Improving the Robustness of Low-Resource Neural Machine Translation with Adversarial Examples

Shuo Sun, Hongxu Hou(✉), Nier Wu, Zongheng Yang, Yisong Wang, Pengcong Wang, and Weichen Jian

Inner Mongolia Key Laboratory of Mongolian Information Processing Technology, National & Local Joint Engineering Research Center of Intelligent Information Processing Technology for Mongolian, College of Computer Science, Inner Mongolia University, Hohhot, China

cshhx@imu.edu.cn

Abstract. Weak robustness and noise adaptability are major issues for Low-Resource Neural Machine Translation (NMT) models. That is, once some tiny perturbs are added to the input sentence, the model will produce completely different translation with high confidence. Adversarial example is currently a major tool to improve model robustness and how to generate an adversarial examples that can degrade the performance of the model and ensure semantic consistency is a challenging task. In this paper, we adopt reinforcement learning to generate adversarial example for low-resource NMT. Specifically, utilizing the actor-critic algorithm to modify the source sentence, the discriminator and translation model in the environment are used to determine whether the generated adversarial examples maintain semantic consistency and the overall deterioration of the model. Furthermore, we also install a language model reward to measure the fluency of adversarial examples. Experimental results on low-resource translation tasks show that our method highly aggressive to the model while maintaining semantic constraints greatly. Moreover, the model performance is significantly improved after fine-tuning with adversarial examples.

Keywords: Reinforcement learning · Adversarial example · Low-resource NMT

1 Introduction

Neural Machine Translation (NMT) [2,16,17] has made significant progress. However, even the best trained translation models still make unpredictable errors in practical applications [3]. Figure 1 illustrates the fragility of NMT. Robustness is the feature that a model can maintain some performance despite perturbations or noise. For machine translation tasks, robustness refers to the ability of the model to adapt to new corpus. The lack of model training and noise learning ability leads to the model generating a completely different translation after adding certain perturbations to the sentence, which seriously affects the model performance. The original method improves

T. Xiao and J. Pino (Eds.): CCMT 2022, CCIS 1671, pp. 60–71, 2022.
https://doi.org/10.1007/978-981-19-7960-6_7

the model robustness by manually compiling error features [7,18], but it is too costly and some features are inapplicable for tasks of machine translation.

Adversarial example is a momentous tool for exploring the robustness of deep learning systems and it's initially applied in the field of computer vision [14]. Recently, some researchers utilizing adversarial examples to Natural Language Processing (NLP) tasks [4,5,19], which currently include character-level, word-level, phrase-level, and sentence-level adversarial examples. It makes the model produce error output by adding carefully designed perturbations to the input data. In general, the generation of adversarial examples implies that the model uses non-robust features, resulting in a less robust model as well. Adversarial training is performed by data augmentation methods, where adversarial examples are proportionally blended into the training set. In this way, the model obtained by training on the new dataset learns these non-robust features, resulting in a more robust model. Thus for machine translation tasks, we can quickly obtain a large amount of parallel data that can be applied for robustness analysis by using the input of an existing parallel corpus to generate adversarial examples along the output of the source text.

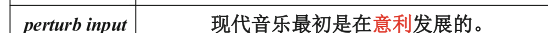
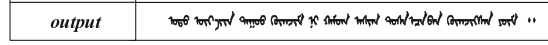
input	现代音乐最初是在意大利发展起来的。
output	
perturb input	现代音乐最初是在意利发展的。
output	

Fig. 1. Fragility of neural machine translation. Leaving out a Chinese character " 大 " and word "起来 " lead to significant errors in Mongolian translation.

However, unlike images that directly use the gradient optimization to obtain adversarial examples, the sentence space in NLP is discrete, so it's difficult to disturb along the gradient update direction when generating adversarial examples for text. On the other hand, if the common noise introduction such as adding, deleting, and modifying words is used to perturb against the source input, the generated adversarial examples aren't only difficult to ensure sentence fluency and semantic consistency, but also may even degrade the model performance. Especially for low-resource translation tasks, its own lack of massively parallel corpus has led to poor model performance, poor robustness and weak adaptation to new corpus or sentences with noise. Therefore, to improve the robustness of low-resource translation tasks, this paper utilizes reinforcement learning to generate adversarial examples and uses discriminators as terminal signals in the environment to further constrain semantics. Furthermore, we also add a language model to evaluate the fluency of the adversarial examples. The method learns how to apply discrete perturbations at the token-level to directly reduce translation quality. The experimental results on the CCMT2019 Mongolian-Chinese and CWMT2017 Uighur-Chinese show that using the adversarial examples generated by this method to fine-tune the model can significantly improve its performance.

