On Adversarial Examples for Character-Level Neural Machine Translation

Javid Ebrahimi, Daniel Lowd, Dejing Dou
Computer and Information Science Department, University of Oregon, USA
{javid, lowd, dou}@cs.uoregon.edu

Abstract

Evaluating on adversarial examples has become a standard procedure to measure robustness of deep learning models. Due to the difficulty of creating white-box adversarial examples for discrete text input, most analyses of the robustness of NLP models have been done through black-box adversarial examples. We investigate adversarial examples for character-level neural machine translation (NMT), and contrast black-box adversaries with a novel white-box adversary, which employs differentiable string-edit operations to rank adversarial changes. We propose two novel types of attacks which aim to remove or change a word in a translation, rather than simply break the NMT. We demonstrate that white-box adversarial examples are significantly stronger than their black-box counterparts in different attack scenarios, which show more serious vulnerabilities than previously known. In addition, after performing adversarial training, which takes only 3 times longer than regular training, we can improve the model's robustness significantly.

1 Introduction

Last year, a mistranslation by Facebook's machine translation (MT) system led to a wrongful arrest (Berger, 2017). Instead of translating an Arabic phrase to "good morning," Facebook's MT translated it as "attack them." Arabic is a morphologically-rich language, and the MT mistook the input word for another which differs from the input by only one character. As MT is used more and more, it is increasingly important to understand its worst-case failures to prevent incidents like this.

Adversarial examples are inputs designed to make a machine learning model perform poorly, and are often constructed by manipulating real-world examples (Goodfellow et al., 2015). Belinkov and Bisk (2018) investigate the sensitivity of neural machine translation (NMT) to synthetic and natural noise containing common misspellings. They show that state-of-the-art models are vulnerable to adversarial attacks even after a spell-checker is deployed. By performing ensemble adversarial training (Tramèr et al., 2018), they improve an NMT's robustness to adversarial noise.

We explore the space of adversarial examples for NMT in two directions: first, we study untargeted adversarial examples in a *white-box* setting, wherein the adversary has access to model parameters and can use its gradients to inflict more damaging manipulations for a larger decrease in the BLEU score; second, equipped with the developed machinery to do white-box attacks, we can perform more interesting attacks. We propose *controlled* and *targeted* adversaries which create adversarial examples with other goals, instead of merely decreasing the BLEU score. A controlled adversary aims to mute a word in the original translation, while a targeted adversary aims to push a word into it. Table 1 shows one example of each category. In both cases, the adversary, which has no word alignment model, has not drastically changed the rest of the translation, and has been able to reach its goals.

There is growing interest in understanding vulnerabilities of NLP systems (Jia and Liang, 2017; Zhao et al., 2018; Belinkov and Bisk, 2018). Previous work in NLP has focused on creating adversarial examples in a *black-box* setting, wherein the attacker can query a model but does not have access to its

src	1901 wurde eine Frau namens Auguste in eine medizinische Anstalt in Frankfurt gebracht.
adv	1901 wurde eine Frau namens **Afuiguste** in eine medizinische Anstalt in Frankfurt gebracht.
src-output	In 1931, a woman named **Augustine** was brought into a medical institution in France.
adv-output	In 1931, a woman named Rutgers was brought into a medical institution in France.
src	Das ist Dr. Bob Childs – er ist Geigenbauer und Psychotherapeut.
adv	Das ist Dr. Bob Childs – er ist Geigenbauer und **Psy6hothearpeiut**.
src-output	This is Dr. Bob Childs – he's a wizard maker and a **therapist**'s **therapist**.
adv-output	This is Dr. Bob Childs – he's a brick maker and a **psychopath**.

Table 1: Controlled and Targeted Attack on DE→EN NMT. In the first example, the adversary wants to suppress a person's name, and in the second example, to replace occurrences of *therapist* with *psychopath*

parameters. Black-box attacks often rely on heuristic methods to create adversarial examples. In contrast, white-box attacks approximate the *worst-case* attack for a particular model and input, within some allowed set of perturbations. Therefore, white-box attacks can demonstrate and defend against a model's most serious vulnerabilities, which may not be discovered by black-box heuristics.

After exploring the space of adversarial examples for NMT and proposing new types of attacks, we focus on adversarial training to make our model more robust. We build on HotFlip (Ebrahimi et al., 2018), a recently-introduced white-box method for generating adversarial examples and performing adversarial training for text classification. At the core of HotFlip lies an atomic *flip* operation, which changes one character to another by using the gradients of the model with respect to the one-hot vector input. In this work, we extend it to include a broader set of attacks, and we also improve it with a better beam search heuristic and faster adversarial training.

Our contributions are as follows:

1. We use a gradient-based estimate, which ranks adversarial manipulations, and we search for adversarial examples using greedy search or beam search methods.

