Generate Adversarial Examples to Attack Semantic Parser

Weiliang Tang*

Department of Computer Science

Chinese University of Hong Kong

2019/8/14

1 Abstract

It is shown that both the image classification [Goodfellow et al., 2014] [Moosavi-Dezfooli et al., 2016] and text classification model [Ebrahimi et al., 2017] [Liang et al., 2017] [Ebrahimi et al., 2018] are vulnerable under adversarial attack, but the study of robustness of semantic parsing model still remains a void to explore. In this summer, we give a new definition of the adversarial examples specified to semantic parser, and manage to develop a new method to significantly and effectively elaborate the successful rate of attacking semantic parser compared to the traditional Fast Gradient Method. We also get insights from the experiment result, and find out the cause of the vulnerability from the successful adversarial examples.

2 Introduction

Semantic parsing is enjoying more and more popularity nowadays and higher and higher accuracy is achieved with the effort of scholars devoting in the NLP study [Hwang et al., 2019]. However, many state-of-art neutral network models referring from image recognition to text classification tasks are reported to be vulnerable under the tiny designed perturbation added to the original input, and the semantic parser should not be an exception. Due to the fact that more scholars are focusing on the elaboration of the accuracy and the adversarial behavior remains unstudied for semantic parser, we determine to focus on attacking the semantic parser this summer. Clearly, different considerations should be taken into account to attack a semantic parser compared to fooling text classification

*Corresponding author: 1155107670@link.cuhk.edu.hk and image recognition model. Firstly, the inputs are usually very short for semantic tasks, so unlike attacking the text classifier, substituting a word with another word could make the input totally a nonsense and human can easily tell the difference between the perturbed input and the original input, and there's no point to observe the behavior given a nonsense as input. Secondly, the inputs are concretely distributed in the word embedding space, so selecting the word with the largest norm of the gradient as the target to perturb as what FGSM method to attack image classifier has shown may not give an optimal result, due to the reason that the word with the largest norm of gradient may be far from any other words else. Our contribution this summer are:

- We defined valid adversarial example for attacking semantic parser.
- We designed a new method (Bert-FGM) to elaborate the successful rate to attack semantic parser.
- We got insights of how the adversarial examples occurs, and discovered that the local word crowded in the embedding space could make the model vulnerable.

3 Definition of Valid Adversarial Example for Semantic Parser

The adversarial generation for text classifier somehow like image classifier focus on the visual distinction. It tries to make a change to the output with as small perturb times as possible in order to fool a human visually. This can be done since inputs for text classifier are usually long enough. However, the input for semantic parser are usually very short(averagely, 13 for Spider Dataset [Yu et al., 2018], 11.1 for ATIS Dataset, 14.4 for Django [Dong and Lapata, 2018]). Any substitution is obvious to distinguish visually. Hence a new standard needs to be set for valid adversarial examples for semantic parser. Since the whole point for a semantic parser is the ability to fetch the semantic meaning behind words, it is reasonable to think if the perturbed input shares the same semantic meaning with the original input, while the output gives an output with different semantic meaning from the original one, then this input should be a valid adversarial example. So to sum up, an adversarial example should:

- Keep the same semantic meaning as the original input.
- Lead to an output that doesn't share the same semantic meaning as the original output

In order to better assess the performance of adversarial generation method and the robustness of the model, we define two index:

- Diff ratio: The ratio of the number of perturbed inputs that leads to a different output and the total number of input examples.

- Valid ratio: The ratio of the number of perturbed inputs with the same semantic meaning as the original input that leads to a output with different semantic meaning and the total number of inputs that lead to a different output.

