Generating Fluent Adversarial Examples for Natural Languages

Huangzhao Zhang[1*] **Hao Zhou**[2] **Ning Miao**[2] **Lei Li**[2]

[1]Institute of Computer Science and Technology, Peking University, China
[2]ByteDance AI Lab, Beijing, China
zhang_hz@pku.edu.cn
{miaoning,zhouhao.nlp,lileilab}@bytedance.com

Abstract

Efficiently building an adversarial attacker for natural language processing (NLP) tasks is a real challenge. Firstly, as the sentence space is discrete, it is difficult to make small perturbations along the direction of gradients. Secondly, the fluency of the generated examples cannot be guaranteed. In this paper, we propose MHA, which addresses both problems by performing Metropolis-Hastings sampling, whose proposal is designed with the guidance of gradients. Experiments on IMDB and SNLI show that our proposed MHA outperforms the baseline model on attacking capability. Adversarial training with MHA also leads to better robustness and performance.

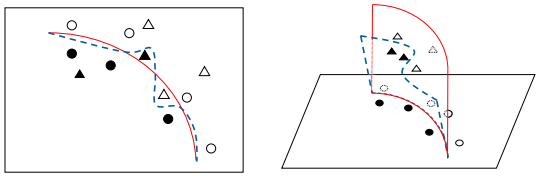
1 Introduction

Adversarial learning has been a popular topic in deep learning. Attackers generate adversarial examples by perturbing the samples and use these examples to fool deep neural networks (DNNs). From the perspective of defense, adversarial examples are mixed into the training set to improve performance and robustness of the victim models.

However, building an attacker for NLP models (such as a text classifier) is extremely challenging. Firstly, it is difficult to perform gradient-based perturbations since the sentence space is discrete. However, gradient information is critical – it leads to the steepest direction to more effective examples. Secondly, adversarial examples are usually not fluent sentences. Unfluent examples are less effective in attacking, as victim models can easily learn to recognize them. Meanwhile, adversarial training on them usually does not perform well (see Figure 1 for detailed analysis).

Current methods cannot properly handle the two problems. Ebrahimi et al. (2018) (HotFlip) propose to perturb a sentence by flipping one of the characters, and use the gradient of each perturbation to guide sample selection. But simple character flipping often leads to meaningless words (*eg.* "moo**d**" to "moo**P**"). Genetic attack (Alzantot et al., 2018) is a population-based word replacing attacker, which aims to generate fluent sentences by filtering out the unreasonable sentences with a language model. But the fluency of examples generated by genetic attack is still not satisfactory and it is inefficient as the gradient is discarded.

To address the aforementioned problems, we propose the Metropolis-Hastings attack (MHA) algorithm in this short paper. MHA is an adversarial example generator based on Metropolis-Hastings (M-H) sampling (Metropolis et al., 1953; HASTINGS, 1970; Chib and Greenberg, 1995). M-H sampling is a classical MCMC sampling approach, which has been applied to many NLP



(a) Adversarial training with fluent adversarial examples (b) Adversarial training with unfluent adversarial examples

Figure 1: Effect of adversarial training on (a) fluent and (b) unfluent adversarial examples. ○ and ● represent positive and negative samples in the training set, while △ and ▲ are the corresponding adversarial examples. Solid and Dotted lines represent decision boundaries before and after adversarial training, respectively. As unfluent adversarial examples are not in the manifold of real sentences, the victim model only needs to adjust its decision boundary out of the sentence manifold to fit them. As a result, fluent adversarial examples may be more effective than unfluent ones.

[*] Work done while Huangzhao Zhang was a research intern in ByteDance AI Lab, Beijing, China.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 5564–5569
Florence, Italy, July 28 - August 2, 2019. ©2019 Association for Computational Linguistics

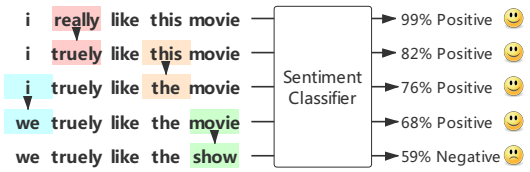


Figure 2: A simple example of adversarial attack on a sentimental classifier by performing word replacement.

tasks, such as natural language generation (Kumagai et al., 2016), constrained sentence generation (Miao et al., 2018), guided open story generation (Harrison et al., 2017), *etc.* We propose two variants of MHA, namely a black-box MHA (*b*-MHA) and a white-box MHA (*w*-MHA). Specifically, in contrast to previous language generation models using M-H, *b*-MHA's stationary distribution is equipped with a language model term and an adversarial attacking term. The two terms make the generation of adversarial examples fluent and effective. *w*-MHA even incorporates adversarial gradients into proposal distributions to speed up the generation of adversarial examples.