2 Background and Related Work

2.1 Neural Machine Translation

Neural machine translation (NMT) mainly utilizes the encoder-decoder structure to achieve semantic encoding of the source language and prediction of the target language. The specific way is to use an $Encoder$ to encode the input source language $x = (x_1, ..., x_n)$ into a fixed vector, and then use $Decoder$ to decode the vector to finally get the target language. For y_t, given its previous word sequence $y_{<t}$ and the source language sentence x, use $P(y|x)$ to determine the probability of the current target word $P(y_t|y_{<t}, x)$. The specific calculation process is shown in Eq. (1):

$$P(y_t|y_{<t}, x) \propto exp(y_t; r_t; C_t) \tag{1}$$

where r_t is the hidden layer state of the neural machine translation model $Encoder$ at time t. C_t is the context state information of the generated word y_t defined according to the hidden layer node state of $Encoder$. NMT is trained using Maximum Likelihood Estimation (MLE). Given J training sentence pairs $\left\{x^i, y^i\right\}_{i=1}^{N}$, at each time step, NMT generates the target word y_t by maximizing the translation probability on the source sentence x. The training objective is to maximize the Eq. (2):

$$L_{MLE} = \sum_{i=1}^{N} logp(y^i|x^i) = \sum_{i=1}^{N}\sum_{t=1}^{M} logp(y_t^i|y_1^i...y_{t-1}^i, x^i) \tag{2}$$

2.2 Adversarial Example, Adversarial Attack and Adversarial Training in NLP

Adversarial Example can be described as $\hat{x}$, which is obtained by adding a restricted perturbation of δ to the original input sample (x, y) and cause model deterioration. For an original sample (x, y), there exists its adversarial sample set $A(x, y)$, and its expression is shown in Eq. (3):

$$\mathcal{A}(x, y) = \{\hat{x}|R(\hat{x}, x) \leq \delta \wedge M(\hat{x}) \neq y\} \tag{3}$$

where $R(\hat{x}, x)$ represents the vector perturbed between the disturbed sample $\hat{x}$ and the original sample x. "Restricted perturbation" requires that $R(\hat{x}, x)$ to be constrained by δ. The model M is generally non-robust, which makes it possible that when the model inputs a sample $\hat{x}$ with minor perturbations, the resulting $M(\hat{x})$ is completely different from the original output $M(x)$. The generation of adversarial samples is usually associated with perturbations of non-robust features.

Adversarial Attack is the process of generating adversarial examples $\hat{x}_1, \hat{x}_2, ..., \hat{x}_n \in \mathcal{A}(x, y)$ against model M and (x, y) sample. It aims to look for the non-robust and useful features to perturb x, and finally make the model produce error output. According to the attack environment, adversarial attack can be divided into black-box attack and white-box attack. For black-box attack, the attack algorithm can only access the output of M without knowing the parameters and structure information. For white-box attack, the attack algorithm can access all the information and parameters of M and generate

adversarial examples based on the gradient of M to attack the network. In NLP task, due to the discreteness of sentence space, it is difficult to disturb effectively with gradient information, so white box attack is difficult.

Adversarial Training is the process in which adversarial examples are generated on the training set through adversarial attacks for data enhancement and the enhanced data is used to retrain model M, so it can be defined as an optimization problem, and the model is expected that both performance and robustness will be enhanced. The original non-robust features may become useless after adversarial training, thereby weakening the association between non-robust features and labels, and achieving the purpose of anti-disturbance model.