2. We propose two translation-specific types of attacks and provide a metric to evaluate adversaries in these scenarios. Our experiments show that white-box adversaries can be significantly stronger than black-box adversarial examples.

3. We investigate the robustness of models trained with white-box adversarial examples and compare their robustness with black-box trained models.

2 Related Work

The need to understand vulnerabilities of NLP systems is only growing. Companies such as Google are using text classifiers to detect abusive language[1], and concerns are increasing over deception (Zubiaga et al., 2016) and safety (Chancellor et al., 2016) in social media. In all of these cases, we need to to better understand the dynamics of how NLP models make mistakes on unusual inputs, in order to improve accuracy, increase robustness, and maintain security or privacy. While this line of research has recently received a lot of attention in the deep learning community, it has a long history in machine learning, going back to adversarial attacks on linear spam classifiers (Dalvi et al., 2004; Lowd and Meek, 2005).

Character-level NMT systems (Lee et al., 2017; Costa-Jussa and Fonollosa, 2016) and those based on sub-word units (Sennrich et al., 2016) are able to extract morphological features which can generalize unseen words. Belinkov and Bisk (2018) show that character-level machine translation systems are overly sensitive to random character manipulations, such as keyboard typos. They use black-box heuristics to generate character-level adversarial examples, without using the model parameters or gradients to generate adversarial examples. The major challenge in creating white-box adversarial examples for text is that optimizing over discrete input is difficult (Miyato et al., 2017), which is why previous work has focused on black-box adversarial examples. Zhao et al. (2018) search for black-box adversarial examples in the space of encoded sentences and generate adversarial examples by perturbing the latent representation until the model is tricked. However, it is not clear how many queries are sent to the model or what the

[1] https://www.perspectiveapi.com

success rate of the adversary is. We contrast black-box and white-box attacks and show how white-box attacks significantly outperform their black-box counterparts in controlled and targeted attack scenarios.

Adversarial training/regularization interleaves training with generation of adversarial examples (Goodfellow et al., 2015). Concretely, after every iteration of training, adversarial examples are created and added to the mini-batches. This technique has been used for text classification (Miyato et al., 2017) using adversarial noise on the word embeddings without creating real-world adversarial examples. Jia and Liang (2017) point out the difficulty of adversarial training with real-world adversarial examples, as it is not easy to create such examples efficiently. In this work, we improve the training time of HotFlip (Ebrahimi et al., 2018) by using a one-shot attack in our inner adversary, which makes the running time of adversarial training only 3 times slower than regular training.

3 White-Box Adversarial Examples

Editing text to trick an NLP model, constrained by the number of characters to change, r, is a combinatorial search problem. We develop a gradient-based optimization method to perform four text edit operations: namely, flip (replacing one character with another), swap (replacing two adjacent characters with each other), delete, and insert. We use derivatives with respect to one-hot representation of the input, to rank candidate changes to text, in order to search for an adversarial example which satisfies the adversary's goal. Compared with a black-box adversary, our method has the overhead of sorting or searching among derivatives, while being considerably more successful.

3.1 Definitions

We use $J(\mathbf{x}, \mathbf{y})$ to refer to the log-loss of the translation model on source sequence $\mathbf{x}$ and target sequence $\mathbf{y}$. Let V be the alphabet, $\mathbf{x}$ be a text of length L characters, and $x_{ij} \in \{0,1\}^{|V|}$ denote a one-hot vector representing the j-th character of the i-th word. The character sequence can be represented by

$$\mathbf{x} = [(x_{11},.. \; x_{1n});(x_{21},.. \; x_{2n});...(x_{m1},.. \; x_{mn})]$$

wherein a semicolon denotes explicit segmentation between words. The number of words is denoted by m, and n is the number of maximum characters allowed for a word[2].

3.2 Derivatives of Operations

We represent text edit operations as vectors in the input space, and estimate the change in loss by directional derivatives with respect to these operations. Based on these derivatives, the adversary can choose the best loss-increasing operation. Our algorithm requires just one function evaluation (forward pass) and one gradient computation (backward pass) to estimate the best possible flip.

A **flip** of the j-th character of the i-th word ($a \rightarrow b$) can be represented by this vector:

$$\vec{v}_{ijb} = (\vec{0},..;(\vec{0},..(0,..\text{-}1,0,..,1,0)_j,..\vec{0})_i; \vec{0},..)$$

where -1 and 1 are in the corresponding positions for the a-th and b-th characters of the alphabet, respectively, and $x_{ij}^{(a)} = 1$. A first-order approximation of change in loss can be obtained from a directional derivative along this vector:

$$\nabla_{\vec{v}_{ijb}} J(\mathbf{x}, \mathbf{y}) = \nabla_x J(\mathbf{x}, \mathbf{y})^T \cdot \vec{v}_{ijb} = \frac{\partial J}{\partial x_{ij}}^{(b)} - \frac{\partial J}{\partial x_{ij}}^{(a)} \tag{1}$$

Using the derivatives as a surrogate loss, we simply need to find the best change by **maximizing** eq. 1, to *estimate* the best character change ($a \rightarrow b$). This requires searching in $|V|mn$ values for a given text of m words with n characters each, in a vocabulary of size $|V|$.