Our contributions include that we propose an efficient approach for generating fluent adversarial examples. Experimental results on IMDB (Maas et al., 2011) and SNLI (Bowman et al., 2015) show that, compared with the state-of-the-art genetic model, MHA generates examples faster, achieving higher success rates with much fewer invocations. Meanwhile, adversarial samples from MHA are not only more fluent but also more effective to improve the adversarial robustness and classification accuracy after adversarial training.

2 Preliminary

Generally, adversarial attacks aim to mislead the neural models by feeding adversarial examples with perturbations, while adversarial training aims to improve the models by utilizing the perturbed examples. Adversarial examples fool the model into producing erroneous outputs, such as irrelevant answers in QA systems or wrong labels in text classifiers (Figure 2). Training with such examples may enhance performance and robustness.

Definitions of the terms in this paper are as follow. The **victim models** are word-level classifiers, which take in tokenized sentences and output their labels. The **attackers** generate sentences by perturbing the original ones, in order to mislead the victim model into making mistakes. **Adversarial attacks** include two categories: (a) **black-box attack** only allows the attackers to have access to model outputs, while (b) **white-box attack** allows full access to the victim model, including model outputs, gradients and (hyper-)parameters. For **adversarial training**, the same victim model is trained from scratch on an updated training set with adversarial examples included.

3 Proposed Method: MHA

In this section, we first introduce M-H sampling briefly, and then describe how to apply M-H sampling efficiently to generate adversarial examples for natural language.

3.1 Metropolis-Hastings Sampling

The M-H algorithm is a classical Markov chain Monte Carlo sampling approach. Given the stationary distribution ($\pi(x)$) and transition proposal, M-H is able to generate desirable examples from $\pi(x)$. Specifically, at each iteration, a proposal to jump from x to x' is made based on the proposal distribution ($g(x'|x)$). The proposal is accepted with a probability given by the acceptance rate:

$$\alpha(x'|x) = \min\{1, \frac{\pi(x')g(x|x')}{\pi(x)g(x'|x)}\} \quad (1)$$

Once accepted, the algorithm jumps to x'. Otherwise, it stays at x.

3.2 Black-Box Attack

In black-box attack (*b*-MHA), we expect the examples to meet three requirements: (a) to read fluently; (b) to be able to fool the classifier; (c) to invoke the classifier for as few times as possible.

Stationary distribution. To meet these requirements, the stationary distribution is designed as:

$$\pi(x|\tilde{y}) \propto LM(x) \cdot C(\tilde{y}|x) \quad (2)$$

where $LM(x)$ is the probability of the sentence (x) given by a pre-trained language model (LM) and $C(\tilde{y}|x)$ is the probability of an erroneous label ($\tilde{y}$) given by the victim model. $LM(x)$ guarantees fluency, while $C(\tilde{y}|x)$ is the attack target.

Transition proposal. There are three word-level transition operations – replacement, insertion and deletion. **Traversal indexing** is applied to select words on which operations are performed. Suppose MHA selects the i-th word (w_i) on the t-th proposal, then on the $(t+1)$-th proposal, the selected word (w^*) is:

$$w^* = \begin{cases} w_{i+1}, & if\ i \neq n \\ w_1, & otherwise \end{cases}$$

The transition function for **replacement** is as Equation 3, where w_m is the selected word to be replaced, and $\mathcal{Q}$ is a pre-selected candidate set, which will be explained later. The **insertion** operation ($T_i^B(x'|x)$) consists of two steps – inserting a random word into the position and then performing replacement upon it. The **deletion** operation is rather simple. $T_d^B(x'|x) = 1$ if $x' = x_{-m}$, where x_{-m} is the sentence after deleting the m-th word (w_m), or $T_d^B(x'|x) = 0$ otherwise.

$$T_r^B(x'|x) = \mathcal{I}\{w^c \in \mathcal{Q}\} \cdot \frac{\pi(w_1, \cdots, w_{m-1}, w^c, w_{m+1}, \cdots, w_n|\tilde{y})}{\sum_{w \in \mathcal{Q}} \pi(w_1, \cdots, w_{m-1}, w, w_{m+1}, \cdots, w_n|\tilde{y})} \quad (3)$$

The proposal distribution is a weighted sum of the transition functions:

$$g(x'|x) = p_r T_r^B(x'|x) + p_i T_i^B(x'|x) + p_d T_d^B(x'|x)$$

where p_r, p_i and p_d are pre-defined probabilities of the operations.