Notice here that the different output doesn't not mean they have different semantic meaning. For an example, in the NL2SQL task, "SELECT ... FROM ... WHERE A=1 AND B=2" is different from "SELECT ... FROM ... WHERE B=2 AND A=1", but they obviously mean the same. Plus, the semantic identical doesn't mean grammatically correctness. For an example, in NL2SQL task, "How many apples are there" and "How many apples is there" should matter the same to a robust NL2SQL model since the grammar correctness has no effect on the meaning of the input for NL2SQL model.
Lastly, we want to point out that the semantic equilibrium is task-aware. Two inputs may be semantic identical in one task while different in another. For an example, "What is the number of xxx" and "What is the accurate number of xxx" may mean the same for NL2SQL task, but have different meanings for NL2Python task since the accuracy requirement can't be neglected sometimes for code generation. Even if in the same task, the situation can vary. Different structure of schema and tables under the same task NL2SQL may have effect on the identical verification of the two inputs.
The term diff ratio and valid ratio will be used for assessment for our experiments.

4 Fast Gradient Method

The simplest method to try first should be the Fast Gradient Method. The algorithm is as followed:

```
FGM
1 // $grad\_data$ = (input_len × embedding_size)
2 for $i = 0$ to $length[grad\_data] - 1$
3       $word\_grad[i] = \|grad\_data[i]\|$
4 $target\_word = \arg\max(word\_grad)$
5 $perturbed\_word = \arg\min_{w \in embed\_space} \|word[idx] + \epsilon \cdot grad\_data[idx] - w\|$
```

5 Bert-FGM Method

We further proposed a new method to combine the FGM with Bert, which significantly improved the successful rate. The original FGM method ignores two facts:
First, when considering the words to substitute with the original word, it only consider the distance in the word embedding space, which means that it only consider the meaning difference of two words without the semantic environment

where they lie in. So the perturbation will always end up a nonsense. In order to allow the word selection process to look into the semantic environment in a higher granularity, we block the word and use Bert [Devlin et al., 2018] to predict a list of ten words $bert\_list = Bert(original\_sentence, blocked\_word, 10)$ that's most suitable to fill in, along with the corresponding probability for each word $bert\_prob$, and then to iterate this list to find out the word that maximize the measurement $c \cdot bert\_prob + cos\_simi(word + \epsilon \cdot grad\_data, w)$. The measurement method takes two factors into considerations: (i). $bert\_prob$, which is derived by Bert by looking into the surrounding semantic environment, representing the semantic rational of a word; (ii). the cosine similarity between the target word and the substituted word, which describes the degree of following the direction of gradient by picking up the substituted word. c here is a constant to balance the two factor mentioned above, and a trade off will be clearly see in the experiment result.
Second, due to the discrete input space, only focusing on the word with the largest norm of gradient may not bring about an optimal result, since the word with the largest norm of gradient may be in an area wherein words are sparsely distributed and every other word is relatively far from this word, so picking up a word with a slightly smaller norm of gradient with more words being closer to it may be a better choice. In order to tackle this issue, we take the top 3 words with largest norm of gradient into attack lists, and pick up the word that maximize the measurement $c \cdot bert\_prob + cos\_simi(word + \epsilon \cdot grad\_data, w)$.

BERT-FGM
```
1  for $i = 0$ to 3
2      // $grad\_data = (\text{input\_len} \times \text{embedding\_size})$
3      for $i = 0$ to $length[grad\_data] - 1$
4          $word\_grad[i] = \|grad\_data[i]\|$
5      $target\_word\_list = n\_\arg\max(word\_grad)$
6      for $i = 0$ to $length[idx\_list] - 1$
7          $target\_word = target\_word\_list[i]$
8          $bert\_list = Bert(sen, target\_word, 10)$
9          $word\_list = \underset{w \in bert\_list}{\arg\max}\ c \cdot bert\_prob[w] + cos\_simi(\epsilon \cdot grad\_data[idx], w - target\_word)$
10     $perturbed\_word = \underset{w \in word\_list}{\arg\max}\ c \cdot bert\_prob[w] + cos\_simi(\epsilon \cdot grad\_data[idx], w - target\_word)$
11     $word \Rightarrow perturbed\_word$
```

Another key detail here is that we used $cos\_simi(word + \epsilon \cdot grad\_data, w)$ instead of $\|word[idx] + \epsilon \cdot grad\_data[idx] - w\|$ to measure the degree of the perturbed word following the gradient. The intuition here is that the words in the word list given by Bert is already close enough to the original word, the angle plays a more important role to affect the deviation of the loss as shown in the graph below:

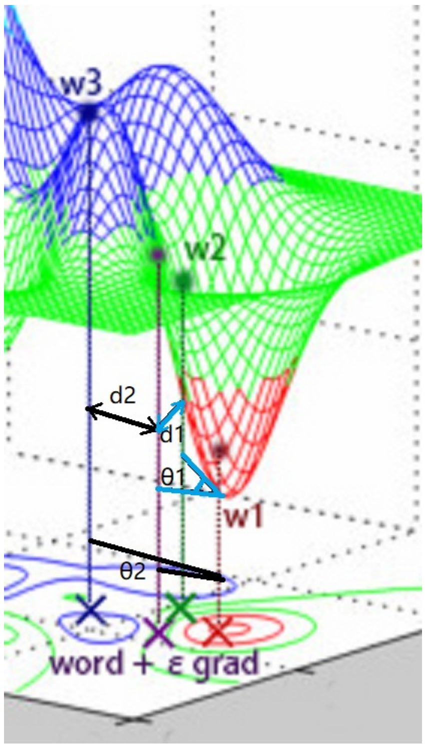


Figure 1: $d_2 > d_1$, however, $loss(w_3) > loss(w_2)$, using the cosine similarity can better reflect the collinearation degree of the gradient and the

6 Experiment

We selected NL2SQL as the semantic parsing task, and utilized the Coarse2Fine model [Dong and Lapata, 2018] as the target model to attack. The model chooses WikiSQL [Zhong et al., 2017] as training set, along with 80654 pieces of data across 24241 different tables. We implemented the model and ran it and got an accuracy of 67.46%.
We ran the model through testing data, and calculated the diff ratio and valid ratio mentioned in the Definition of Valid Adversarial Example for Semantic Parser session. The result with different ϵ value is shown as followed:

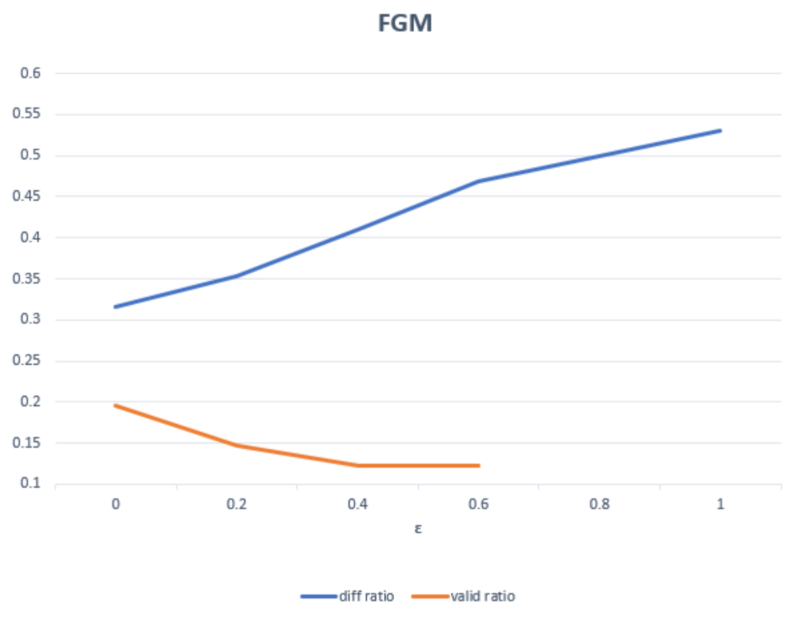


Figure 2: Experiment Result of FGM

For the Bert-FGM method, we set $\epsilon = 3$, and vary c from 0 to 2000. Here's what we've got:

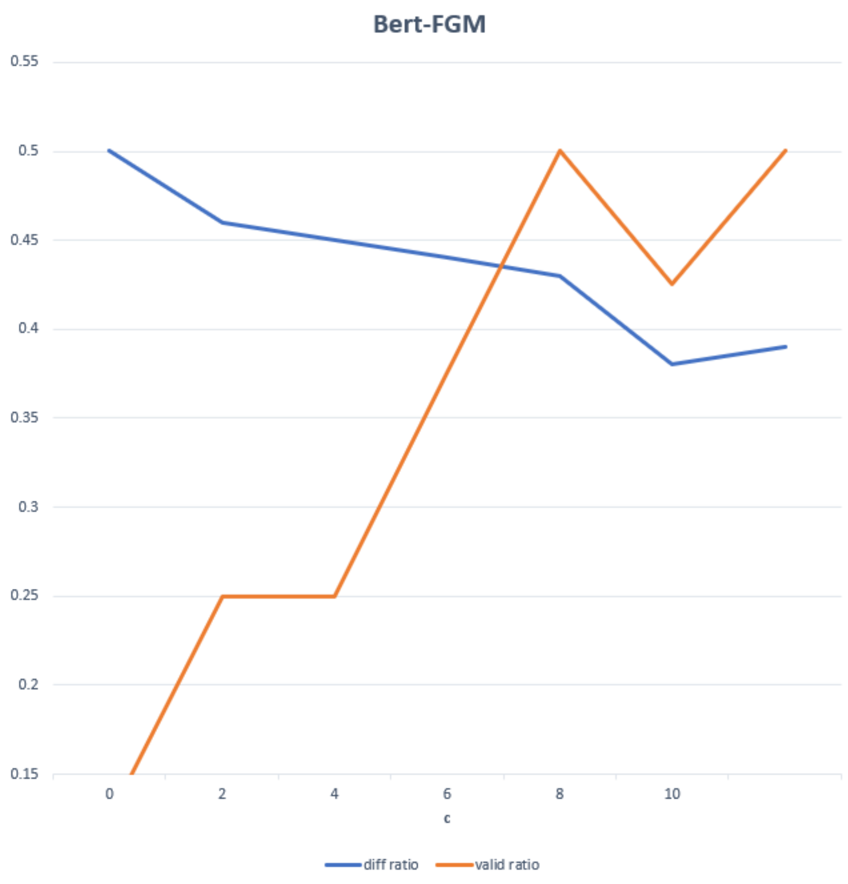


Figure 3: Experiment Result of Bert-FGM

7 Discussion

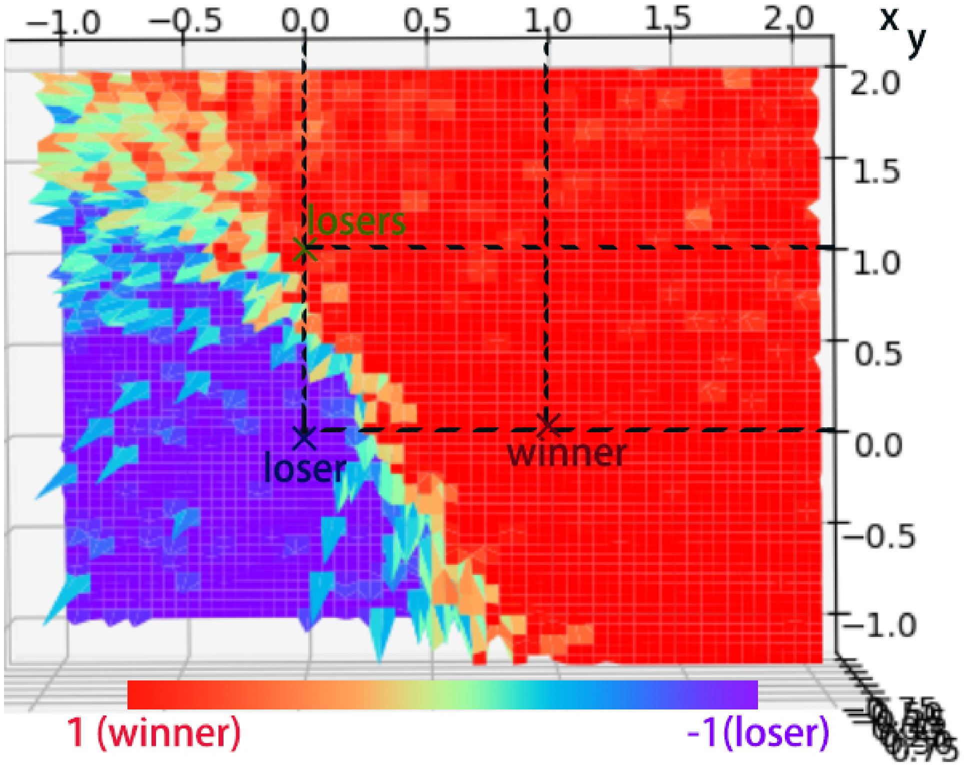
In this session, we are going to display some interesting experiment results, announce our discoveries and explain reasons behind the occurrence of adversarial examples.