Similarly, character **insertion**, **deletion**, and **swap** of adjacent characters can be represented as vectors which carry the information about the direction and number of flips. Since the magnitudes of operation vectors are different, we normalized them by both L_1 and L_2 norm but found it had little impact.

A black-box adversary can perform similar manipulations which would be randomly picked. For example, Belinkov and Bisk (2018) define `Key` operation, which flips one character with an adjacent one on the keyboard, at random.

[2]Padding is applied if the number of characters is fewer than the maximum.

3.3 Controlled and Targeted Attacks

The previous adversary was untargeted, and its only goal was to increase the loss of the model. However, some corruptions of the output may be much worse than others – translating "good morning" as "attack them" is much worse than translating it as "fish bicycle." By changing the loss function, we can force the adversary to focus on more specific goals.

In a *controlled attack*, the adversary tries to remove a specific word from the translation. This could be used to maintain privacy, by making more sensitive information harder to translate, or to corrupt meaning, by removing key modifiers like "not," "joked,"' or "kidding." Concretely, we maximize the loss function $J(x, y_t)$, where t is the target word. This way, the adversary ignores the rest of the output and focuses on parts of the input that would affect the target word most.

In a *targeted attack*, the adversary aims to not only mute a word but also replace it with another. For example, changing the translation from "good morning"' to "good attack" could lead to an investigation or an arrest. Making specific changes like this is much more dangerous, but also harder for the adversary to do. For this attack, we maximize the loss $-J(x, y_{t'})$, where $t^{'}$ is the new word chosen to replace t. Note that the negation makes this equivalent to minimizing the predictive loss $J(x, y_{t'})$ on $t^{'}$. We represent this as maximization so that it fits in the same framework as the other attacks. Our derivative-based approach from the previous subsection can be then used directly to generate these new attacks, simply by substituting the alternate loss function.

3.4 Multiple Changes

We explained how to estimate the best single change in text to get the maximum increase/decrease in loss. We now discuss approaches to perform multiple changes.

(a) **one-shot:** In this type of attack, the adversary manipulates all the words in the text with the best operation in parallel. That is, the best operation for each word is picked locally and independently of other words. This is efficient, as with only one forward and backward pass, we can collect the gradients for all operations for all words. In addition, compared with the global one-shot method of Ebrahimi et al. (2018), it does not require sorting the gradients globally, and can further reduce the time to create adversarial examples. It is less optimal than the next approaches, which apply changes one by one. Due to its efficiency, this is the approach we choose to do adversarial training. Our experiments in section 6, which are untargeted, follow this approach. We also investigate black-box and white-box variants of one-shot attacks in section 5.1. The budget for the adversary is the number of words, and it is spent in the first shot.

(b) **Greedy:** In this type of attack, after picking the best operation in the whole text, we make another forward and backward pass, and continue our search. Our controlled attack in section 5.2 follows this approach, where we allow a maximum of 20% of the characters in text as the budget for the adversary. As will be explained in 5.2, the adversary spends much less than this amount.

(c) **Beam Search**: And finally, we can strengthen our greedy search by beam search. Our beam search requires only $\mathcal{O}(br)$ forward passes and an equal number of backward passes, with r being the budget and b, the beam width. At every step, the beam will be sorted by the sum of the true loss up to that point, which we have computed, plus the gradient-based estimate of candidate operations. This showed better performance than using the sum of the gradients in the path for sorting the beam, as was previously done (Ebrahimi et al., 2018). Since targeted attacks are the most difficult type of attack, we use this strategy for targeted attacks, as described in section 5.3. We allow a maximum of 20% of the characters in text as the budget for the adversary, and set the beam width to 5.

4 Experiments

We use the TED talks parallel corpus prepared by IWSLT 2016 (Mauro et al., 2016) for three pairs of languages: German to English, Czech to English, and French to English. We use the development sets and test sets of previous years except 2015 as our development set. The statistics of the dataset can be

Pair	Train	Test	Target vocab.
FR-EN	235K	1.1k	69k
DE-EN	210K	1.1k	66k
CS-EN	122K	1.1k	49k

Table 2: Data Statistics

found in Table 2. Throughout our experiments, we only allow character changes if the new word does not exist in the vocabulary, to avoid changes that an MT would respond to as expected. For example, it is not surprising that changing the source German word "nacht" to "nackt" would cause an MT to introduce the word "nude" in the translation.

