightweight methods such as Spell correctors (Pruthi et al., 2019) and Robust Encoding (Jones et al., 2020) can be adapted quickly, cheaply, and independently of the NLP models to cope with additional types of input noise.

Spell correctors (Pruthi et al., 2019) and Robust Encoding (Jones et al., 2020) modify the inputs

[*] This work was initiated during internship at A*STAR.
[1] Code is available at `https://github.com/mnhng/FiRo`

Original	Alex Trebek, host of Jeopardy!, is recovering from a minor heart attack in Los Angeles.
RobEn	alex **think**, **heart** of jeopardy!, is recovering from a **mother** heart attack in **less** angeles.
scRNN	**[U] [U]**, host of **[U]**!, is **[U]** from a minor heart attack in a los angeles.
Original	**Tom** testified against John. **Tom** refused to turn on his friend. (**NLI label:** Contradiction)
Perturbed	**Tom** testified against John. **Tim** refused to turn on his friend. (**NLI label:** Neutral?)

Table 1: Impacts of altering input. (Above) Information lost due to robustification. RobEn replaces words with non-synonyms (e.g. host to **heart**). scRNN replaces infrequent words with UNK (**[U]**) tokens. (Below) Label flips due to adversarial noise altering a single word (invalid constant ground-truth label assumption).

to remove variations due to noise (see Table 1). Such modification may reduce semantic fidelity and make NLP models unable to perceive the semantic differences between text inputs. For example, spell correctors often have limited vocabulary so they replace low-frequency and OOV words with UNK (unknown) tokens while Robust Encoding may map non-synonymous words to the same token. This could lower the accuracy of downstream NLP models. Figure 1 illustrates this fidelity-robustness trade-off. Robust approaches (red and blue) lead to better downstream performance on noisy inputs while high-fidelity approaches (red and green) are more suitable than low-fidelity ones for clean inputs. A high-fidelity and robust approach leads to good performance across the noise spectrum and would be ideal for deployment.

We propose *FiRo* (stands for Fidelity-Robustness), a fidelity-preserving neural pre-processor that helps NLP models cope with input character-level noise. Given a noisy sequence of words, *FiRo* predicts the words' identities. *FiRo* can help downstream models achieve high accuracy on both clean and noisy inputs. Instead of using a common softmax covering all vocabularies, *FiRo*'s output space is input-specific and is restricted to only probable vocabularies. The restricted output space makes *FiRo*'s output less susceptible to input noise. Although the vocabulary size at each position is small, the effective model's vocabulary size (union of all softmaxes) can be sufficiently large. Since *FiRo* can scale up the effective vocabulary size with minimal penalty on robust lexical prediction accuracy, *FiRo* covers more low-frequency and OOV words than prior models, thus better maintains input fidelity. *FiRo* indexes into the restricted output spaces using contextual input token embeddings. However, context window is finite instead of spanning the whole sequence so as to localize the effect of input noise. Experiment results show that models that use *FiRo* achieve better results on six classification tasks and one sequence labeling task at various levels of character-level noise.

2 Background

2.1 Realistic Imperceptible Character-Level Noise

Adversarial noise can flip models' prediction while being *imperceptible* to humans (Szegedy et al., 2014). Since the noise is *imperceptible* to humans, humans' prediction is invariant to the existence of the noise. Thus, if models change their prediction as the result of the injected *imperceptible* noise, they are not robust. However, if adversarial noise was *perceptible*, this type of robustness evaluation based on the invariant humans' prediction assumption might be invalid. This is because humans' prediction could have changed as humans perceive the input difference (see Table 1). In NLP, it is non-trivial to design *imperceptible* adversarial noise (Zhao et al., 2018), since sentence-level noise (Jia and Liang, 2017) or word-level noise (Glockner et al., 2018) are perceptible to humans (Alzantot et al., 2018). In contrast, character-level noise (Ebrahimi et al., 2018; Belinkov and Bisk, 2018) could be imperceptible to humans, as psycholinguistic studies demonstrated that humans may not be affected by jumbled internal characters (Rawlinson, 1976; McCusker et al., 1981).

Yet, experts disagree about what level of character-level noise would qualify as *perceptible*. Rawlinson (1976) and Perea and Rosa (2002) suggested that humans are unaffected by character-level noise created by permuting internal characters, altering font size, or mixing cases (capitalization). However, Mayall et al. (1997); Davis (2003), and Rayner et al. (2006) showed that mixing cases and character swaps would be *perceptible* since it sometime causes humans to fail to comprehend the text. When noise that may cause comprehension failure is injected into the text, it is unreasonable to expect the same prediction from both humans and models. Given the lack of consensus, evaluation at one level of noise is inadequate since the chosen level may result in comprehension failure in humans, resulting in misleading conclusions. We evaluated our models using multiple levels of character-level noise to analyze model performance when faced with noisy input.

2.2 Controlled Noise Injection

As it is difficult to gather human-written text at different noise levels, we conducted experiments using data injected with different levels of adversarial noise. Adversarial noise is generated based on human error patterns (see Section 3.4) so it can appear naturalistic. However, results at high noise level must be interpreted with caution as ground-truth labels may flip unknowingly (see Table 1).

Adversarial noise can be created using white-box attacks or black-box attacks. In white-box attacks (Goodfellow et al., 2015), attackers have access to either (a) the attacked model architecture and parameters, or (b) unlimited number of examples labeled by the attacked model. In black-box attacks (Papernot et al., 2017), attackers have no access to the attacked model parameters and only a limited number of examples labeled by the attacked model. Due to this constraint, an auxiliary model is usually needed to craft black-box attacks. In this work, we consider both black-box and white-box attacks since they are complementary. White-box attacks is harder to execute since the attackers must first gain access to the targeted ML model or collect a large labeled training set (Papernot et al., 2017). Black-box attacks are easier to execute but may be less effective. Nevertheless, commercial systems have been attacked successfully using only black-box attacks (Liu et al., 2017).