2.3 Genetic Algorithm-Based Adversarial Attack

GeneticAttack [1] is a black-box adversarial attack method that performs word-level perturbation on examples, and uses genetic algorithm to optimize the examples. Inspired by the theory of biological evolution, the core of genetic algorithm lies in population mutation, crossover and selection. The population of GeneticAttack consists of several sentences x, and the size of the population is limited by hyperparameters. The mutation operation is completed by synonym replacement, and synonyms are obtained through an independently obtained word-embedding matrix. In mutation operations, GeneticAttack additionally uses a language model to filter out inappropriate word substitutions. The crossover operation takes out two sentences in the population and randomly selects words from one of them in the position of each word to form a new sentence. The new sentence collection forms the next generation population. The selection fitness is the output $M(\hat{x})$.

2.4 Gradient-Based Adversarial Attack

HotFlip [6] is a typical white-box attack method, which uses gradient ascending to directly select the largest disturbance among the acceptable perturbations by limiting the degree of perturbation, thereby generating adversarial samples quickly and efficiently. HotFlip performs one-hot encoding on sentences, which are represented as a three-dimensional tensor, in which each word corresponds to a matrix, and each column in the matrix is a one-hot character vector. It has the advantage of allowing character substitutions to be represented using a tensor of the same size as the sentence, as well as a tensor representation of character substitutions from the gradient tensor. During character replacement, Hotflip directly selects the character closest to the gradient direction for replacement. The insertion and deletion operations are completed by character substitutions of words.

3 Adversarial Examples Based on Reinforcement Learning

Adversarial examples must be semantically consistent with the original input, while causing the model to produce incorrect output. GeneticAttack (Sect. 2.3) generates

adversarial examples through synonym substitution, and the resulting perturbations are often tiny in semantic space, but the algorithm cannot effectively use gradient information to efficiently generate perturbations. HotFlip (Sect. 2.4) uses gradient information to make attacks extremely efficient. However, it can only attack the character-level model and produce several meaningless words, which will greatly reduce the overall fluency of the sentence.

Therefore, this paper utilizes Reinforcement Learning (RL) to generate adversarial examples by combining the advantages of above two methods. According to [20], it is regarded as a restricted Markov Decision Process (MDP), which edits the tokens at each position in the source sentence from left to right. Each editing decision depends on the impact of the existing modification on the semantics and the degradation expectation of the system output. Furthermore, inspired by GeneticAttack [1], we also add a Language Model (LM) to measure the fluency of adversarial examples. The generation strategy of adversarial examples is obtained through the continuous interactive feedback of the degree of attack on the translation model and the fluency of the examples.

3.1 Reinforcement Learning

As an momentous branch in the field of machine learning, reinforcement learning aims to study the use of agents to conduct model training through interacting with the environment and receiving "feedback" information, so as to "automatically" decide the optimal solution [8]. Figure 6 illustrates the process of reinforcement learning. At each time t, $Agent$ receive state s_t from $Environment$, and $Agent$ make action a_t on basis of s_t, while act on $Environment$ to generate reward r_t. $Agent$ reach the new state s_{t+1} according to r_t. Figure 7(a) shows the overall framework of the model, in which the $Environment$ (Sect. 3.2) and the $Agent$ (Sect. 3.3) are two significant parts. $Agent$ learns to modify the token of each position in the original sample sequentially from left to right, and uses a discriminator in the $Environment$ to determine whether the modified sample is semantically consistent with the original sample, at the same time, inputs the modified sample into the language model and translation model to evaluate the fluency of adversarial examples and whether reach the deterioration of the model. The specific process is as in Fig. 7(b).

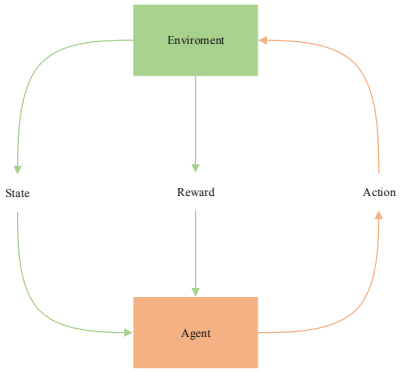
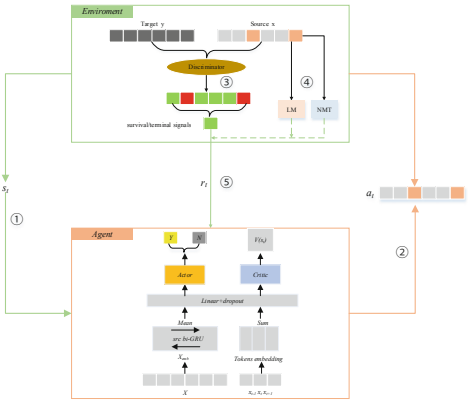


Fig. 2. Reinforcement learning.