The architecture we study was first proposed by Kim et al. (2016) for language modeling, and later adapted by Costa-Jussa and Fonollosa (2016) for translation, and is also used by Belinkov and Bisk (2018) for their adversarially-trained models. In this architecture, feature extraction is performed by convolutions over characters, which are passed to layers of highway networks, and finally given to stacks of recurrent neural nets for modeling a sequence of words. Using this architecture, the BLEU scores on our datasets are competitive with the submissions to the IWSLT (Mauro et al., 2016). Our implementation[3] relies largely on Yoon Kim's seq2seq implementation[4], with similar hyper-parameters, which mostly follows the guidelines of Luong et al. (2015) for attentional translation.

For experiments in section 6, where we report the BLEU score for a vanilla model and several adversarially-trained models against different attackers, we use a decoder with a beam width of 4. However, our white-box attacker uses a model with greedy decoding to compute gradients. The reason is that in order to calculate correct gradients, we either need to use greedy decoding, or use models which incorporate beam search in the decoder architecture such that gradients could flow in the beam paths, too (Wiseman and Rush, 2016), which incur more computational cost. Correct gradients are more of an important issue for targeted attacks, where we want to achieve a goal beyond simply breaking the system. For the sake of consistency, we use greedy decoding for both the vanilla model which is being attacked, and the white-box attacker in all experiments of section 5, where we contrast white-box and black-box adversaries in different scenarios.

5 Analysis of Adversaries

We first study whether first-order approximation gives us a good estimator to be employed by white-box adversaries. Figure 1 compares the true increase in log-loss (i.e., $J(x+v) - J(x)$), with our gradient-based estimate (i.e., $\nabla_v J(x)$). We create adversarial examples using the best estimated character flip for every word, over the German test set. Then, we compare the true increase in loss for the created adversarial examples, with our gradient-based estimate. The log-loss is evaluated by summing the log-loss of individual words, and similarly, the gradient-based estimate is the sum of all gradients given by flips which are performed once on every word. Figure 1.a plots the histograms for both of these measures, and Figure 1.b shows a scatter plot of them with the least squares fitting line. Due to linearization bias of the first-order approximation, we have a distribution with smaller variance for the gradient-based estimate measure. We can also observe a moderately positive correlation between the two measures (Spearman coefficient $\rho = 0.61$), which shows we can use the gradient-based estimate for *ranking* adversarial manipulations.

Next, we contrast black-box and white-box adversaries in untargeted, controlled, and targeted scenarios, and demonstrate that white-box adversaries significantly outperform black-box adversaries especially in controlled and targeted scenarios.

5.1 Untargeted Attack

Table 3 shows the BLEU score after one-shot white-box and black-box attacks are performed. Unlike delete and swap, insert and flip have the advantage of making changes to one-letter words, so we expect

[3]https://github.com/jebivid/adversarial-nmt

[4]https://github.com/harvardnlp/seq2seq-attn

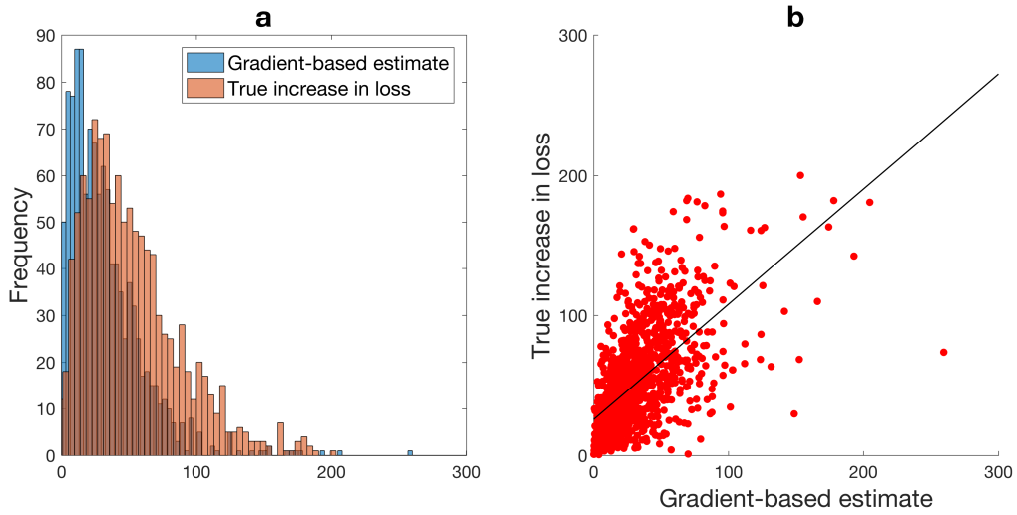

Figure 1: Comparing the distribution of the true increase in loss and its gradient-based estimate, and their correlation, using best flips for each word in a sentence of the German test set.

them to perform better. We see this for the white-box attacks which can pick the best change to every word using the gradients. On the contrary, a black-box adversary performs worst for flip, which is because the black-box attacker is not enabled to pick the best change when more options (possible character flips) are available, as opposed to swap and delete which are governed by the location of the change and contain no additional flip. Nevertheless, a black-box adversary has competitive performance with the white-box one, even though it is simply randomly manipulating words. We argue that evaluating adversaries, based on their performance in an untargeted setting on a brittle system, such as NMT, is not appropriate, and instead suggest using goal-based attacks for evaluation.