2.3 Robustify Against Character-Level Noise

Although adversarial training can theoretically robustify models against character-level noise (Liu et al., 2020b; Li et al., 2020; Zhao et al., 2021; Si et al., 2021), in practice its impact can be limited (Pruthi et al., 2019; Jia et al., 2019) as old weaknesses can resurface during training. An alternative is to integrate inductive biases such as character-permutation invariant representation (Belinkov and Bisk, 2018; Wang et al., 2020; Liu et al., 2020a; Sankar et al., 2021) into models. For example, RoVe (Malykh and Lyalin, 2018; Malykh, 2019; Malykh et al., 2023) generates word embeddings that are invariant to character swaps by encoding each word as a bag of characters. Another example is Robust Encoding (Jones et al., 2020), a representation that is invariant to most perturbations within one-character edit distance. Nevertheless, both lines of work require re-training the models (Eshel et al., 2017; Michel and Neubig, 2018; Ribeiro et al., 2018) which may be inconvenient or costly, especially as NLP models grow rapidly in size. General plug-and-play robustification techniques (Contractor et al., 2010) are more appealing since they can be deployed right away, regardless of the tasks or the models. Pruthi et al. (2019) proposed a plug-and-play model to sanitize the input text, obviating the need for re-training downstream models. However, this model struggles with unseen words due to its limited vocabulary. While Pruthi et al. (2019) did propose back-off strategies to handle unseen words, such strategies may work only in specific tasks and may compromise fidelity.

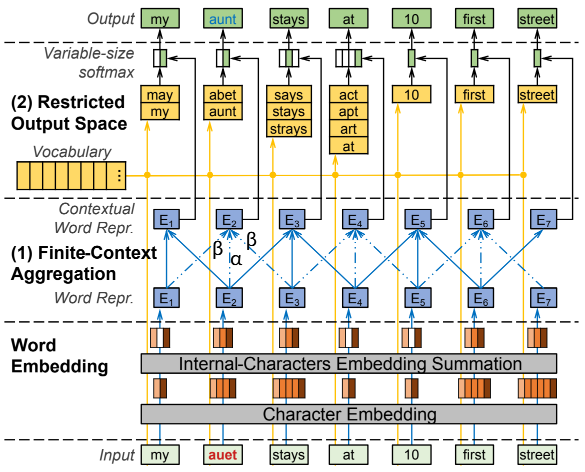

Figure 2: *FiRo*. (1) Construct contextual embedding by finite-context aggregation. (2) Retrieve cluster of words similar to input words (restricted output spaces). Predict output word from cluster using contextual input embedding. The cluster sizes (<100) are much smaller than the vocabulary size (100k).

3 Method

3.1 Proposed Approach

Figure 2 outlines *FiRo* which processes noisy inputs in two steps: (1) Finite-Context Aggregation and (2) Restricted Output Space Indexing. Although *FiRo* operates at the word-level internally, *FiRo* outputs text sequences (by concatenating the output words) that can be analyzed by both word-based or subword-based NLP models.

Finite-Context Aggregation: First, the input string is tokenized into words and the words are turned into embeddings using the same approach used by Sakaguchi et al. (2017). Specifically, all characters in a word are mapped into character embeddings. Subsequently, the first character's embedding, the last character's embedding, and the average of the internal characters' embeddings are concatenated to form the word's embedding. Unlike word embeddings constructed using CNN or RNN, these word embeddings are invariant to noise induced by letter swaps (Sakaguchi et al., 2017; Belinkov and Bisk, 2018). Contextual input embeddings are then weighted averages of adjacent word embeddings (finite-context). Let h_i be the word embedding at position i, the contextual embedding at position i is defined as $\alpha h_i + 0.5(1 - \alpha)(h_{i-1} + h_{i+1})$. The coefficient α is learned during *FiRo* training.

In humans, accurate word recognition also requires identifying constituent characters and surrounding context (Whitney and Grainger, 2004). Higher-order linguistic knowledge and lexical context can refine the representation of individual characters in words and correct for perturbations induced by noise (Heilbron et al., 2020). While global self-attention (Vaswani et al., 2017) is often used for contextual embeddings (a word attending to all other words), it will allow perturbation from a single position to potentially spread to all positions, leading to low robustness. Even though local self-attention (Yang et al., 2018) is more robust than global self-attention as the effect of noise is curtailed to only local words, perturbations may still result in noisy keys that cause self-attention to fail to aggregate information from neighbors. By using local weighted averages, finite-context aggregation localizes the impact of perturbation while ensuring that contextual information from neighboring words is always considered (also see Section 5.1).

Restricted Output Space Indexing: At each position, the output space is restricted to a small variable-size cluster containing words similar to the input word. Like Jones et al. (2020), similarity is defined as one edit distance apart. In particular, for an input word A, all words that are within one edit distance of A are put into the cluster. Indexing into the output space is done by taking the softmax of the dot product of the input word's contextual embedding and the embeddings of words in the cluster. *FiRo* then outputs the word in the cluster with the highest probability. As the cluster sizes vary, so do the softmax sizes. As the clusters are much smaller in size than the full vocabulary (<100 vs 100k), perturbing input words leads to limited and more predictable change to *FiRo*'s output. Despite using small clusters, *FiRo*'s effective vocabulary (union of all clusters) is considerable and can be scaled up (with minimal increase in cluster sizes) to avoid predicting UNK for infrequent and OOV words. Thus, *FiRo* can preserve input fidelity better while being robust to perturbations. In contrast, while Robust Encoding also use clusters to map input words to output words, its mapping ignores context, leading to input fidelity loss.

