Universal Adversarial Attack on Natural Language Model

이상헌

성균관대학교 전자전기컴퓨터공학과 정보 및 지능시스템 연구실

1. Abstract

최근의 많은 연구에 의해 딥러닝 모델은 adversarial attack 에 취약하다는 것이 입증되었다. Adversarial attack 은 딥러닝 모델의 입력에 추가함으로써 출력을 다르게 할 수 있는 noise 를 생성하는 기법으로, image 도메인에서 널리 연구되었다. Adversarial attack 은 Image 도메인뿐만 아니라 text 도메인 모델에 adversarial attack 이 적용될 수 있고, 여러 연구들은 text 분류 모델에 대한 adversarial attack 이 가능하다는 것을 보였다. 본 연구는 주어진 NLP 모델에 대해 단 하나의 perturbation 문장을 생성함으로써 모델을 속이는 universal adversarial attack 기법을 제안한다. Universal adversarial attack 에 의해 생성되는 문장은 입력 문장에 추가되고, 이로 인해 문장의 분류 결과는 기존과 다르게 나타난다. IMDB review dataset 을 이용하여 학습된 분류 모델에 대한 실험을 통해, 제안 기법은 baseline 기법에 비해 약간 좋은 성능을 나타내고, 보다 효율적으로 attack 을 수행할 수 있다는 것이 검증되었다.

2. Objective

Adversarial attack 은 모델의 입력에 인간이 구별할 수 없는 perturbation 을 섞음으로써 모델의 결과를 바꾸는 기법으로, image 도메인 모델에 대해 처음으로 연구되었다. 모든 adversarial attack 기법은 다음과 같은 공통된 개념을 가진다: attack 에 의해 생성되는 adversarial image 의 결과가 원본 image 의 결과와 다르게 하는, 최소한의 크기의 perturbation 을 생성한다. 대표적인 adversarial attack 기법은 Goodfellow et al. [1] 이 제안한 gradient-based 기법인 FGSM (Fast Gradient Sign Method) 이다. 만일 머신 러닝 모델이 시스템 보안에 있어 중요한 부분을 수행하는 경우, adversarial attack 은 전반적인 시스템의 동작에 있어서 치명적인 위협이 될 수 있다.

최근에 text 도메인 모델에 대한 adversarial attack 연구가 진행되었다. Liang et al. [2] 은 character-level CNN 모델에 대해 adversarial text 를 생성하는 gradient-based attack 기법을 제안하였고, text 도메인 모델에 adversarial attack 의 적용이 가능함을 처음으로 보였다. Hossein et al. [3] 은 문장에 의도적으로

구두점을 삽입함으로써 분류 모델을 속이는 기법을 제안하였다. Samanta et al. [4] 은 문장 내의 특정 단어를 동의어 혹은 typo 가 섞인 단어로 치환함으로써 adversarial text 를 생성하는 기법을 제안하였다. 하지만 기존의 text 도메인 모델을 대상으로 한 adversarial attack 기법에 관한 연구들은 모두, 각 입력 문장마다 다른 perturbation 혹은 수정할 부분을 계산하여 적용하기 때문에 높은 computational cost 를 필요로 한다는 공통점을 갖는다. 또한 일부 기법들은 적용되는 language 또는 모델이 수행하는 task 의 특성을 기반으로 생성되기 때문에, 도메인에 대한 지식이 필요하다는 단점을 갖는다.

본 연구에서는 특정 분류 모델의 모든 입력 문장에 대해 동일하게 적용할 수 있는 단 하나의 perturbation 문장을 생성하고 이를 통해 분류 모델을 속이는 universal adversarial attack 기법을 제안한다. 제안 기법은 모든 입력 문장에 대해 adversarial perturbation 혹은 수정할 부분을 각각 계산해야 하는 기존 attack 기법들에 비해 필요 computational cost 가 작기 때문에, 더욱 효율적으로 attack 을 수행할 수 있다. 또한 적용되는 입력 문장의 언어 혹은 모델의 task 에 대한 지식을 반영하지 않기 때문에, 제안 기법은 사전 지식 없이 수행할 수 있다는 장점을 갖는다.

3. Proposed Method

Image 도메인에서의 universal adversarial attack 에 관한 연구는 Moosavi-Dezfooli et al. [5] 에 의해 이미 수행되었다. 본 연구에서 제안하는 기법은 이를 base 로 하지만, 모델의 task 도메인이 image 가 아닌 text 라는 것이 다른 점이며, text 의 특성을 고려하여 기법을 구성하였다.