Attack	Flip		Insert		Delete		Swap	
	white	black	white	black	white	black	white	black
FR	**4.27**	6.98	**4.74**	4.85	**4.99**	5.86	**4.87**	5.20
DE	**4.50**	6.87	**3.91**	4.31	**5.63**	5.73	4.94	**4.74**
CS	**4.31**	6.09	**4.66**	5.86	**6.30**	6.62	6.05	**5.82**

Table 3: BLEU score after greedy decoding in the existence of different types of untargeted attacks.

5.2 Controlled Attack

We introduce more interesting attacks, in which the adversary targets the MT for more specific goals. A perfect *mute* attack removes a word successfully and keeps the rest of the sentence intact. For example, consider the translation T, containing words w_1, w_2,..., w_t, ...w_n, where w_t is the target word. A perfect mute attack will cause the NMT to create a translation T_p which contains words w_1, w_2,..., UNK, ...w_n, wherein w_t is replaced with UNK. With this observation in mind, we define the success rate of an attack, which generates T_{adv}, as follows:

$$\text{success}(T_{adv}) = \begin{cases} 1, & \text{if } \frac{\text{BLEU}(T, T_{adv})}{\text{BLEU}(T, T_p)} \geq \alpha \\ 0, & \text{otherwise} \end{cases}$$

We can control the quality of an attack with α, for which a larger value punishes the adversary for ad-hoc manipulations, which could cause the NMT to generate a radically different and possibly gibberish translation. Success rate is defined by the number of successful attacks divided by the number of total sentences in the test set. Figure 2 plots the success rate against α. As can be seen, the white-box adversary is significantly more successful than a black-box adversary. By taking advantage of the knowledge of

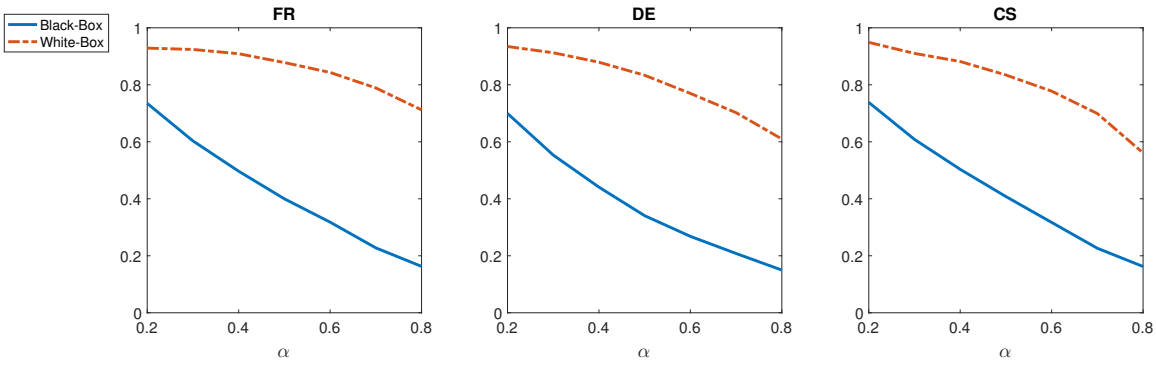

Figure 2: Success rate of white-box and black-box adversaries in a controlled setting as a function of α.

gradients of the model, a white-box adversary can perform better targeted attacks. For this experiment, the black-box attacker uniformly picks from the four possible changes and randomly applies them.

For this attack, we follow the greedy approach, and use a budget of 20% of the characters in text. Table 4 shows the average number of character changes and the number of queries made to the model. The reported character changes are for attacks wherein the attacker only muted a target word successfully, regardless of the quality of the translation. The reported number of queries takes the unsuccessful trials into account too. The white-box adversary is more efficient due to fewer queries and fewer manipulated characters, which can be crucial for a real-word adversary. Nevertheless, unlike a black-box adversary, a white-box adversary requires additional backward passes, and has the overhead of operations on the gradient values, mainly sorting. This makes the running times of the two comparable.

Compared with the results in the previous section, controlled attacks show a more convincing superiority of the white-box attacks over black-box attacks.

	Character Changes		Queries	
source	white	black	white	black
FR	**1.9**	7.7	**2.3k**	8.9k
DE	**1.9**	6.5	**1.9k**	7.8k
CS	**1.5**	5.3	**1.2k**	6.1k

Table 4: Efficiency of attacks.

