Data Augmentation: Can BERT Do The Work For You?

Stanford CS224N Default Project

Yubo Han
Department of Computer Science
Stanford University
hidyhan@stanford.edu

Abstract

Data augmentation has been proved effective in analyzing a neural model's robustness and improving it by re-training with augmented data. Because text data's discrete feature space, most data augmentation techniques require querying multiple systems for language knowledge and meticulous augmentation rule design by researchers. This paper aims to explore the effectiveness of an automatic, black-box data augmentation method using language models, `bert context rewriter`, and to compare it with another augmentation algorithm, `token reorderer`, which uses Universal Sentence Encoder's semantic knowledge. Given a baseline question answering model, we employ DistilBERT masked language model (mlm) to rewrite masked context data and evaluate whether re-training with the augmented data can improve the robustness of the baseline model. This augmentation relies on the existing language knowledge learnt by DistilBERT mlm and does not use additional hand-crafted rules. We also explore how different configurations, including masked token percentage and additional mlm fine-tuning, affect our method's effectiveness. Preliminary experiments show that both our methods obtain improved performance on out-of-domain dev set over the baseline and reduce the performance gaps between in-domain and out-of-domain datasets. However, `token reorderer`'s performance is consistently better than `bert context rewriter`'s in both out-of-domain evaluation (+2.9 F1/+2.9 EM versus +1.9 F1/+1.6 EM) and reducing in-domain out-of-domain gaps (-5.3 F1/-4.8 EM versus -1.7 F1/-2.5 EM) and therefore is more effective in improving the baseline model's robustness.

1 Introduction

In recent years, machine learning models can achieve outstanding performance on held-out test sets in a variety of NLP tasks. BERT is the first model that beats human performance in tasks such as question answering [1]. However, there have also been increasing concerns about whether the powerful models that achieve high BLEU and F1 scores indeed understand language or instead just learn superficial patterns. In particular, research has shown that many neural models' performances significantly decrease with minor input perturbation that is not perceivable to humans [2]. Because standard metrics are lenient on models that rely on superficial cues, adversaries with data augmentation have been proposed as an additional method to evaluate model robustness [3].

In addition to being an auxiliary evaluation method, data augmentation introduces more variety to the original dataset, and re-training with the augmented data can in some cases improve the original model's performance on existing data and resilience to other adversarial attacks. In the case of unbalanced train and test datasets, data augmentation may effectively disrupt domain-specific features and improve model's transfer-ability.

Data augmentation in NLP is challenging because language token space is discrete: gradient and generator based models are oftentimes not applicable [4]. Many existing data augmentation techniques

rely on linguistics knowledge from either existing or newly built language knowledge bases. However, it is hard to separate the contribution of each system and evaluate their impact independently, and to find effective manual augmentation rules researchers need to do extensive experiments. This paper aims to explore whether it is feasible to utilize language models to implicitly retrieve this knowledge, instead of explicitly querying systems and writing augmentation rules. In particular, our data augmentation techniques augment data for question answering tasks and are evaluated on a baseline DistilBERT question answering (QA) model. We implement `bert context rewriter` by employing DistilBERT masked language model to rewrite question answering context data. We also experiment different configurations and analyze how they affect the question answering baseline model's performance on in-domain and out-of-domain datasets. As a comparison, we also implement another simple data augmentation method, `token reorderer`, that changes the structure of questions based on semantic knowledge from Universal Sentence Encoder [5]. Our results show, while both techniques can improve baseline model's performance on the out-of-domain dataset, `token reorderer` consistently performs better than `bert context rewriter` in both test dataset evaluation and reducing the evaluation gaps between in-domain and out-of-domain datasets. We conclude that before we can fully understand what BERT knows about language, data augmentation with explicit linguistics rules is still more effective than black-box augmentation with BERT.

2 Related Work

Text data augmentation techniques have been extensively explored in the context of adversarial attack and adversarial training for NLP neural models. Due to text data's distinct discrete space, it is challenging to perturb the original data without altering its meaning too much or breaking its coherence [6]. Researchers have found success in utilizing manual linguistics rules to mutate text at character, word and short phrase levels. Such methods include introducing typos [7], synonym replacement [8] and inserting/removing phrases with no semantic meanings [2]. Recently, powerful language models have been employed to generate augmented data with less rule design by human [9]. However, to construct meaningful attacks, most techniques listed above perturb tokens in a sentence sequentially and greedily until the attacked model produces erroneous outputs. While they are applicable to some NLP tasks, such as sentence classification, where text data is relatively short, the techniques are often not parallelizable and therefore not scalable in tasks such as question answering where contexts can be several hundred words long.