제안 기법은 그림 1 과 같이 진행된다. 문장을 분류하는 모델과 입력 문장들에 대해, 입력 문장의 끝에 추가함으로써 분류 결과를 달리할 수 있는 길이 n 의 문장을 설정한다. 추가되는 문장인 adversarial text 의 각 단어의 word embedding 값을 학습 가능한 weight 로 설정 및 random initialize 하고, adversary 조건을 만족하도록 weight 를 최적화함으로써 adversarial text 를 생성한다.

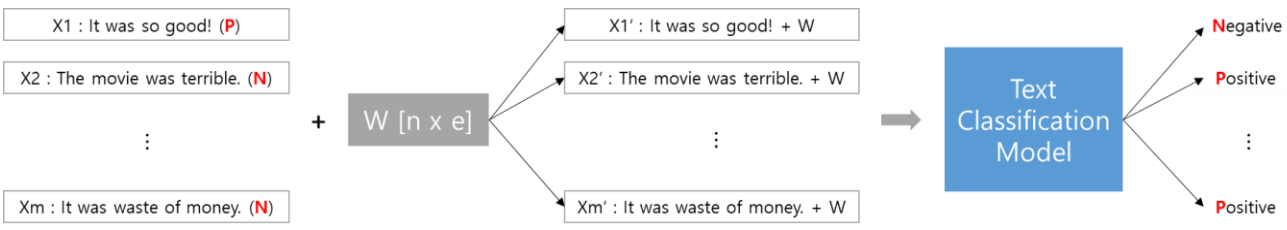

그림 1. Universal adversarial attack 기법. W는 길이 n인 adversarial text의 각 단어에 대한 word embedding 값이며, 학습을 통해 최적화된다.

제안 기법의 수행 단계는 그림 2 와 같다. 먼저, 주어진 문장에 대해 분류를 수행하는 모델을 original training dataset 을 이용하여 학습한다. 실험에서 사용된 모델은 입력 문장의 긍정 및 부정을 판단하는 task 인 sentimental analysis 를 수행하는 모델로, embedding layer, 2-layer LSTM cell, 그리고 fully-connected layer 로 구성되어 있다. Embedding layer 의 word embedding 차원 e 는 300, LSTM cell 의 hidden node 수는 128, fully-connected layer 의 hidden node 수는 256 으로 설정하였다.

이후, 입력 문장이 embedding layer 를 통과한 embedding 값의 뒤에 random initialize 된 길이 n 의 adversarial text 의 word embedding 값들을 concatenate 하고, 이에 따라 모델의 출력이 바뀌도록 하는 adversarial text embedding 값을 찾는다. 이 과정에서 이전에 미리 학습된 문장 분류 모델의 weight 를 고정하여, 오로지 adversarial text 의 embedding 값만 학습에 의해 최적화되도록 한다. 실험에서는 adversarial text 의 길이 n 을 20 으로 설정하였다.

마지막으로, 생성된 adversarial text 의 embedding 값에 embedding layer 의 lookup table 을 곱하고 각 행의 가장 높은 column index 를 뽑아 냄으로써 word embedding 값 각각에 해당하는 단어를 추출한다. 이는 adversarial text 의 embedding 값인 W 와 lookup table 사이 cosine similarity 를 구하여 가장 가까운 단어를 찾는 과정이다. 이렇게 추출된 문장을 original test dataset 의 각 문장 뒤에 concatenate 하여, adversarial example 을 생성한다.