5.3 Targeted Attack

A more challenging attack is to not only mute a word but also replace it with another one. The evaluation metric for targeted attack is similar to controlled attack with one difference that a perfect *push* attack produces a translation, T_p, which contains words w_1, w_2, $w_{t'}$,..w_n, wherein $w_{t'}$ has replaced w_t. In classification domains with few classes, targeted attacks are relatively simple, since an adversary can perturb the input to move it to the other side of a decision boundary. Whereas in MT, we deal with vocabulary sizes in the order of at least tens of thousands, and it is less likely for an adversary to be successful in targeted attacks for most possible target words. To address this, we evaluate our adversary with n^{th}-most likely class attacks. In the simplest case, we replace a target word with the second-most likely word at decoding time.

As can be seen in Table 3, targeted attacks are much more difficult with a much lower success rate for the adversary. Nevertheless, the white-box adversary still performs significantly better than the black-box adversary. The success rate dramatically goes down for large values of n. For example, for the value of 100, the success rate will be more than ten times smaller than second-most likely attack.

5.4 Some Adversarial Examples

Table 5 shows three adversarial examples. The first example shows a controlled attack, where the adversary has successfully removed a swear word from the sentence. The BLEU ratio, used in our success

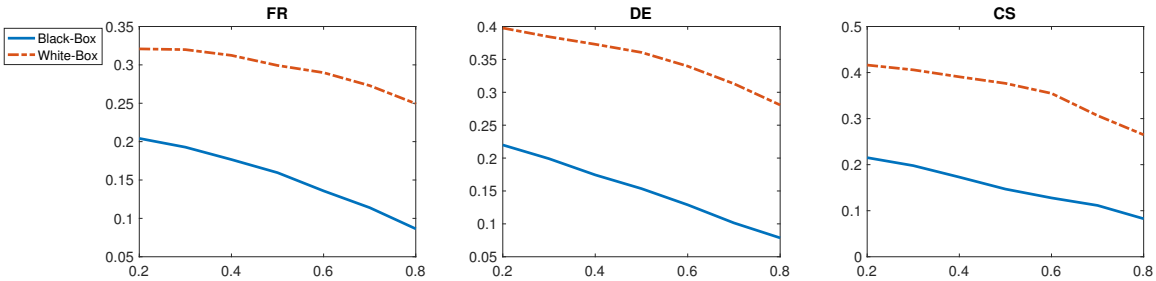

Figure 3: Success rate of white-box and black-box adversaries in the second-most likely targeted attack as a function of α.

src	Wir erwarten Perfektion von Feministinnen, weil wir immer noch für so viel kämpfen, so viel wollen, so verdammt viel brauchen.
adv	Wir erwarten Perfektion von Feministinnen, weil wir immer noch für so viel kämpfen, so viel wollen, so **öberdammt** viel brauchen.
src-output	We expect perfection from feminist, because we're still fighting so much, so **damn**, so **damn**, so **damn**.
adv-output	We expect perfection from feminist, because we still fight for so much, so much of all, so much of all that needs to be.
src	In den letzten Jahren hat sie sich zu einer sichtbaren Feministin entwickelt.
adv	In den letzten Jahren hat sie sich zu einer sichtbaren **FbeminisMin** entwickelt.
src-output	In the last few years, they've evolved to a safe **feminist**.
adv-output	In the last few years, they've evolved to a safe **ruin**.
src	Ein Krieg ist nicht länger ein Wettbewerb zwischen Staaten, so wie es früher war.
adv	Ein Krieg ist nicht länger ein **erkBkaSzeKLlWmrt** zwischen Staaten, so wie es früher war.
src-output	A war is no longer a **competition** between states, like it used to be.
adv-output	A war is no longer a **throwaway** planet between states, as it used to be.

Table 5: A controlled attack and two targeted attacks on our DE-EN NMT. First example shows a controlled attack, the second and third examples show a second-most and a 100_{th}-most likely targeted attack, respectively.

rate measure, for this example is 0.52. The second example shows a second-most likely targeted attack where the new translation has managed to keep the rest of the translation intact and achieve its goal. The BLEU ratio for this example is 1.00. The third example, which has a BLEU ratio of 0.70, shows a 100^{th}-most likely attack, where the word *competition* is replaced with *throwaway*. Due to the difficulty of this change, the adversary has committed a considerably larger number of manipulations.

6 Robustness to Adversarial Examples

6.1 Baselines

We use the black-box training method of Belinkov and Bisk (2018) as our baseline. They train several models using inputs which include noise from different sources. We used their script[5] to generate random (`Rand`), keyboard (`Key`), and natural (`Nat`) noises. Their best model was one which incorporated noise from all three sources (`Rand+Key+Nat`).