Pre-selection. The **pre-selector** generates a candidate set for $T_r^B(x'|x)$ and $T_i^B(x'|x)$. It chooses the most possible words according to the score ($S^B(w|x)$) to form the candidate word set $\mathcal{Q}$. $S^B(w|x)$ is formulated as:

$$S^B(w|x) = LM(w|x_{[1:m-1]}) \cdot LM_b(w|x_{[m+1:n]})$$

where $x_{[1:m-1]} = \{w_1, \cdots, w_{m-1}\}$ is the prefix of the sentence, $x_{[m+1:n]}$ is the suffix of the sentence, and LM_b is a pre-trained backward language model. Without pre-selection, $\mathcal{Q}$ will include all words in the vocabulary, and the classifier will be invoked repeatedly to compute the denominator of Equation 3, which is inefficient.

3.3 White-Box Attack

The only difference between white-box attack (*w*-MHA) and *b*-MHA lies in the pre-selector.

Pre-selection. In *w*-MHA, the gradient is introduced into the pre-selection score ($S^W(w|x)$). $S^W(w|x)$ is formulated as:

$$S^W(w|x) = S^B(w|x) \cdot S(\frac{\partial \tilde{\mathcal{L}}}{\partial e_m}, e_m - e)$$

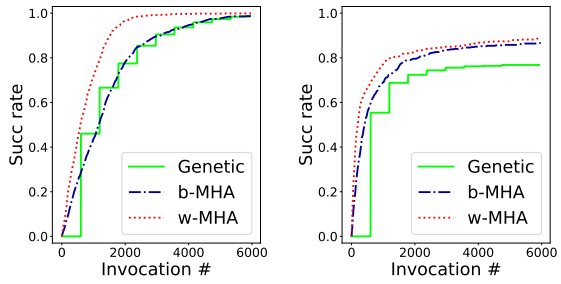
where S is the cosine similarity function, $\tilde{\mathcal{L}} = \mathcal{L}(\tilde{y}|x, C)$ is the loss function on the target label, e_m and e are the embeddings of the current word (w_m) and the substitute (w). The gradient ($\frac{\partial \tilde{\mathcal{L}}}{\partial e_m}$) leads to the steepest direction, and $e_m - e$ is the actual changing direction if e_m is replaced by e. The cosine similarity term ($S(\frac{\partial \tilde{\mathcal{L}}}{\partial w_m}, \Delta w)$) guides the samples to jumping along the direction of the gradient, which raises $C(\tilde{y}|x)$ and $\alpha(x'|x)$, and eventually makes *w*-MHA more efficient.

Note that insertion and deletion are excluded in *w*-MHA, because it is difficult to compute their gradients. Take the insertion operation for instance. One may apply a similar technique in *b*-MHA, by first inserting a random word forming intermediate sentence $x^* = \{w_1, \cdots, w_m, w^*, w_{m+1}, \cdots, w_n\}$ and then performing replacement operation upon x^*. Computing $\frac{\partial \mathcal{L}(\tilde{y}|x^*, C)}{\partial w^*}$ is easy, but it is not the actual gradient. Computing of the actual gradient ($\frac{\partial \mathcal{L}(\tilde{y}|x, C)}{\partial w^*}$) is hard, since the change from x to x^* is discrete and non-differential.

4 Experiments

Datasets. Following previous works, we validate the performance of proposed MHA on IMDB and SNLI datesets. The IMDB dataset includes 25,000 training samples and 25,000 test samples of movie reviews, tagged with sentimental labels (positive or negative). The SNLI dataset contains 55,000 training samples, 10,000 validation samples and 10,000 test samples. Each sample contains a premise, a hypothesis and an inference label (entailment, contradiction or neutral). We adopt a single layer bi-LSTM and the BiDAF model (Seo et al., 2016) (which employs bidirectional attention flow mechanism to capture relationships between sentence pairs) as the victim models on IMDB and SNLI, respectively.

Baseline Genetic Attacker. We take the state-of-the-art genetic attack model (Alzantot et al., 2018) as our baseline, which uses a gradient-free population-based algorithm. Intuitively, it maintains a population of sentences, and perturbs them by word-level replacement according to the embedding distances without considering the victim model. Then, the intermediate sentences are



(a) IMDB (b) SNLI

Figure 3: Invocation-success curves of the attacks.