Several data augmentation algorithms have been introduced to specifically target question answering. `AddSent` [3], the first algorithm to attack reading comprehension models, concatenates a distracting sentence, either random or carefully crafted, at the end of a context. `AddSentDiverse` [10] further expands on `AddSent` by introducing more thorough attack rules and proves that re-training on the augmented data significantly improves the attacked model's robustness. These two methods, just like most methods listed above, both query extensive language knowledge from existing systems and use the knowledge to generate attacks in well-designed pipelines. For example, to construct a distraction sentence, `AddSent` would consult WordNet for antonym replacement, query GloVe for named entity replacement and call Stanford's CoreNLP library for POS tagging information. While the depending systems provide ample knowledge in English, they may not be all as powerful in other lower-resource languages. Because the depending systems are used jointly, it is hard to measure whether the augmentation techniques will be equally effective if one of the systems is unavailable or weaker. Further, as can be seen from the complexity of such a pipeline, researchers need to spend quite a lot of time doing experiments to craft manual rules. Therefore, it is useful to develop a generation method that has fewer dependencies and can effectively augment data with minimal human guidance, which motivates the design of `bert context rewriter`.

3 Approach

The baseline model is a DistilBERT QA model [11] trained on in-domain datasets. We come up with and implement two data augmentation techniques: `token reorderer` and `bert context rewriter`. The first method is suitable for short sentences, whereas the second works for long paragraphs. In the context of question answering, we use the first method to augment questions and the second to augment contexts.

3.1 Token Reorderer

For each token in the question, we determine its importance by removing it and comparing the similarity of the new sentence with the original one, which is measured with Universal Sentence Encoder [5]. The more dissimilar the mutated sentence is, the more important we deem the removed token is. In the end, we concatenate the reordered question sentence with the original one. The pseudo code is shown in Algorithm I.

Algorithm 1: Token Reorderer

$q\_enc \leftarrow encode(original\_question)$;
$sim\_to\_token \leftarrow \{\}$;
for *(i in q)* **do**
  $mutated \leftarrow q.remove(i)$;
  $sim\_to\_token[cosine(encode(mutated), q\_enc)] = i$;
end
$sim\_to\_token.sort\_by\_key(ascending = True)$;
$reordered \leftarrow sim\_to\_token.values()$;
$return\ original\_question + reordered$;

3.2 Bert Context Rewriter

The pseudo code for masking logic is shown in Algorithm II. On a high level, we rewrite contexts by masking tokens and replacing them with DistilBERT's most confident predictions corresponding to the masks, which is a fairly standard usage of language models. There are a few design choices we'd like to highlight. First, our masking scheme implementation is inspired by SpanBERT [12]. Without SpanBERT's extensive pre-training, however, DistilBERT mlm would produce gibberish for long mask streams. As a result, we change the max mask length to 3 and change the parameter of geometric distribution, from which the mask length is sampled, to 0.8. Note that long continuous masks can still form because multiple mask spans may merge. Analysis over an empirical run shows that while around 75% mask spans are of length 1, the longest mask span can be up to 55 tokens. Originally other than masking we also implement keeping and replacing with random words logic just like BERT and SpanBERT, but we realize in practice they have little effect, so we remove the latter operations in all experiments.

In order to preserve question answer labels, we don't mask words directly, since after tokenization the mutated words may be longer or shorter than the original ones, which will make the labels inaccurate. Instead, we directly mask on the token level after tokenization but always round up the mask spans to whole words. We never mask tokens in answer spans. To prevent exceedingly long run time, we stop masking when 2/3 of all tokens are either masked or belong to the answer span.

Our masking logic is both used for fine-tuning DistilBERT mlm and rewriting contexts. More details in the experiment section.

4 Experiments

4.1 Data

Datasets and splits are listed in Table 1 and they are standard for this default project. Training is conducted on in-domain train datasets and evaluation is conducted on in-domain and out-of-domain dev datasets respectively. In some cases, we also report metrics on the test datasets.

4.2 Evaluation method

We use EM and F1 to evaluate the performance of our QA model. In cases where in-domain metrics drop and out-of-domain metrics improve, we believe the data augmentation methods have removed domain-specific features the model learns and successfully increased the system's robustness.

Algorithm 2: Bert Context Rewriter

$nr\_masked \leftarrow 0$;
$mutated \leftarrow \{\}$;
while *(nr_masked < budget)* **do**
  $sample\_len \leftarrow min(Geometric(0.8), CAP)$;
  $start\_idx \leftarrow Uniform(context\_start, context\_end)$;
  $start\_idx, end\_idx \leftarrow$
   $round\_to\_whole\_word(start\_idx, sampled\_len, offset\_mapping)$;
  if *range(start_idx, end_idx) not in mutated or answer_span* **then**
    $original\_tensor[start\_idx : end\_idx] \leftarrow tokenizer.mask\_token\_id$;
    $nr\_masked \leftarrow nr\_masked + end\_idx - start\_idx$;
    $mutated.appends(range(start\_idx, end\_idx))$
  end
end

Table 1: Experiment Datasets

Dataset	Train	Dev	Test
in-domain datasets			
SQuAD [13]	50000	10507	-
NewsQA [14]	50000	4212	-
Natural Questions [15]	50000	12836	-
oo-domain datasets			
DuoRC [16]	127	126	1248
RACE [17]	127	128	419
RelationExtraction [18]	127	128	2693

4.3 Experimental details

For `token reorderer`, we augment both training and test datasets, since the algorithm only mutates question data and does not rely on any knowledge about answer span. We study two configurations for `token reorderer`: whether the original question is completely replaced by or concatenated with the reordered question. For `bert context rewriter`, we only augment training data, because we don't have labels for test data and may unwittingly alter tokens in the answer span, making it impossible for the model to predict correctly. There are several different configurations that we study: 1) whether the DistilBERT mlm is further fine-tuned on out-of-domain dataset 2) what the percentage of masked tokens is. We do not tune hyperparameters, including learning rate, batch size and training time otherwise, in order to have an apple-to-apple comparison with the baseline.