3.2 Baselines

We compared *FiRo* against 4 baselines: adversarial training, two variants of the *scRNN* spell correctors (Pruthi et al., 2019), and a variant of Robust Encoding called Agglomerative Cluster Encodings (Jones et al., 2020).

- *AdvT*: Adversarial training
- *scRNNu*: *scRNN*, predict UNK for OOV
- *scRNNp*: *scRNN*, let OOV pass through
- *RobEn*: Robust Encoding

Adversarial training is an end-to-end approach that fine-tunes the NLP model to make it more accurate when facing noisy inputs. In contrast, *FiRo* and the spell correctors do not change the NLP model's weights. Between the spell correctors, *scRNNp* is less robust since it lets OOVs pass through unmodified, exposing downstream models to (adversarial) noise. However, *scRNNu* is has lower fidelity since it maps OOVs to UNK. Among the baselines, *RobEn* is the most robust but also has the lowest fidelity (also see Section 5.2). Due to *RobEn*'s low fidelity, a downstream model would not work out-of-the-box on text encoded using *RobEn*. Hence, for *RobEn* specifically, downstream models are fine-tuned using *RobEn* encoded text instead of original text (similar to Jones et al. (2020)). Although RoVe (Malykh et al., 2023) and *FiRo* share some architectural similarities (Sakaguchi et al., 2017), using RoVe to robustify pretrained language models is much harder. This is because RoVe's outputs are sequences of embeddings instead of sequences of words. Thus, combining RoVe with pretrained language models requires replacing the input embeddings of the pretrained language models which may lead to low performances for tasks

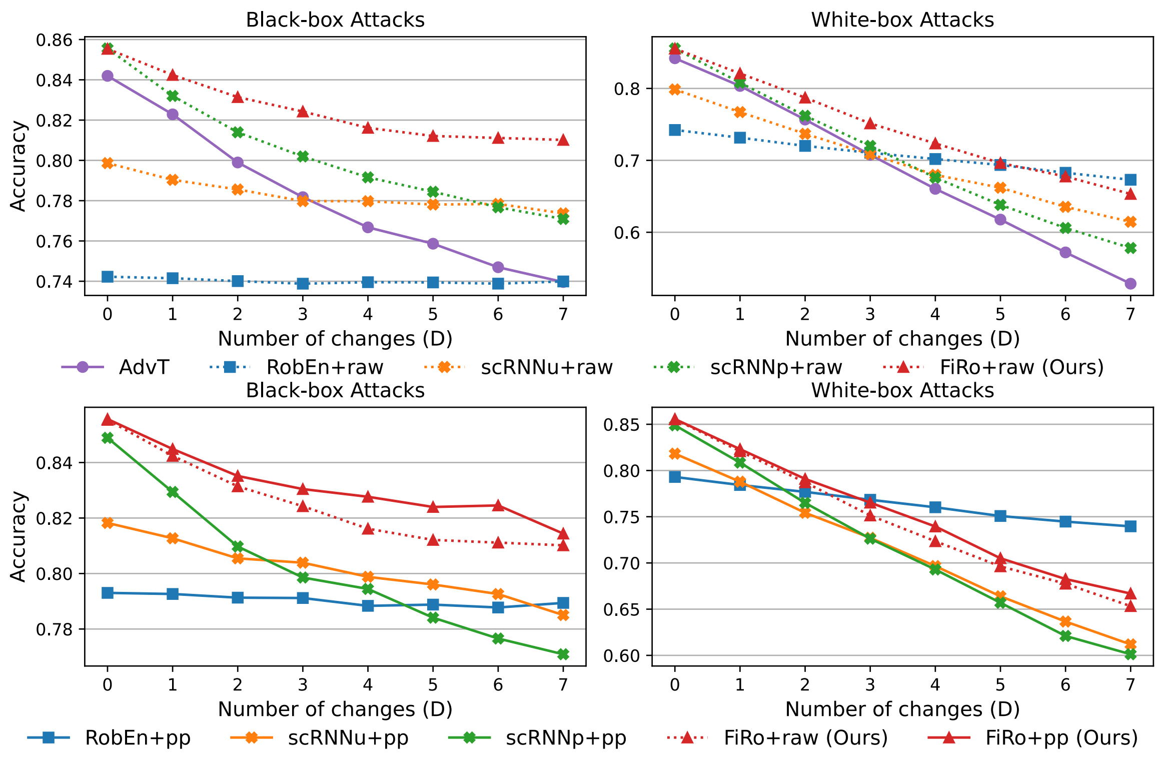

Figure 3: Average GLUE accuracy as the number of adversarial changes introduced into the input text varies. "+raw": BERT fine-tuned using raw text. "+pp": BERT fine-tune using **pre-p**rocessed text. Best approach from top panels are included in bottom panels for comparison. Using *FiRo* leads to higher accuracy than baselines in general.

with limited data for finetuning. Given this limitation, RoVe was not chosen as a baseline.

3.3 Implementation Details

We used PyTorch (Paszke et al., 2019) and *transformers* (Wolf et al., 2020) libraries in this work. BERT is the NLP model and is fine-tuned for 3 epochs using a learning rate of $2e^{-5}$ and a batch size of 8 with AdamW (Loshchilov and Hutter, 2018). *FiRo* and scRNN are trained using Adam (Kingma and Ba, 2014) using a batch size of 50 until convergence (about 10 hours using an NVIDIA TitanXp GPU). We used the GLUE's training sets as data for training *FiRo* and scRNN.