Similar to their approach, we train a model which incorporates noisy input scrambled by Flips, Inserts, Deletes, and Swaps in training (`FIDS-B`). Since natural noise was shown to be the most elusive adversarial manipulation (Belinkov and Bisk, 2018), we used this source of noise to determine the proportion of each of the FIDS operations in training. Concretely we found that the majority of the natural noise can be generated by FIDS operations, and we used the ratio of each noise in the corpora to sample from these four operations. Figure 4 shows the distribution of manipulations for each language. A single swap is the least likely operation in all three languages[6]. FIDS operations account for 64%, 80%, and 70% of natural noise for Czech, German, and French, respectively. This can be regarded as a background knowledge incorporated into the adversary.

[5] https://github.com/ybisk/charNMT-noise

[6] Excluding single swaps from manipulations with two flips.

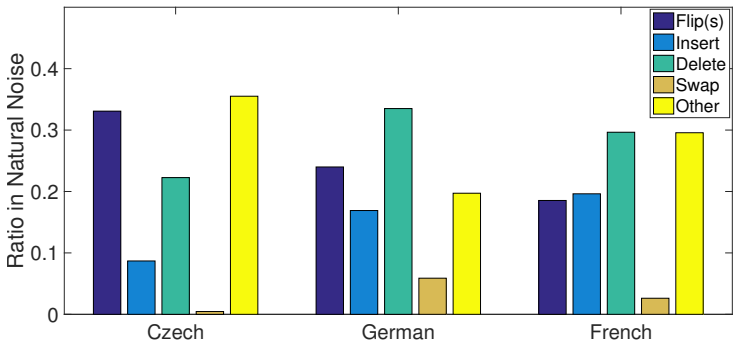

Figure 4: Distribution of types of noise in the natural noise corpora.

6.2 White-Box and Ensemble Methods

Our white-box adversary `FIDS-W` generates adversarial examples using our four text edit operations in accordance with the distribution of operations on the natural noise. At every epoch, adversarial examples are generated for every mini-batch using the one-shot approach. More precisely, all words in the sentence are changed by a single FIDS operation in parallel. While training on both clean and adversarial examples has been the standard approach in adversarial training, some evidence in computer vision (Madry et al., 2017; Shaham et al., 2015) suggests that training on white-box adversarial examples alone can boost models' robustness to adversarial examples, with a minor decrease in accuracy on clean examples. However, we found that in order to get a good BLEU score on the clean dataset, we need to train on both clean and white-box adversarial examples. We also train an ensemble model, `Ensemble`, which incorporates white-box and black-box adversarial examples, with 50/50 share for each. The black-box adversarial examples come from `Nat` and `Rand` sources.

When evaluating models against `White` adversarial examples at test-time, we use the test set which corresponds to their method of training. For instance, the `White` adversarial examples for the `Rand` model, come from the test set which has manipulated clean examples by `Rand` noise first. For `Vanilla`, `FIDS-W`, and `Ensemble` models, the adversarial examples are generated from clean data. This makes the comparison of models, which are trained on different types of data, fair. We use the one-shot attack for our white-box attack evaluation, using the same distribution based on natural noise.

6.3 Discussion

Table 6 shows the results for all models on all types of test data. Overall, our ensemble approach performs the best by a wide margin. As expected, adversarially-trained models usually perform best on the type of noise they have seen during training. However, we can notice that our `FIDS-W` model performs best on the `Nat` noise amongst models which have not been trained on this type of noise. Similarly, while `FIDS-W` has not directly been trained on `Key` noise, it is trained on a more general type of noise, particularly flip, and thus can perform significantly better on the `Key` than on other models which also have not been trained on this type of noise. However, it cannot generalize to `Rand`, which is an extreme case of attack, and we need to use an ensemble approach to perform well on it too. Nevertheless, `FIDS-W` performs best on the `Rand` noise, compared with models which are not trained on `Rand` either. This validates our earlier claim that training on white-box adversarial examples, which are harder adversarial examples, can make the model more robust to weaker types of noise. We also observe that `FIDS-B` performs better on the `White` examples compared with other baselines; although it has not been trained on white-box adversarial examples, it is trained on black-box adversarial examples of the same family of FIDS operations.

Figure 5 shows the training loss of white-box adversarially-trained model on adversarial examples. The model is getting more resilient to adversarial examples, which are created at the start of each epoch.

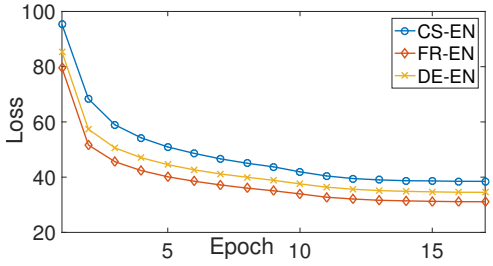

Figure 5: Training loss on adversarial examples for FIDS-W.