3.2 Environment

This section details the environment state and the calculation of rewards.



(a)

Process 1 Generation of adversarial examples

① Environment states are processed as inputs for agent;
② Agent modifies the original samples using the Actor-Critic algorithm based on the state;
③ Utilizing the discriminator to determine the survival or termination signals and step reward of the environment;
④ Determine degradation with NMT and fluency of LM;
⑤ Update agent with total rewards.
Loop ①to ③and accumulate rewards for each step until environment terminates.

(b)

Fig. 3. The architecture and specific process of our method.

State. The state of the $Environment$ is described as $s_t = (x, t)$, where $x = (x^1, ..., x^N)$ are N sequences. Adding the begin and end tags (BOS and EOS) to each sequence $x^i = (x_1, x_2, ..., x_n)$ and padding them to the same length. $t \in (1, n)$ indicates the token position to be perturbed by $Agent$. $Environment$ will consecutively loop for all token positions and update s_t based on $Agent$'s modification. $Environment$ also yields reward signals until the end or intermediately terminated.

Reward Calculation. The reward r_t consists of a survival reward r_s for each step, a final degradation r_d and the fluency reward r_l when the $Agent$ survives till the end. Therefore, the reward for each time step is calculated as follows:

$$r_t = \begin{cases} -1, terminated \\ \frac{1}{N}\sum_N \alpha \cdot r_s, survive \wedge t \in [1, n) \\ \frac{1}{N}\sum_N (\alpha \cdot r_s + \beta \cdot r_d + \gamma \cdot r_l), survive \wedge t = n \end{cases} \tag{4}$$

where α, β γ and are hyper-parameters. Since the adversarial examples must maintain semantic consistency with the original examples, the $Agent$ must survive for its goal by also fooling the discriminator D, which determines terminal or survival signal by judging whether the modified sequence matches the original target translation y. Once D determines the pair as positive, its corresponding possibility is regarded as the reward, otherwise 0:

$$r_s = \begin{cases} P(positive|(\hat{x}, y); \theta_d), positive \\ 0, otherwise \end{cases} \tag{5}$$

When all sequences in x are intermediately terminated, the overall reward r_t yields -1. For example which is defined as "negative" during survival phase, it's subsequent rewards and actions will be disguised as zero. If the agent survives to the end, $Environment$ generates an additional average reward r_d and fluency reward r_l as the

final reward for the current training episode. For r_d, we adopt relative degradation [11]:

$$r_d = \frac{score(y, refs) - score(y^{'}, refs)}{score(y, refs)} \tag{6}$$

where y and $y^{'}$ denote original and perturbed output, $refs$ are references, and $score$ is a translation metric. If $score(y, refs)$ is 0 and return 0 as r_d.

For the sake of receiving smoother adversarial example, we use a language model to participate in the reward calculation in the $Environment$ so that the $Agent$ can consider the fluency of the examples when modifying the original examples. Typical language models have problems such as zero probability or statistical inadequacies. Katz smoothing is a probability formula to alleviate the "unsmoothness" problem. The basic idea of this method is if exists N-gram language model, directly using the discounted probability; If the higher-order language model is non-exist, the saved probability will be allocated according to the N-1 order language model probability, and so on. We adopt 3-gram and r_l represents the fluency score of sequence $\hat{x}$:

$$r_l = \sum_{t=1}^{n} P_{katz}\left(x_t | x_{t-2}^{t-1}\right) \tag{7}$$

$$P_{katz}\left(x_t | x_{t-2}^{t-1}\right) = \begin{cases} P_{ML}\left(x_t | x_{t-2}^{t-1}\right), & if \quad count\left(x_{t-2}^{t-1}\right) > 0 \\ \lambda P_{katz}^{(n-1)}\left(x_t | x_{t-1}\right), & if \quad count\left(x_{t-2}^{t-1}\right) = 0 \end{cases} \tag{8}$$