The architecture of scRNN follows that in prior studies (Pruthi et al., 2019; Jones et al., 2020) with the vocabulary size set at 10,000. scRNN is based on a bidirectional LSTM (Graves and Schmidhuber, 2005) with one layer of size 50. Similar to Jones et al. (2020), the 100,000 most frequent words from the COCA corpus (Davies, 2008) are used as RobEn's vocabulary. *FiRo*'s vocabulary is the same as RobEn's. For the adversarial training baseline, the fine-tuned BERT model is further trained using a equal mixed of normal and adversarial examples until there is no further increase in accuracy for a hold-out adversarial set of data.

3.4 Parameters of Character-Level Attacks

Assume that the attacked model is represented by a function f. For an input x with ground truth label y, the model predicts $f(x)$ as the label. An adversarial input x^* is an instance close to x, such that x^* has the same ground truth as x, while $f(x^*) \neq y$ (Szegedy et al. (2014); Liu et al. (2017)). The 'closeness' between x and x^* is denoted as $d(x, x^*)$, the number of words that differ between x and x^* due to some character perturbations. Let A denote the search for an adversarial example, Equation 1 and 2 show the two types of attacks. The auxiliary model used by the black-box attack is denoted f^{aux}. Both attacks need to satisfy the 'closeness' constraint in Equation 3. The level of adversarial noise can be controlled by choosing the choice of character perturbations and the value of D (number of allowable modified words).

$$x^*_{white-box} = A(f, x, y) = arg_{x'} f(x') \neq y \quad (1)$$
$$x^*_{black-box} = A(x, y) = arg_{x'} f^{aux}(x') \neq y \quad (2)$$
$$d(x, x^*) \leq D \quad (3)$$

Adversarial attacks are valid when x and x^* have the same ground-truth label (Szegedy et al., 2014; Liu et al., 2017). For tasks like natural language inference (NLI), even a single letter substitution could violate the label-invariant assumption (see Table 1). Thus, the types of character perturbations and the constant D must be chosen carefully to avoid drawing invalid conclusions.

We followed Gao et al. (2018); Pruthi et al. (2019); Jones et al. (2020) by crafting character-level attacks using four basic operations: (1) character substitution, (2) character deletion, (3) character insertion, and (4) swapping of two adjacent characters. The operations must not cross word-boundary and the characters are picked at random. The assumption that x and x^* have the same ground-truth label is more likely to be violated for larger D, so we evaluated for D in the range from 0 to 7.

3.5 Quantifying Robustness and Fidelity

We can empirically estimate (1) Robustness and (2) Fidelity of *FiRo* by comparing its denoised outputs against the clean input. For each clean input x, a set of perturbed inputs $X^* = \{x^*\}$ is generated. For each noisy input x^* in this set, *FiRo* outputs a denoised version z. This results in a set of denoised outputs $Z = \{z\}$ for each x. The identity (i.e. x) is also included in Z. Robustness quantifies how similar the denoised outputs are (Equation 4), while Fidelity quantifies how closely denoised outputs match the clean input (Equation 5). Let $|Z|$ be the size of Z; L be the length of z; $\mathsf{uniq}(Z)$ be the set of unique elements in Z; $\mathbb{1}$ be the indicator function; and z_i be the ith token in z.

$$\mathsf{Robustness} = \frac{|Z| + 1 - |\mathsf{uniq}(Z)|}{|Z|} \tag{4}$$

$$\mathsf{Fidelity} = \frac{1}{|Z|} \sum_{z \in Z} \frac{1}{L} \sum_{1 \le i \le L} \mathbb{1}_{\{z_i = x_i\}} \tag{5}$$

Robustness and Fidelity range from 0 to 1. Robustness is maximized when all elements in Z are identical. Fidelity is maximized when all elements in Z are the same as x. For multiple x, the Robustness and Fidelity values are averaged. The same estimate can be calculated for the baselines.

Specifically, we estimated the empirical robustness and fidelity using the GLUE data. For each input x, 10 noisy copies x^* are created sequentially by sampling 10 positions in x and inject character-level noise into the tokens at these positions. Thus, the inputs x^* have increasing level of noise.

4 Experiments

4.1 GLUE Experiment Setup

Followed Jones et al. (2020), we experimented on six GLUE (Wang et al., 2019) tasks: MRPC, MNLI, QNLI, QQP, RTE, and SST-2. Approaches are evaluated using average task accuracy. We used the BERT (Devlin et al., 2019) base uncased model as the backbone for all six classification tasks.

For each task, BERT is fine-tuned using the training set and evaluated on the validation set. For evaluation, the input text is first processed by *FiRo*, scRNN, or RobEn before being passed to BERT. There are two ways to fine-tune BERT. The first is to fine-tune BERT using the raw GLUE text. Thus, BERT is oblivious to the text pre-processor (i.e. *FiRo*, scRNN, or RobEn) and fine-tuning does not have to be redone every time the text pre-processor is changed or improved. The second is to fine-tune BERT using the pre-processed text (output of *FiRo*, scRNN, or RobEn). This allows BERT to adapt to the idiosyncrasies of the text pre-processor, resulting in more robust models although at the expense of frequently redoing fine-tuning. We evaluated the approaches using both ways of fine-tuning.