Task	Approach	Succ(%)	Invok#	PPL	α(%)
IMDB	Genetic	98.7	1427.5	421.1	–
	b-MHA	98.7	1372.1	385.6	17.9
	w-MHA	**99.9**	**748.2**	**375.3**	34.4
SNLI	Genetic	76.8	971.9	834.1	–
	b-MHA	86.6	681.7	358.8	9.7
	w-MHA	**88.6**	**525.0**	**332.4**	13.3

Table 1: Adversarial attack results on IMDB and SNLI. The acceptance rates (α) of M-H sampling are in a reasonable range.

Case 1
Premise: *three men are sitting on a beach dressed in orange with refuse carts in front of them.* **Hypothesis:** *empty trash cans are sitting on a beach.* **Prediction:** ⟨Contradiction⟩
Genetic: ***empties*** *trash cans are sitting on a beach.* **Prediction:** ⟨Entailment⟩
b*-MHA:** ***the *trash cans are sitting* ***in*** *a beach.* **Prediction:** ⟨Entailment⟩
w*-MHA:** ***the *trash cans are sitting on a beach.* **Prediction:** ⟨Entailment⟩
Case 2
Premise: *a man is holding a microphone in front of his mouth.* **Hypothesis:** *a male has a device near his mouth.* **Prediction:** ⟨Entailment⟩
Genetic: *a* ***masculine*** *has a device near his mouth.* **Prediction:** ⟨Neutral⟩
b*-MHA:** *a man has a device near his* ***car*.* **Prediction:** ⟨Neutral⟩
w*-MHA:** *a man has a device near his* ***home*.* **Prediction:** ⟨Neutral⟩

Table 2: Adversarial examples generated on SNLI.

filtered by the victim classifier and a language model, which leads to the next generation.

Hyper-parameters. As in the work of Miao et al. (2018), MHA is limited to make proposals for at most 200 times, and we pre-select 30 candidates at each iteration. Constraints are included in MHA to forbid any operations on sentimental words (eg. "great") or negation words (eg. "not") in IMDB experiments with SentiWordNet (Esuli and Sebastiani, 2006; Baccianella et al., 2010). All LSTMs in the victim models have 128 units. The victim model reaches 83.1% and 81.1% test accuracies on IMDB and SNLI, which are acceptable results. More detailed hyper-parameter settings are included in the appendix.

4.1 Adversarial Attack

To validate the attacking efficiency, we randomly sample 1000 and 500 correctly classified examples from the IMDB and SNLI test sets, respectively. Attacking success rate and invocation times (of the victim model) are employed for testing efficiency. As shown in Figure 3, curves of our proposed MHA are above the genetic baseline, which indicates the efficiency of MHA. By incorporating gradient information in proposal distribution, *w*-MHA even performs better than *b*-MHA, as the curves rise fast. Note that the ladder-shaped curves of the genetic approach is caused by its population-based nature.

We list detailed results in Table 1. Success rates are obtained by invoking the victim model for at most 6,000 times. As shown, the gaps of success rates between the models are not very large, because all models can give pretty high success rate. However, as expected, our proposed MHA provides lower perplexity (PPL) [1], which means the examples generated by MHA are more likely to appear in the corpus of the evaluation language model. As the corpus is large enough and the language model for evaluation is strong enough, it indicates the examples generated by MHA are more likely to appear in natural language space. It eventually leads to better fluency.

Human evaluations are also performed. From the examples that all three approaches successfully attacked, we sample 40 examples on IMDB. Three volunteers are asked to label the generated examples. Examples with false labels from the victim classifier and with true labels from the volunteers are regarded as actual adversarial examples. The adversarial example ratios of the genetic approach, *b*-MHA and *w*-MHA are 98.3%, 99.2% and 96.7%, respectively, indicating that almost all generated examples are adversarial examples. Volunteers are also asked to rank the generated examples by fluency on SNLI ("1" indicating the most

[1] We use the open released GPT2 (Radford et al.) model for PPL evaluation.

Model	Attack succ (%)		
	Genetic	*b*-MHA	*w*-MHA
Victim model	98.7	98.7	99.9
+ Genetic adv training	93.8	99.6	100.0
+ *b*-MHA adv training	93.0	95.7	99.7
+ *w*-MHA adv training	92.4	97.5	100.0

Table 3: Robustness test results on IMDB.