3.3 Agent

As it is shown in Fig. 7(a), $Agent$ uses Actor-Critic algorithm [9] to modify samples, the actor and critic share the same input layers and encoder. Actor takes in $Source$ and current token with its surrounding (x_{t-1}, x_t, x_{t+1}), then yields a binary distribution to determine whether to attack a token on step t, while critic emits a value $V(s_t)$ for every state. Once the $Actor$ determines that a token at specific location can be perturbed, it is replaced with one of the token's candidates within the distance of σ in the vocabulary. See [20] for more details about training and inferencing.

4 Experiment

4.1 Data Preprocessing

We test our adversarial example generations on Mongolian-Chinese (Mo-Zh) translation tasks of CCMT2019 and Uighur-Chinese (Ug-Zh) translation tasks of CWMT2017. In this paper, we use the open-source Chinese word splitting tool THULAC [10] to split the Chinese corpus, so that the corpus can be better adapted to the model and reduce the problem of poor model performance caused by word order to a certain extent. Moreover, Byte Pair Encoding (BPE) [13] encoding is used to process the Mongolian and Uighur corpus, which is firstly sliced into the corresponding smallest granularity. The training of the translation model is assisted by extracting the highest frequency character substrings into the newly generated dictionary. This approach slices the sentences into a granularity between words and characters, which can preserve the contextual semantics to a certain extent while alleviating the data sparsity problem.

4.2 NMT Model

This paper selects the state-of-the-art RNNSearch [2] and Transformer [15] as victim translation models. For RNN-Search, it's an encoder-decoder framework based on RNN, we set the hidden layer nodes and word-embedding dimensions to 512 and $dropout = 0.1$. By averaging the single model obtained from the last 20 checkpoints, we use adaptive to adjust the learning rate. For Transformer, we set $dropout = 0.2$ and the dimension of word embedding to 1024, with the learning rate and checkpoint settings consistent with RNNSearch.

4.3 Evaluating Indicator

We report de-tokenized BLEU with SacreBLEU [12] as the evaluation metric of adversarial examples and also test source semantic similarity with human evaluation (HE) ranging from 0 to 5 (Table 1) used by [11] by randomly sampling 20% of total sequences for a double-blind test.

Table 1. Human evaluation metrics.

0	Meaning of the two sentences is completely different
1	The topic is the same but the meaning is different
2	Some key messages are different
3	The key messages is the same, but the details differ
4	Meaning is essentially equal but some expressions are unnatural
5	Meaning is essentially equal and the expression is fluent and natural

4.4 Adversarial Attack Results and Analysis

We utilize GeneticAttack [1], HotFlip [6] and our method to generate adversarial examples for the test set respectively to attack the translation model. Table 2 illustrates the deterioration degree of adversarial examples to different translation tasks. We randomly select 20% of the adversarial examples for double-blind human evaluation to evaluate the semantic similarity between the adversarial example and the original example.

As it is shown in Table 2, GeneticAttack uses synonym replacement and genetic algorithm optimization to modify the original sample, resulting in less disturbance, but it lacks some semantic and fluency constraints compared with our method, which is easy to produce grammar problems, and low efficiency due to the inability to use gradient information effectively. Hotflip mainly uses gradient information to generate typo adversarial examples (such as " 香蕉 " → " 香交 ") to improve the model's ability to adapt and correct typos. Although the efficiency is high, it may produce some meaningless words which greatly reduce the overall smoothness of the sentence. Our method uses the actor-critic to modify the token of each position of the original example from left to right, uses the discriminator to restrict the semantics of the confrontation sample, and the language model and the translation model are utilized to evaluate its fluency

Table 2. Experiment results for Mo-Zh and Ug-Zh MT attacks. We list BLEU for perturbed test sets generated by each adversarial example generation method. An ideal adversarial example shouldachieve low $BLEU$ and high HE.