Adversarial examples are found using beam-search with a beam size of 5 similar to Jones et al. (2020). For black-box attacks, beam search uses the backbone BERT model. For white-box attacks, beam search uses the combined model comprising of the backbone and a defender (e.g. *FiRo* or RobEn). The latter are white-box attacks because the combined model is queried without any limit (scaling linearly with the number of test examples).

4.2 Results from GLUE Experiment

Figure 3 shows the average accuracy of 6 GLUE tasks under different adversarial attacks. Left panels show performance under black-box attacks. Right panels show performance under white-box attacks. BERT is fine-tuned using the raw GLUE text in the top panels while it is fine-tuned using the pre-processed text in the bottom panels.

For brevity, in this section, *FiRo* refers to the BERT model using *FiRo* preprocessed text and so on. *FiRo* outperforms scRNNp, scRNNu and adversarial training baselines in all scenarios. *FiRo* is better than RobEn when under black-box attacks. When under white-box attacks, *FiRo* is better when the noise is low ($D \le 5$ top right panel; $D \le 3$ bottom right panel), but RobEn is better when the noise is high. *FiRo* and scRNNp preserve the fi-

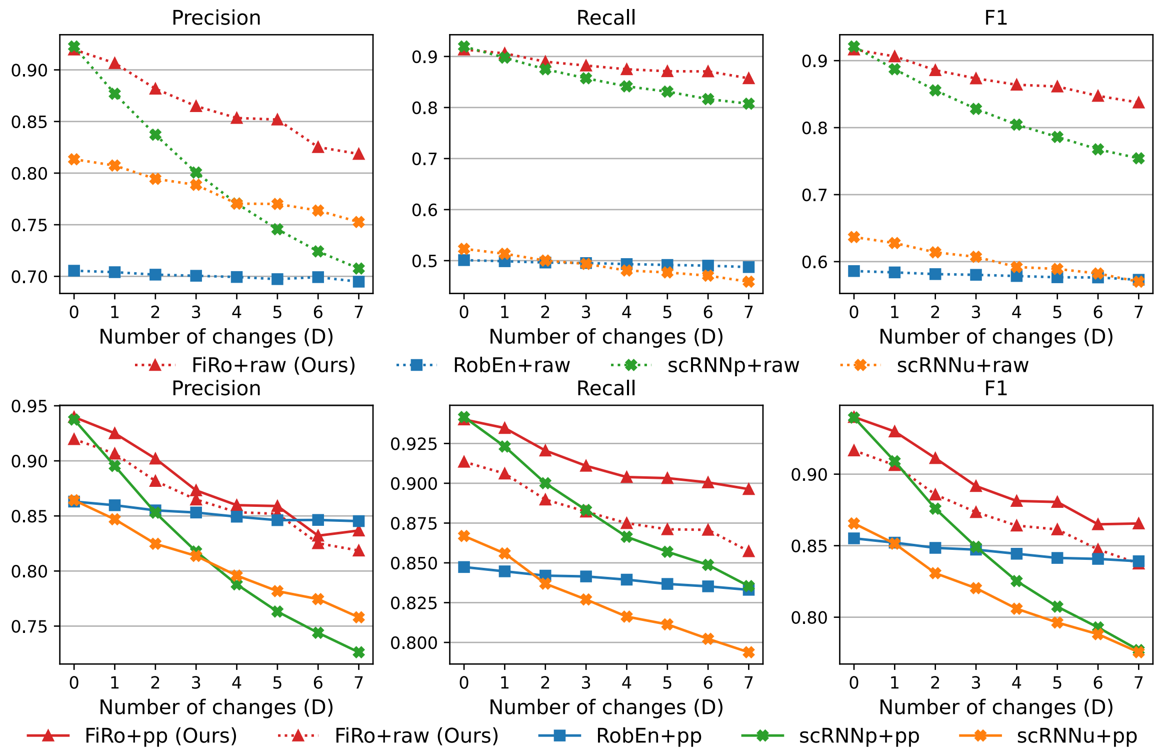

Figure 4: CoNLL-2003 NER performance as a function of D. "+raw": BERT fine-tuned using raw text. "+pp": BERT fine-tune using **p**re-**p**rocessed text. BERT using *FiRo* pre-processed text obtains higher F1 and recall.

delity of the input well since they did well on clean text ($D = 0$). However, as scRNNp lets unrecognized words (e.g. OOVs or words modified by adversarial attacks) pass through unmodified, it is less robust. In contrast, by supporting a large vocabulary using the scope output layer, *FiRo* can cover more infrequent words while also being quite robust. *FiRo* and RobEn are more robust than the other approaches since their accuracy drops less rapidly as the noise level increases.

Another advantage of *FiRo* is that it can be used out-of-the-box. When BERT is not adapted to the pre-processor models (top panels), *FiRo* is generally better than baselines. In addition, the bottom panels show that *FiRo*'s performance when BERT is fine-tuned using raw GLUE text (dotted red line) is quite close to *FiRo*'s performance when BERT is instead of processed GLUE text (solid red line). Being able to use *FiRo* straight away without having to redo fine-tuning the BERT model could lower the cost of NLP model deployment.

4.3 NER Experiment Setup

We use the CoNLL-2003 named entity recognition (NER) dataset (Sang and De Meulder, 2003) for this experiment. We compare *FiRo* against scRNNu, scRNNp, and RobEn. We also use the BERT (Devlin et al., 2019) base uncased model as the backbone and reuse the models (*FiRo*, scRNNu, and scRNNp) trained using the GLUE data. Training of the backbone model is done similar to the procedure in the sequence classification experiment. For sequence tagging, we only explore black-box attacks since, as far as we know, there is no prior work on conducting adversarial attack for tasks that are not classification. Similar to the GLUE experiment, adversarial examples are found using beam-search with a beam size of 5. For sequence tagging, the beam search's objective is maximizing the non-overlapped named entities between the ground truth entity set and the predicted entity set.