Model	Acc (%)		
	Train # = 10K	30K	100K
Victim model	58.9	65.8	73.0
+ Genetic adv training	58.8	66.1	**73.6**
+ *w*-MHA adv training	**60.0**	**66.9**	**73.5**

Table 4: Accuracy results after adversarial training.

fluent while "3" indicating the least fluent). 20 examples are sampled in the same manners mentioned above. The mean values of ranking of the genetic approach, *b*-MHA and *w*-MHA are 1.93, 1.80 and 2.03, indicating that *b*-MHA generates the most fluent samples. Samples generated by *w*-MHA are less fluent than the genetic approach. It is possibly because the gradient introduced into the pre-selector could influence the fluency of the sentence, from the perspective of human beings.

Adversarial examples from different models on SNLI are shown in Table 2. The genetic approach may replace verbs with different tense or may replace nouns with different plurality, which can cause grammatical mistakes (*eg.* Case 1), while MHA employs the language model to formulate the stationary distribution in order to avoid such grammatical mistakes. MHA does not have constraints that word replacement should have similar meanings. MHA may replace entities or verbs with some irrelevant words, leading to meaning changes of the original sentence (*eg.* Case 2). More cases are included in the appendix.

4.2 Adversarial Training

In order to validate whether adversarial training is helpful for improving the adversarial robustness or classification accuracy of the victim model, a new model is trained from scratch after mixing the generated examples into the training set.

To test the adversarial robustness, we attack the new models with all methods on IMDB. As shown in Table 3, the new model after genetic adversarial training can not defend MHA. On the contrary, adversarial training with *b*-MHA or *w*-MHA decreases the success rate of genetic attack. It shows that the adversarial examples from MHA could be more effective than unfluent ones from genetic attack, as assumed in Figure 1.

To test whether the new models could achieve accuracy gains after adversarial training, experiments are carried out on different sizes of training data, which are subsets of SNLI's training set. The number of adversarial examples is fixed to 250 during experiment. The classification accuracies of the new models after the adversarial training by different approaches are listed in Table 4. Adversarial training with *w*-MHA significantly improves the accuracy on all three settings (with p-values less than 0.02). *w*-MHA outperforms the genetic baseline with 10K and 30K training data, and gets comparable improvements with 100K training data. Less training data leads to larger accuracy gains, and MHA performs significantly better than the genetic approach on smaller training set.

5 Future Works

Current MHA returns the examples when the label is changed, which may lead to incomplete sentences, which are unfluent from the perspective of human beings. Constraints such as forcing the model to generate ⟨EOS⟩ at the end of the sentence before returning may address this issue.

Also, entity and verb replacements without limitations have negative influence on adversarial example generations for tasks such as NLI. Limitations of similarity during word operations are essential to settle the problem. Constraints such as limitation of the embedding distance may help out. Another solution is introducing the inverse of embedding distance in the pre-selection source.

6 Conclusion

In this paper, we propose MHA, which generates adversarial examples for natural language by adopting the MH sampling approach. Experimental results show that our proposed MHA could generate adversarial examples faster than the genetic baseline. Obtained adversarial examples from MHA are more fluent and may be more effective for adversarial training.

7 Acknowledgments

We would like to thank Lili Mou for his constructive suggestions. We also would like to thank the anonymous reviewers for their insightful comments.

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 the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In *Lrec*, volume 10, pages 2200–2204.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Siddhartha Chib and Edward Greenberg. 1995. Understanding the metropolis-hastings algorithm. *The american statistician*, 49(4):327–335.

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

Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *LREC*, volume 6, pages 417–422. Citeseer.

Brent Harrison, Christopher Purdy, and Mark O Riedl. 2017. Toward automated story generation with markov chain monte carlo methods and deep neural networks. In *Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference*.

WK HASTINGS. 1970. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109.

Kaori Kumagai, Ichiro Kobayashi, Daichi Mochihashi, Hideki Asoh, Tomoaki Nakamura, and Takayuki Nagai. 2016. Human-like natural language generation using monte carlo tree search. In *Proceedings of the INLG 2016 Workshop on Computational Creativity in Natural Language Generation*, pages 11–18.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. 1953. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2018. Cgmh: Constrained sentence generation by metropolis-hastings sampling. *arXiv preprint arXiv:1811.10996*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *ICLR'17; arXiv preprint arXiv:1611.01603*.