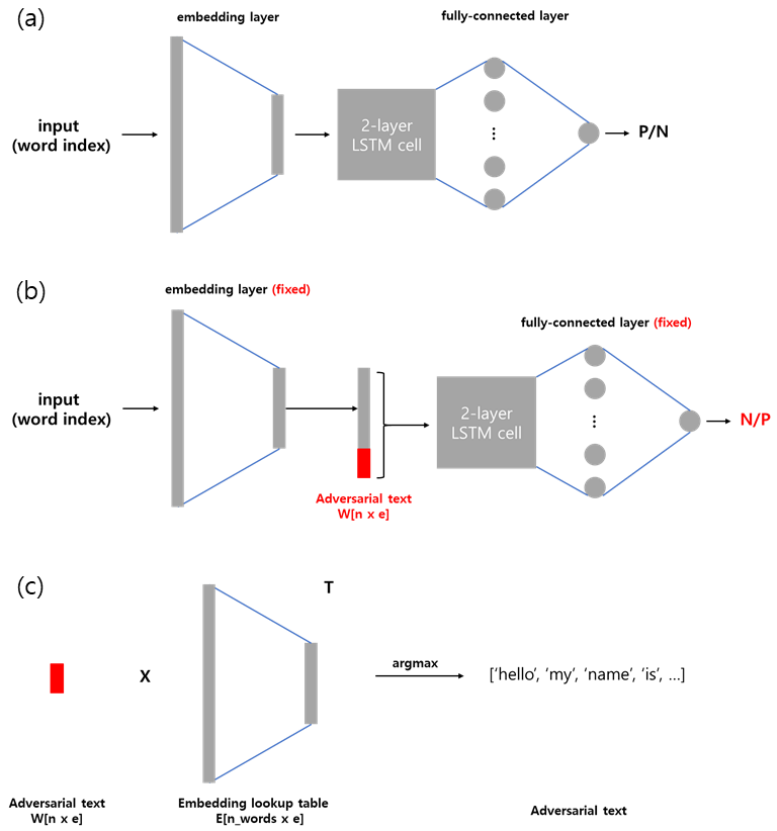

그림 2. 제안 기법의 수행 단계. (a) Original dataset을 이용하여 text classification 모델 학습 단계. (b) Adversarial text의 word embedding 값 최적화 단계. (c) Adversarial text의 각 단어 추출 단계.

4. Evaluation

제안 기법의 성능을 검증하기 위해 사용된 dataset 은 sentimental analysis 를 위한 IMDB 영화 리뷰 dataset 으로 [6], 25,000 개 review 의 training dataset 과 25,000 개 review 의 test dataset 으로 구성되어 있다. Sentimental analysis 수행을 위해 사용된 모델은 앞서 언급한 2-layer LSTM 모델이며, 22,500 개 데이터가 학습을 위해, 2,500 개 데이터가 학습 과정에서의 검증을 위해 사용되었다. 모델 학습을 위한 hyperparameter 로 학습 epoch 는 20, learning rate 는 0.001, batch size 는 250 으로 설정하였으며, 학습에는 Adam optimizer 를 사용하였다.

Adversarial text 의 word embedding 값을 최적화하기 위해 사용된 dataset 은 2-layer LSTM 을 학습하기 위해 사용된 dataset 과 유사하지만, 각 데이터의 출력은 original dataset 의 출력과 반대로 설정하였다. Adversarial text 의 embedding 값의 최적화 과정을 위한 hyperparameter 로, epoch 는 10, learning rate 는 0.0001, batch size 는 250 으로 설정하였으며, 최적화에는 Adam optimizer 를 사용하였다.

기존의 연구들은 perturbation 혹은 수정할 부분을 입력 문장들에 따라 각각 따로 계산하는 기법들이기 때문에 본 연구와는 다소 다른 의미를 가지며, 제안 기법과 같은 환경에서의 비교 기법은 엄밀하게는 존재하지 않는다. 하지만 대략적인 성능 판단을 위해 gradient-based adversarial attack 기법인 TextFool [7] 을 적용한 분류 모델의 성능을 측정하고, 제안 기법과 성능을 비교하였다.

정성적인 실험 결과는 표 1 과 같다. Adversarial text 를 추가하지 않은 original test dataset 에 대한 모델의 accuracy 는 84.56% 이며, baseline 기법인 TextFool 과 제안 기법으로 생성된 adversarial test dataset 에 대한 모델의 accuracy 는 각각 82.63%, 82.09%로 하락하였다. 모델의 accuracy 하락 정도에 따라 제안 기법은 baseline 기법과 비슷하거나 조금 더 좋은 성능을 갖는다는 것을 알 수 있다. 특히 adversarial example 을 생성하는 데 경과되는 시간을 보면 baseline 기법은 한 adversarial example 당 10-20 분이 경과되는 반면, 제안 기법은 한 모델에 대해 단 하나의 adversarial text 를 생성하고 이를 각 데이터 뒤에 추가하는 방식이기 때문에 전체 test data 25,000 개에 대해 10 분이 경과되었다. 따라서 제안 기법은 비슷한 성능을 나타내는 baseline 기법에 비해 더욱 효율적으로 수행될 수 있다는 것을 알 수 있다.