4.4 Results from NER Experiment

Figure 4 shows the performance under varying degrees of black-box attack ($D = 0, 1, 2, \ldots 7$). *FiRo* obtains higher F1 score and higher recall than the baselines across all scenarios due to *FiRo*'s high fidelity. In contrast, RobEn and scRNNu underperform in this task because they fail to preserve the fidelity of the inputs. Although scRNNp does well for clean inputs ($D = 0$), its performance degrades quickly as the noise level increases.

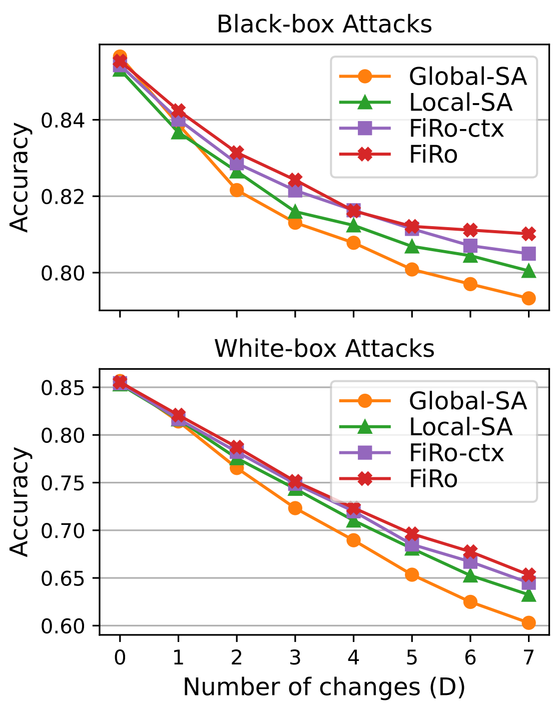

Figure 5: Average GLUE accuracy. Using finite-context aggregation (*FiRo*) results in higher accuracy than using local or global self-attention (SA).

4.5 Spell Correction Experiment

In the previous two experiments, *FiRo* was evaluated on data with synthetic character-level noise. Thus, we additionally evaluate *FiRo*'s performance on a spell correction task with realistic character-level noise. However, it must be emphasized that although *FiRo* functions like a spell corrector with the current cluster design (clusters are defined based on textual edit distance; Section 3.1), *FiRo* is more general than a spell corrector because the cluster design can be adapted for other tasks (e.g. spoken language processing tasks).

For this experiment, we use the GitHub Typo Corpus (Hagiwara and Mita, 2020) which comprises of typos (character-level noise) and grammatical errors collected from public code repositories on GitHub. Since this corpus includes multi-lingual text and diverse types of noise (e.g. character-level, word-level, capitalization noise), we exclude samples that are not typos in English. Specifically, text sequences that are kept must be: (1) written in English, (2) with high probability of being typos, (3) with low perplexity (≤ 5). Consequently, the remaining 53,154 samples are used to test spell correction performance. The models were trained using the GLUE data (see Section 3.3).

Method	Precision	Recall	F1
scRNNu	0.076	0.294	0.120
scRNNp	0.306	0.298	0.302
FiRo	**0.514**	**0.463**	**0.487**

Table 2: Spell correction performance evaluated using word-level metrics (precision, recall, F1)

4.6 Results from Spell Correction Experiment

Table 2 shows the spell correction performance of *FiRo* and scRNN. RobEn is excluded because it is not devised as a spell corrector. Although these models were trained using the GLUE data injected with synthetic character-level noise, they managed to achieve decent spell correction performance on the GitHub corpus which contained typos made by humans. *FiRo* obtains higher precision, recall, and F1 score than scRNN. The performance of *FiRo* could be improved further with better cluster design since the clusters used are based on textual similarity of one edit distance apart but the GitHub corpus definitely contains more diverse typos.

5 Ablation

5.1 Finite-context Aggregation

Figure 5 shows that without context aggregation (FiRo-*ctx*), *FiRo* performs worse. Using global self-attention (*Global-SA*) or local self-attention with the same neighborhood size as the finite-context aggregation (*Local-SA*) results in less robust models. Thus, finite-context aggregation seems to give the best overall performance across noise spectrum.

Method	Fi	Ro	Arith	Geo	Har
RobEn	0.678	0.931	0.794	0.804	0.784
scRNNu	0.863	0.811	0.837	0.837	0.837
scRNNp	**0.938**	0.684	0.801	0.811	0.791
FiRo	0.910	**0.962**	**0.936**	**0.936**	**0.935**

Table 3: Estimates of Robustness (*Ro*) and Fidelity (*Fi*). *Arith*, *Geo*, and *Har* are the arithmetic, geometric, and harmonic means of Robustness and Fidelity.

5.2 Empirical Robustness-Fidelity Estimation

Table 3 reports the empirical Robustness and Fidelity and their arithmetic, geometric, and harmonic mean (also see Section 3.5). *FiRo* has the best Robustness-Fidelity trade-off (highest arithmetic, geometric, and harmonic means). *FiRo*'s Fidelity leads to good performance for clean input ($D = 0$) in Section 4.2 and 4.4. One limitation of the Robustness measure is that it cannot distinguish between "trivially" robust models and those that are genuinely robust. A "trivially" robust but vacuous model can achieve very high Robustness by predicting the same output regardless of inputs. Thus, Robustness should be considered in tandem with Fidelity instead of considered as a standalone metric. For models with similar Fidelity, those with higher Robustness would perform better overall and under noisy condition in particular. For example, *FiRo* and scRNNp have similar Fidelity but since *FiRo* has higher Robustness, *FiRo* beats scRNNp as the noise level increases (see Figure 3).