Training \ Test		Clean	Nat	Key	Rand	FIDS-B	FIDS-W	Avg.
French	Vanilla	37.54	19.17	12.12	4.75	6.85	5.36	15.95
	Nat	26.35	**33.23**	11.16	5.32	8.28	6.65	15.16
	Key	33.02	17.30	**35.97**	4.63	7.00	5.17	17.17
	Rand	36.06	18.54	8.31	**36.10**	8.76	7.14	19.14
	FIDS-B	34.48	21.59	28.48	6.82	32.62	13.60	22.92
	FIDS-W	37.15	23.65	31.18	7.78	**32.72**	**31.94**	27.40
	Rand+Key+Nat	34.55	30.74	32.82	34.01	12.05	7.08	25.20
	Ensemble	**37.81**	30.27	29.36	34.42	32.00	30.01	**32.30**
German	Vanilla	31.81	17.24	10.36	4.20	6.78	5.50	12.64
	Nat	24.89	**32.14**	10.22	4.61	7.53	5.99	14.23
	Key	27.20	15.98	**30.62**	4.64	7.68	4.74	15.13
	Rand	31.01	17.90	6.59	**30.70**	9.19	5.83	16.86
	FIDS-B	28.27	20.22	23.84	6.29	27.35	10.79	19.45
	FIDS-W	**31.81**	21.72	26.23	7.75	**27.38**	**26.51**	23.56
	Rand+Key+Nat	29.22	29.78	27.83	28.88	10.30	6.14	22.01
	Ensemble	31.54	31.11	23.91	28.95	26.38	25.06	**27.82**
Czech	Vanilla	**26.44**	13.55	9.49	4.78	7.30	5.93	11.24
	Nat	18.73	**23.06**	9.07	4.45	7.36	5.42	11.34
	Key	22.76	13.09	**23.79**	4.83	7.93	5.82	13.03
	Rand	24.23	12.00	7.26	**24.53**	7.24	5.47	13.45
	FIDS-B	22.31	14.15	17.91	6.48	19.67	8.60	14.84
	FIDS-W	25.53	15.57	19.74	7.18	**20.02**	**19.42**	17.90
	Rand+Key+Nat	22.21	20.59	20.60	21.33	10.06	5.89	16.77
	Ensemble	25.45	20.46	17.15	21.39	18.52	17.03	**19.99**

Table 6: BLEU score of models on clean and adversarial examples, using a decoder with beam size of 4. The best result on each test set is shown in bold. FIDS-W performs best on all noisy test sets compared with models which have not been trained on that particular noise (shown in red). FIDS-B performs best on white-box adversarial examples compared with other black-box trained models (shown in blue).

7 Conclusion and Future Work

As MT methods become more effective, more people trust and rely on their translations. This makes the remaining limitations of MT even more critical. Previous work showed that NMT performs poorly in the presence of random noise, and that its performance can be improved through adversarial training. We consider stronger adversaries which are attacking a specific model and may also have specific goals, such as removing or changing words. Our white-box optimization, targeted attacks, and new evaluation methods are a step towards understanding and fixing the vulnerabilities in NMT: we're able to find more effective attacks and train more robust models than previous black-box methods. Next steps include exploring other types of targeted attacks, such as attacks that target more than one word, and other types of constraints, such as better characterizing which character changes affect intelligibility the least.

8 Acknowledgment

This work was funded by ARO grant W911NF-15-1-0265.

References

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

Yotam Berger. 2017. Israel arrests Palestinian because Facebook translated 'good morning' to 'attack them'. Retrieved from `https://www.haaretz.com`.

Stevie Chancellor, Jessica Annette Pater, Trustin Clear, Eric Gilbert, and Munmun De Choudhury. 2016. # thyghgapp: Instagram content moderation and lexical variation in pro-eating disorder communities. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*, pages 1201–1213. ACM.

Marta R Costa-Jussa and José AR Fonollosa. 2016. Character-based neural machine translation. In *Proceedings of ACL*.

Nilesh Dalvi, Pedro Domingos, Sumit Sanghai, Deepak Verma, et al. 2004. Adversarial classification. In *Proceedings of KDD*, pages 99–108.

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

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

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

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. *Proceedings of AAAI*.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *TACL*.

Daniel Lowd and Christopher Meek. 2005. Adversarial learning. In *Proceedings of KDD*, pages 641–647.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP*.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Cettolo Mauro, Niehues Jan, Stüker Sebastian, Bentivogli Luisa, Cattoni Roldano, and Federico Marcello. 2016. The iwslt 2016 evaluation campaign. In *International Workshop on Spoken Language Translation*.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In *Proceedings of ICLR*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. *Proceedings of ACL*.

Uri Shaham, Yutaro Yamada, and Sahand Negahban. 2015. Understanding adversarial training: Increasing local stability of neural nets through robust optimization. *arXiv preprint arXiv:1511.05432*.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick McDaniel. 2018. Ensemble adversarial training: Attacks and defenses. In *Proceedings of ICLR*.

Sam Wiseman and Alexander M Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of EMNLP*.

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

Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. 2016. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PloS one*, 11(3):e0150989.