Firstly, the result of FGM method shows that when ϵ is relatively small,the larger the ϵ it is, the higher diff ratio it would be due to two reasons:(i). The larger ϵ means the larger distance traveling from the original word's position following the routine where the loss grows the fastest. (ii). The ϵ is too large and reaches a region wherein the words have nothing to do with the previous word, and this has no difference with substituting the original word with a random word. As $\epsilon > 0.6$ and grows larger and larger, more and more words will change the result the way (ii). When ϵ is getting larger, the perturbed word's meaning is going further from the original word, so the valid ratio drops.

Secondly, there's a trade off between the diff ratio and valid ratio w.r.t. the value c in the Bert-FGM experiment. The higher the value of c, the more contribution is *bert_prob*, the picking process focus more on the semantic fluency rather than the following the direction of the gradient, and as a result, more valid examples is derived but diff ratio drops. On the contrary, the smaller the c is , the selection focuses more on following the direction given by the gradient, and it will more strictly go along the direction that makes the loss grow the fastest, so more examples that lead to a different output is generated, but the valid ratio drops.
Thirdly, Bert-FGM elaborate the diff ratio and valid ratio, which fits our intuition and show our new method is efficient for attack. In addition, our attacking method results in generating more pattern-various examples, due to the reason that since the FGM method only focuses on the distance of different word, one word can only be perturbed into only another fixed word under no circumstances, like it will turn all the "How many" into "How well", so the successful perturbation only has limited patterns, like they turn all the "total" into "amount". Bert-FGSM breaks this limitation and find words according to semantic environment.
Some interesting examples are shown below:
Example1:
original input: what is the name of the loser when the winner was new england patriots , . . . ?
original SQL: SELECT loser WHERE winner = new england patriots . . .
perturbed input: what is the name of the losers when the winner was new england patriots , . . . ?
perturbed output: SELECT winner WHERE winner = new england patriots . . .
Example2:
original input: How many types of organization . . .
original SQL: SELECT MAX types WHERE. . .
perturbed input:How many kinds of organization . . .
perturbed SQL: SELECT MAX organization WHERE. . .
It seems that it is odd that changing the word from 'loser' to 'losers' will result in "SELECT winner", but after all, this three words are very close to each other in the word embedding space. So an intuition come into our minds that whether the word 'winner' affect the word 'losers' due to the close distance. And we set up another experiment and find out the close distance between the input word can easily cause regional under-fitting problem , which eventually leads to the existence of adversarial example. (More representative examples is will be attacked in the github)

8 Under-fitting Due to Local Word Crowded

We firstly find the embedding vector of the word "losers, "loser" and "winner", noted as $\vec{v}("losers")$, $\vec{v}("loser")$, $\vec{v}("winner")$ respectively, and define the input space $C = \{(x,y)|x,y \in [-1,2]\}$. We construct a two-dimensional space $P = \{\vec{v}|\vec{v} = \vec{v}(''loser'') + x\cdot(\vec{v}(''winner'') - \vec{v}(''loser'')) + y\cdot(\vec{v}(''losers'') - \vec{v}(''loser''))\}$. Notice that this is a plane in the high-dimensional word embedding space, where the embedding vector of the words "loser", "losers" and "winner" lie in. When $x = y = 0$, $\vec{v} = \vec{v}(''loser'')$. When $x = 0, y = 1$, $\vec{v} = \vec{v}(''losers'')$. When $x = 1, y = 0$, $\vec{v} = \vec{v}(''winner'')$. We then vary (x,y) to measure the output $z = \frac{p('loser')-p('winner')}{p('loser')+p('winner')}$. z is a value $\in [-1,1]$, and the larger the z is, the more belief the model is put on "SELECT winner" rather than "SELECT loser", vice versa. We color the value, making it redder as it's closer to 1, and bluer while it's closer to -1, and take a snapshot through the negative-z axis, and the the result:



It's clearly to be seen from the graph that input "loser" and input "winner" will give "SELECT loser" and "SELECT winner" respectively, which means that the model is well-trained and fits the point "loser" and "winner" very well. However, the word "losers", although is close to the word "loser" and share the same meaning to "loser", is in the red area, due to the reason that "winner" is also close to "loser" and affect the result. In order to more clearly express the idea, we adopt 2 idealize graph to help explanation.

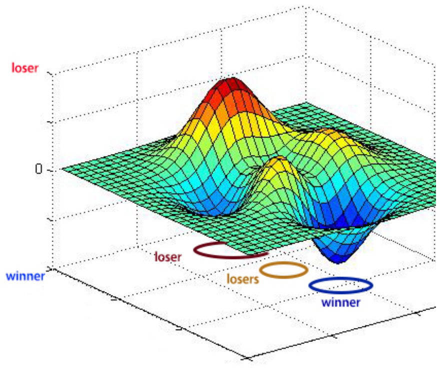


Figure 4: What is a robust model should suppose to be like

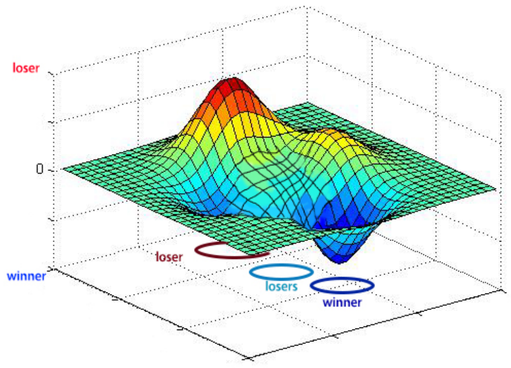


Figure 5: What it is like actually

The headers of the table in the NL2SQL task are usually very close to each other, and may be vulnerable under adversarial attack(we will carry out more experiments in the future to show this). This is the new adversarial feature that NL2SQL model holds.

9 Feature Work

In the future we will carry out more experiment conducting with more semantic parser, to test and improve our attacking method. Furthermore, the valid diff so far is counted manually which is of low efficiency, we desire to design a better method to count for valid ratio.

10 Conclusion

This this summer, we dig into the study of robustness of semantic parser. We define new standard for valid adversarial example according to the input feature that a semantic parser holds. We design a new attacking method which elaborate not only the diff ratio, but also the valid ratio as well. We have insights of the experiment result and get inspired to find out the under-fitting problem in semantic parser.

11 Reference

References

[Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[Dong and Lapata, 2018] Dong, L. and Lapata, M. (2018). Coarse-to-fine decoding for neural semantic parsing. *arXiv preprint arXiv:1805.04793*.

[Ebrahimi et al., 2018] Ebrahimi, J., Lowd, D., and Dou, D. (2018). On adversarial examples for character-level neural machine translation. *arXiv preprint arXiv:1806.09030*.

[Ebrahimi et al., 2017] Ebrahimi, J., Rao, A., Lowd, D., and Dou, D. (2017). Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*.

[Goodfellow et al., 2014] Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

[Hwang et al., 2019] Hwang, W., Yim, J., Park, S., and Seo, M. (2019). Achieving 90% accuracy in wikisql.

[Liang et al., 2017] Liang, B., Li, H., Su, M., Bian, P., Li, X., and Shi, W. (2017). Deep text classification can be fooled. *arXiv preprint arXiv:1704.08006*.

[Moosavi-Dezfooli et al., 2016] Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. (2016). Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582.

[Yu et al., 2018] Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., Ma, J., Li, I., Yao, Q., Roman, S., et al. (2018). Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*.

[Zhong et al., 2017] Zhong, V., Xiong, C., and Socher, R. (2017). Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.