6 Discussion

In the tasks used to evaluate robustness, we only used synthetic noise. However, the design of the synthetic noise is motivated by observations from naturally generated data, i.e. noisy text written by humans. Besides, while it is true that the noise characteristics do not change between training and test, all methods evaluated exploit this understanding about the noise characteristics in their model architecture/algorithm to enable more robust text processing. Thus, *FiRo* does not benefit from any unfair advantages in this comparison setup. Furthermore, Section 4.6 shows that *FiRo* trained on synthetic noise can generalize to human typos.

Robustification methods should avoid information loss (preserve fidelity) so as to be applicable to many different tasks. Some tasks can be completed using only a few clues from the input, therefore loss of information does not affect task accuracy. However, information loss can greatly affect performance in tasks that require exact phrasing (e.g. summarization, translation). Since *FiRo* can preserve input fidelity better than other approaches, it can be applied to a wider variety of tasks.

FiRo achieved a reasonable level of robustness without sacrificing word recognition performance chiefly due to its restricted output space. This allows *FiRo* to scale up the vocabulary size while still being robust to misspellings. Extending *FiRo* to natural noise would require expanding the output space to cover phenomena such as abbreviations and word play using phonetic spelling (e.g. using 'b4' for 'before', 'gr8' for 'great'). Orthographic and phonological similarity constraints have been explored to improve accuracy of correcting character-level misspellings in Chinese (Nguyen et al., 2021). However, this study is only feasible because of the relatively clear orthographic and phonological relationships between Chinese characters (Nguyen et al., 2018, 2020). Such endeavors for English is beyond the scope of the current study, though there are on-going efforts to address the complex relationship between English spelling, pronunciation, and the presence of different sources of natural noise.

NLP models' lack of robustness to noise limits their usage since user-generated inputs can be noisy. Yet, sacrificing performance on clean inputs to increase robustness is also unacceptable as user-generated inputs can also be clean. We propose a input-sanitizing model named *FiRo* to help deployed NLP models process clean and noisy user-generated text. By combining finite-context aggregation with restricted output space, *FiRo* largely preserves the semantic content of the input while imparting reasonable robustness to NLP models. Thus, *FiRo* can be applied to tasks other than classification, where task-completion requires precise semantic content such as named entity recognition or summarization. Experimental results show *FiRo* outperforming competitive baselines on six classification tasks and one sequence labeling task under various noise conditions. On-going work focus on extending *FiRo* to other noisy inputs such as social media text (Derczynski et al., 2017) or conversation transcripts (Kaplan, 2020; Nguyen and Yu, 2021; Fu et al., 2022).

Limitations

In this work, we focused on improving NLP models resistance to noisy input due to realistic adversarial misspellings. However, natural noise include other types beyond misspellings. For example, natural noise includes the use of emoticons (e.g. <3), abbreviations (e.g. 'lol'), wordplay using phonetic spelling, mixed casing (capitalization) (e.g. 'so COOOOL'), LEET words (Perea et al., 2008) (e.g. 'b4', 'R34D1NG'...). Tackling natural noise would require integrating more explicit visual, phonemic and linguistic knowledge into modeling (Belinkov and Bisk, 2018). Besides, the clusters used by *FiRo* are hand-crafted. Learning the clusters from data may allow models to adapt more quickly to noisy data.

Acknowledgements

We thank Gia H. Ngo and the anonymous reviewers for their helpful and constructive feedback.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of EMNLP*, pages 2890–2896.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *Proceedings of ICLR*.

Danish Contractor, Tanveer A Faruquie, and L Venkata Subramaniam. 2010. Unsupervised cleansing of noisy text. In *Proceedings of COLING*, pages 189–196.

M. Davies. 2008. The corpus of contemporary american english (coca): 560 million words, 1990-present. https://www.english-corpora.org/faq.asp.

M. Davis. 2003. Psycholinguistic evidence on scrambled letters in reading. https://www.mrc-cbu.cam.ac.uk/.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of W-NUT*, pages 140–147.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *Proceedings of ACL*, pages 31–36.

Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuya Yamada, and Omer Levy. 2017. Named entity disambiguation for noisy text. In *Proceedings of CoNLL*, pages 58–68.

Xue-Yong Fu, Cheng Chen, Md Tahmid Rahman Laskar, Shashi Bhushan Tn, and Simon Corston-Oliver. 2022. An effective, performant named entity recognition system for noisy business telephone conversation transcripts. In *Proceedings of W-NUT*, pages 96–100.

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*, pages 50–56. IEEE.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking nli systems with sentences that require simple lexical inferences. In *Proceedings of ACL*, pages 650–655.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *Proceedings of ICLR*.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610.

Masato Hagiwara and Masato Mita. 2020. Github typo corpus: A large-scale multilingual dataset of misspellings and grammatical errors. In *Proceedings of LREC*, pages 6761–6768.

Micha Heilbron, David Richter, Matthias Ekman, Peter Hagoort, and Floris P De Lange. 2020. Word contexts enhance the neural representation of individual letters in early visual cortex. *Nature communications*, 11(1):1–11.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of EMNLP*.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of EMNLP*, pages 4120–4133.

Erik Jones, Robin Jia, Aditi Raghunathan, and Percy Liang. 2020. Robust encodings: A framework for combating adversarial typos. In *Proceedings of ACL*, pages 2752–2765.