표 1. 제안 기법과 baseline 기법인 TextFool에 대한 정량적인 실험 결과.

	TextFool	Proposed method
Accu. using original test set	84.56%	84.56%
Accu. using adversarial test set	82.63%	82.09%
Time to generate adv. example	10 - 20 min / example	10 min / model

실험에서 adversarial text 의 word embedding 값은 random initialize 되기 때문에, 반복되는 실험마다 매번 다른 문장을 생성한다. 한 번의 실험을 통해 추출된 adversarial text 는 다음과 같다.

['assumed', 'confession', 'tah', 'mizer', 'nowt', 'lurve', 'hennessy', 'attila', 'authenticity', 'canteens', 'gummi', 'mcbain', 'ubiquetous', 'critiques', 'yojimbo', 'auh', 'headtrip', 'dismissable', 'shameful', 'churl']

정성적인 실험 결과, 제안 기법은 자연스러운 adversarial text 문장을 생성하지 못하였다. 이는 제안 기법이 각 단어의 word embedding 값을 독립적으로 최적화하여 찾기 때문에, 언어의 문법적인 혹은 어휘적인 특성을 고려하지 못하여 발생한 현상이다.

5. Discussion

실험 결과 제안 기법은 baseline 기법에 비해 약간 높은 attack 성공률과 높은 효율성을 보였으나, 고려할 만한 몇 가지 한계점을 가지고 있었다.

먼저, adversarial text 의 각 word embedding 값은 범위가 제한되지 않아, 최적화 결과 그림 3 과 같이 기존 word embedding 값 범위 내에 존재하지 않는 것을 확인할 수 있었다. 따라서 제안 기법의 adversarial text 의 word embedding 값을 최적화하는 과정에서 값의 범위에 대한 제약조건이 필요하다. 학습 과정에서 weight 의 범위를 제한하는 기법이 현재까지 없었기 때문에 이러한 제약조건을 위한 새로운 기법이 연구될 필요가 있다.

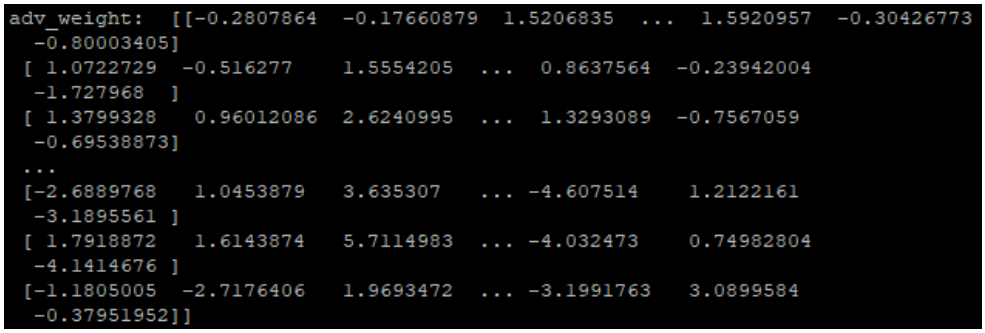

그림 3. 생성된 adversarial text의 각 word embedding값이 적정 범위를 초과하는 현상.

두 번째로, 제안 기법에 의해 생성된 adversarial text 는 자연스럽지 못하다는 한계를 갖는다. 이는 앞서 말한 제약 조건의 부재에 의한 현상으로 볼 수 있지만, 제안 기법이 adversarial text 의 각 단어의 word embedding 값을 각각 독립적으로 찾기 때문에 발생한 현상으로도 볼 수 있다. 이는 그림 4 와 같이 k-

nearest neighbor 기법을 통해, 최적화된 word embedding vector 값과 가장 유사한 단어 k 개를 각각 추출하고, 문장이 구성될 수 있는 단어의 조합들에 대해 beam search 를 적용하여 가장 가까이에 있는 문장(W_{best}) 뿐만 아니라 가장 자연스러운 문장(W_{nature}) 을 생성할 수 있다.