Model	Mo→Zh		Zh→Mo		Ug→Zh		Zh→Ug	
	BLEU↓	HE↑	BLEU↓	HE↑	BLEU↓	HE↑	BLEU↓	HE↑
RNNSearch	26.51	–	22.25	–	28.24	–	23.18	–
GeneticAttack	20.27	1.98	18.56	2.34	22.12	2.06	19.67	2.43
HotFlip	19.15	2.60	18.03	2.91	20.76	2.47	18.32	2.98
Ours	22.14	3.36	20.75	3.73	23.47	3.26	21.48	3.84
Transformer	30.44	–	25.57	–	32.67	–	26.32	–
GeneticAttack	24.02	2.16	22.13	2.91	26.43	2.21	22.18	2.94
HotFlip	23.17	2.67	21.21	3.14	24.67	2.74	21.35	3.13
Ours	26.87	3.45	23.12	3.62	26.98	3.47	23.69	3.88

and the overall deterioration of the model. Therefore, our model stably generate adversarial examples without significant change in semantics and any handcrafted semantic constraints by the same training setting among different models and tasks, achieving stably model degradation and high HE.

4.5 Adversarial Training Results and Analysis

Due to $Agent$ can effectively generate adversary examples that retain semantic information, we can directly use these samples to tune the original translation model. Given the original training data, Transformer models of different methods are used to generate equal number of adversarial examples, which are then paired with the initial target sentences.

Table 3. Fine-tuning with adversarial examples.

Model	Mo→Zh		Zh→Mo		Ug→Zh		Zh→Ug	
	BLEU↑	Promote	BLEU↑	Promote	BLEU↑	Promote	BLEU↑	Promote
Transformer	30.44	–	25.57	–	32.67	–	26.32	–
GeneticAttack	31.19	0.75	26.13	0.56	33.46	0.79	26.93	0.61
HotFlip	32.07	1.63	26.41	0.84	34.39	1.72	27.24	0.92
Ours	33.29	2.85	27.33	1.76	35.41	2.79	28.11	1.79

We directly train the model by mixing the augmented sentence pairs with the original sentence pairs. As shown in Table 3, utilizing the adversarial examples generated by GeneticAttack and HotFlip to adversarial training enable improve the performance of the model, but the effect is unapparent. The reason is they aren't guaranteeing semantic consistency and sentence fluency, and has a poor effect attack on the model. Our method can not only guarantee the semantics, but also have strong aggression against the translation model. The results of fine-tuning using the adversarial examples show that the robustness of the model can be significantly improved.

4.6 Ablation Study

Table 4 shows the results of ablation study. Line 1 represent only use discriminator (D) rewards to guide $Agent$ optimization. It is clear that NMT reward r_d plays a critical role since removing it impairs model performance (line 2 and line 3). The language model reward is also shown to be benefit for improving performance (line 4) but seem to have relatively smaller contributions than r_d.

Table 4. The results of ablation study, "○" means utilize this module and "×" means not.

ID	D	NMT	LM	MO-ZH
1	○	×	×	28.35
2	○	○	×	33.02
3	○	×	○	28.94
4	○	○	○	33.29

5 Conclusion

This paper adopts a novel approach to generate adversarial examples for low-resource machine translation tasks. It can expose the defects of the translation model without manual error features, and ensure the semantic consistency with the original examples. The experimental results on CCMT2019 Mongolian-Chinese and CWMT2017 Uighur-Chinese show that this method achieves stable model degradation on different attacked models. Furthermore, we use adversarial examples to fine-tune the model, and the performance is significantly improved after adversarial training.

References

1. Alzantot, M., Sharma, Y., Elgohary, A., Ho, B., Srivastava, M.B., Chang, K.: Generating natural language adversarial examples. In: Riloff, E., Chiang, D., Hockenmaier, J., Tsujii, J. (eds.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018, pp. 2890–2896. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/d18-1316
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015, Conference Track Proceedings (2015). http://arxiv.org/abs/1409.0473
3. Belinkov, Y., Bisk, Y.: Synthetic and natural noise both break neural machine translation. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018, Conference Track Proceedings. OpenReview.net (2018). https://openreview.net/forum?id=BJ8vJebC-
4. Cheng, M., Yi, J., Chen, P., Zhang, H., Hsieh, C.: Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. 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, 7–12 February 2020, pp. 3601–3608. AAAI Press (2020). https://ojs.aaai.org/index.php/AAAI/article/view/5767