Micaela Kaplan. 2020. May i ask who's calling? named entity recognition on call center transcripts for privacy law compliance. In *Proceedings of W-NUT*, pages 1–6.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *Proceedings of ICLR*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of ACL*, pages 7871–7880.

Xiangci Li, Hairong Liu, and Liang Huang. 2020. Context-aware stand-alone neural spelling correction. In *Proceedings of EMNLP: Findings*, pages 407–414.

Hui Liu, Yongzheng Zhang, Yipeng Wang, Zheng Lin, and Yige Chen. 2020a. Joint character-level word embedding and adversarial stability training to defend adversarial text. In *Proceedings of AAAI*, pages 8384–8391.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. 2017. Delving into transferable adversarial examples and black-box attacks. In *Proceedings of ICLR*.

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*, cs.CL/1907.11692.

Zhengxiao Liu, Fali Wang, Zheng Lin, Lei Wang, and Zhiyi Yin. 2020b. De-co: A two-step spelling correction model for combating adversarial typos. In *Proceedings of ISPA/BDCloud/SocialCom/SustainCom*, pages 554–561. IEEE.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *Proceedings of ICLR*.

Valentin Malykh. 2019. Robust to noise models in natural language processing tasks. In *Proceedings of ACL: Student Research Workshop*, pages 10–16.

Valentin Malykh, Taras Khakhulin, and Varvara Logacheva. 2023. Robust word vectors: Context-informed embeddings for noisy texts. *Journal of Mathematical Sciences*, pages 1–14.

Valentin Malykh and Vladislav Lyalin. 2018. Named entity recognition in noisy domains. In *Proceedings of the International Conference on Artificial Intelligence Applications and Innovations*, pages 60–65. IEEE.

Kate Mayall, Glyn W Humphreys, and Andrew Olson. 1997. Disruption to word or letter processing? the origins of case-mixing effects. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 23(5):1275.

Leo X McCusker, Philip B Gough, and Randolph G Bias. 1981. Word recognition inside out and outside in. *Journal of Experimental Psychology: Human Perception and Performance*, 7(3):538.

Paul Michel and Graham Neubig. 2018. Mtnt: A testbed for machine translation of noisy text. In *Proceedings of EMNLP*, pages 543–553.

Minh Nguyen, Gia H Ngo, and Nancy F Chen. 2018. Multimodal neural pronunciation modeling for spoken languages with logographic origin. In *Proceedings of EMNLP*, pages 2916–2922.

Minh Nguyen, Gia H Ngo, and Nancy F Chen. 2020. Hierarchical character embeddings: Learning phonological and semantic representations in languages of logographic origin using recursive neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28(1):461–473.

Minh Nguyen, Gia H Ngo, and Nancy F Chen. 2021. Domain-shift conditioning using adaptable filtering via hierarchical embeddings for robust chinese spell check. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Minh Nguyen and Zhou Yu. 2021. Improving named entity recognition in spoken dialog systems by context and speech pattern modeling. In *Proceedings of SigDial*.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of ACM ASIA CCS '17*, pages 506–519.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of NeurIPS*, pages 8026–8037.

Manuel Perea, Jon Andoni Duñabeitia, and Manuel Carreiras. 2008. R34d1ng w0rd5 w1th numb3r5. *Journal of Experimental Psychology: Human Perception and Performance*, 34(1):237.

Manuel Perea and Eva Rosa. 2002. Does "whole-word shape" play a role in visual word recognition? *Perception & psychophysics*, 64(5):785–794.

Danish Pruthi, Bhuwan Dhingra, and Zachary C Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Proceedings of ACL*, pages 5582–5591.

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

Graham Ernest Rawlinson. 1976. *The significance of letter position in word recognition*. Ph.D. thesis, University of Nottingham.

Keith Rayner, Sarah J White, and SP Liversedge. 2006. Raeding wrods with jubmled lettres: There is a cost.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging nlp models. In *Proceedings of ACL*, pages 856–865.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. Semeval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of SemEval-2017*, pages 502–518.

Keisuke Sakaguchi, Kevin Duh, Matt Post, and Benjamin Van Durme. 2017. Robsut wrod reocginiton via semi-character recurrent neural network. In *Proceedings of AAAI*.

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL*, pages 142–147.

Chinnadhurai Sankar, Sujith Ravi, and Zornitsa Kozareva. 2021. On-device text representations robust to misspellings via projections. In *Proceedings of EACL*, pages 2871–2876.

Chenglei Si, Zhengyan Zhang, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Qun Liu, and Maosong Sun. 2021. Better robustness by more coverage: Adversarial training with mixup augmentation for robust fine-tuning. In *Proceedings of ACL: Findings*.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *Proceedings of ICLR*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NeurIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of ICLR*.

Zhiwei Wang, Hui Liu, Jiliang Tang, Songfan Yang, Gale Yan Huang, and Zitao Liu. 2020. Learning multi-level dependencies for robust word recognition. In *Proceedings of AAAI*, pages 9250–9257.

Carol Whitney and J Grainger. 2004. Does the huamn mnid raed wrods as a wlohe. *Trends in cognitive sciences*, 8:58–59.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP: System Demonstrations*, pages 38–45.

Baosong Yang, Zhaopeng Tu, Derek F Wong, Fandong Meng, Lidia S Chao, and Tong Zhang. 2018. Modeling localness for self-attention networks. In *Proceedings of EMNLP*, pages 4449–4458.

Jiahao Zhao, Penghui Wei, and Wenji Mao. 2021. Robust neural text classification and entailment via mixup regularized adversarial training. In *Proceedings of SIGIR*, pages 1778–1782.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In *Proceedings of ICLR*.