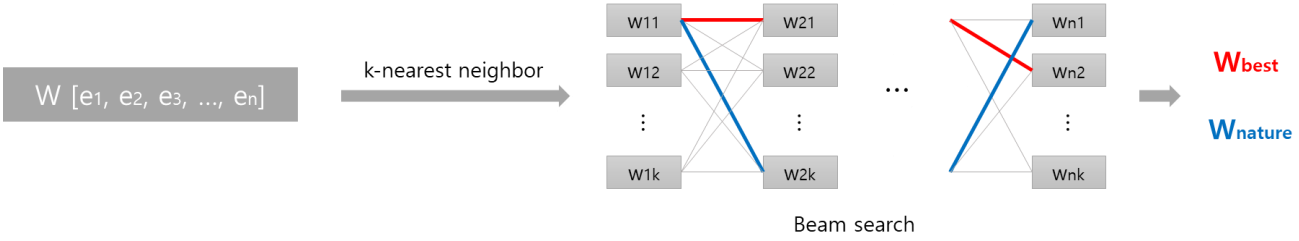

그림 4. K-nearest neighbor 및 beam search를 통해 가장 자연스러운 문장 생성.

6. Conclusion

본 연구는 하나의 모델에 대해 단 하나의 adversarial text 를 생성하는 universal adversarial attack 을 NLP 모델에 적용할 수 있는 기법을 제안하였다. 제안 기법에 의해 생성된 adversarial text 가 각 입력 문장 뒤에 추가됨으로써 adversarial example 이 생성되고, 실험 결과 제안 기법은 성능이 비슷한 baseline 기법에 비해 더욱 효율적으로 attack 을 수행할 수 있었다.

하지만 제안 기법에 의해 생성된 adversarial text 는 자연스럽지 못하다는 단점이 있다. 이를 극복하기 위해 향후에는 adversarial text 내 각 word embedding 값을 제한하는 constraint 를 적용하거나, k-nearest neighbor 및 beam search 를 통해 가장 자연스러운 문장을 생성하는 등의 보완 작업을 진행할 예정이다. 또한 sentimental analysis 뿐만 아니라 다른 task 를 수행하는 NLP 모델들에 대해 제안 기법의 적용 가능성을 검증하는 실험을 진행할 예정이다.

연구를 하면서 본 연구의 제안 기법과는 다른 방식의 universal adversarial attack 기법을 생각해보았다. Text 데이터는 image 데이터처럼 continuous 하지 않고 discrete 하다는 특징이 있기 때문에, 이러한 특징을 이용하여 각 입력 문장의 동일한 위치의 단어에 대한 modification 혹은 동일한 위치에 특정 단어를 insertion 하는 등의 작업을 통해 universal adversarial attack 을 수행할 수 있을 것이라고 생각하였다. 이러한 기법은 어떤 위치에 어떤 작업을 수행할지를 설정하는 것이 가장 중요한데, 이를 일일이 사람이 설정하지 않고 학습을 통해 최적화할 수 있는 알고리즘에 대해 현재 구상 중에 있다. Text 의 discrete 한 특징을 기반으로 한 이러한 universal adversarial attack 기법은 자연스러운 문장을 생성할 수 있고, 모델이 어떠한 부분을 중요하게 고려하는지를 interpretable 하게 설명할 수 있기 때문에 defense 의 관점에서 attack 에 강인한 NLP 모델을 설계할 수 있을 것이라고 생각한다.

7. References

[1] I.J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," in Proc. of International Conference on Learning Representations (ICLR), 2015.

[2] B. Liang, H. Li, M. Su, P. Bian, X. Li, and W. Shi, "Deep Text Classification Can be Fooled," ArXiv preprint arXiv:1704.08006, 2017.

[3] H. Hosseini, S. Kannan, B. Zhang, and R. Poovendran, " Deceiving Google's Perspective API Built for Detecting Toxic Comments," ArXiv e-prints arXiv:1702.08138, 2017.

[4] S. Samanta, and S. Mehta, "Generating Adversarial Text Samples," in Proc. of European Conference on Information Retrieval (ECIR), 2018.

[5] S.M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Forssard, "Universal adversarial perturbations," in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[6] A.L. Maas, R.E. Daly, P.T. Pham, D. Huang, A.Y. Ng, and C. Potts, " Learning Word Vectors for Sentiment Analysis, " in Proc. of Association for Computational Linguistics: Human Language Technologies (ACL-HLT), 2011.

[7] B. Kulynych, Project title, https://github.com/bogdan-kulynych/textfool, 2017.