5. Ebrahimi, J., Lowd, D., Dou, D.: On adversarial examples for character-level neural machine translation. In: Bender, E.M., Derczynski, L., Isabelle, P. (eds.) Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, 20–26 August 2018, pp. 653–663. Association for Computational Linguistics (2018). https://aclanthology.org/C18-1055/
6. Ebrahimi, J., Rao, A., Lowd, D., Dou, D.: Hotflip: white-box adversarial examples for text classification. In: Gurevych, I., Miyao, Y. (eds.) Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, 15–20 July 2018, Volume 2: Short Papers, pp. 31–36. Association for Computational Linguistics (2018). https://aclanthology.org/P18-2006/
7. Karpukhin, V., Levy, O., Eisenstein, J., Ghazvininejad, M.: Training on synthetic noise improves robustness to natural noise in machine translation. In: Xu, W., Ritter, A., Baldwin, T., Rahimi, A. (eds.) Proceedings of the 5th Workshop on Noisy User-generated Text, W-NUT@EMNLP 2019, Hong Kong, China, 4 November 2019, pp. 42–47. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/D19-5506
8. Keneshloo, Y., Shi, T., Ramakrishnan, N., Reddy, C.K.: Deep reinforcement learning for sequence to sequence models. CoRR abs/1805.09461 (2018). http://arxiv.org/abs/1805.09461
9. Konda, V.R., Tsitsiklis, J.N.: Actor-critic algorithms. In: Solla, S.A., Leen, T.K., Müller, K. (eds.) Advances in Neural Information Processing Systems, NIPS Conference, Denver, Colorado, USA, 29 November–4 December 1999, vol. 12, pp. 1008–1014. The MIT Press (1999). http://papers.nips.cc/paper/1786-actor-critic-algorithms
10. Li, Z., Sun, M.: Punctuation as implicit annotations for Chinese word segmentation. Comput. Linguist. **35**(4), 505–512 (2009). https://doi.org/10.1162/coli.2009.35.4.35403
11. Michel, P., Li, X., Neubig, G., Pino, J.M.: On evaluation of adversarial perturbations for sequence-to-sequence models. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019, Volume 1 (Long and Short Papers), pp. 3103–3114. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/n19-1314
12. Post, M.: A call for clarity in reporting BLEU scores. In: Bojar, O., et al. (eds.) Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, 31 October–1 November 2018, pp. 186–191. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/w18-6319
13. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 2016, Berlin, Germany, Volume 1: Long Papers, pp. 7–12 (2016). https://doi.org/10.18653/v1/p16-1162
14. Szegedy, C., et al.: Intriguing properties of neural networks. In: Bengio, Y., LeCun, Y. (eds.) 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, 14–16 April 2014, Conference Track Proceedings (2014). http://arxiv.org/abs/1312.6199
15. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4–9 December 2017, Long Beach, CA, USA, pp. 5998–6008 (2017). http://papers.nips.cc/paper/7181-attention-is-all-you-need
16. Wu, L., Tian, F., Qin, T., Lai, J., Liu, T.: A study of reinforcement learning for neural machine translation. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October– 4 November 2018, pp. 3612–3621 (2018). https://www.aclweb.org/anthology/D18-1397/

17. Yang, Z., Chen, W., Wang, F., Xu, B.: Improving neural machine translation with conditional sequence generative adversarial nets. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, 1–6 June 2018, Volume 1 (Long Papers), pp. 1346–1355 (2018). https://www.aclweb.org/anthology/N18-1122/
18. Zhao, Y., Zhang, J., He, Z., Zong, C., Wu, H.: Addressing troublesome words in neural machine translation. In: Riloff, E., Chiang, D., Hockenmaier, J., Tsujii, J. (eds.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018, pp. 391–400. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/d18-1036
19. Zhao, Z., Dua, D., Singh, S.: Generating natural adversarial examples. CoRR abs/1710.11342 (2017). http://arxiv.org/abs/1710.11342
20. Zou, W., Huang, S., Xie, J., Dai, X., Chen, J.: A reinforced generation of adversarial examples for neural machine translation. In: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R. (eds.) Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, 5–10 July 2020, pp. 3486–3497. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.acl-main.319