4.4 Results

Experiment results can be found in Table 2 and we analyze how different configurations affect augmentation's effectiveness. While we recognize parameters may work in conjunction in subtle ways, due to time constraint we mostly control one variable at a time and do pair-wise analysis.

4.4.1 Baseline

The baseline model shows a significant gap between in-domain validation and out-of-domain validation metrics, indicating that the model has learnt domain-specific, non-transferable features. Overall, the performance of the baseline model is already quite good.

4.4.2 Token Reorderer

`Token reorderer` #1 replaces the original question with the reordered one, whereas `token reorderer` #2 concatenates the reordered question after the original one. `Token reorderer` #2

Table 2: Experiment Results

Method	F1/EM (oo dev)	F1/EM(in dev)	F1/EM (oo test)	mask ratio	mlm fine tuning
baseline	48.43/33.25	70.08/55.07	59.0/-	-	-
token reorderer 1	47.66/31.94	66.55/50.49	-/-	-	-
* token reorderer 2	51.31/36.13	68.33/52.48	60.58/42.5	-	-
bert context rewriter 1	50.38/34.82	70.29/54.15	59.62/41.08	5%	none
bert context rewriter 2	49.81/34.55	69.49/53.37	-/-	15%	none
bert context rewriter 3	48.12/33.25	67.45/51.09	-/-	15%	60 epochs

achieves the best performance overall across all methods and configurations - it not only beats the baseline in both F1 and EM for out-of-domain evaluation (+2.9 F1/+2.9 EM), but its in-domain validation scores also decrease (-1.7 F1/-2.5 EM). These two signals imply that `token orderer` #2 helps the model drop some features overfitting to the in-domain datasets and learn transferable features. Compared to `token reorderer` #1, `token reorderer` #2 also retains more signals by preserving the original semantics of the question and therefore performs better. `Token reorderer` #1 shows a slight decrease in out-of-domain evaluation (-0.8F1/-1.3EM) but an even larger decrease in in-domain evaluation (-3.5F1/-4.5EM). We suspect that some domain-specific question structures may be disrupted due to token reordering, and consequently the model becomes less domain-specific. However, question re-ordering also breaks its semantics and makes answering harder across both domains.

4.4.3 Bert Context Rewriter

`Bert context rewriter` #1 achieves the best performance on out-of-domain dev set among all `bert context rewriters` with different configurations (+1.9 F1/+1.6 EM). Likely, this rewriter magnifies some signals such as replacing a context word with a synonym that aligns well with a question word. However, it barely decreases the in-domain performance (+0.2 F1/-1.9 EM). While the out-of-domain performance is marginally improved, we don't believe the algorithm has instructed the model to become more robust or domain-independent. Below we also discuss how two parameter choices affect augmentation.

- Masked token percentage. `Bert context rewriter` #1 and #2 differ only in masking percentage: #1 masks 5% tokens whereas #2 masks 15%. Both achieve marginal improvement over the baseline for out-of-domain evaluation, and compared to #1 both in-domain and out-of-domain performances for #2 are a bit worse (by around 0.5 in F1). This is expected, because more tokens are mutated and the datasets are more noisy. However, in general the mutation at 15% surprisingly does not affect model performance much. We conclude that the DistilBERT QA model is quite resilient to token-level replacement noise, especially when it is introduced by another BERT model.

- Mlm fine-tuning. In `bert context rewriter` #3, before rewriting the contexts we fine-tune the mlm on the out-of-domain test dataset for 60 epochs. Comparing to #2, this significantly decreases in-domain evaluation (-2.8 F1/-2.9 EM). We believe rewrites adapt original contexts closer to the distribution of out-of-domain test dataset. Unfortunately, this does not translate to a better performance on out-of-domain dev dataset. Rather than removing domain-dependent features and improving model's robustness, #3's augmentation merely makes in-domain question answering harder by rewriting the context in an unfamiliar language. We also experiment fine-tuning with 10 epochs and 30 epochs. In those cases, in-domain and out-of-domain performances are almost identical with those of the baseline mlm, indicating that our fine-tuning does not change the mlm in any significant way.

5 Analysis

In retrospect, there are many reasons why `token reorderer`, which is based on external semantic knowledge, may outperform `bert context rewriter`. We list a couple below.

- `Token reorderer` introduces new semantic knowledge from Universal Sentence Encoder that DistilBERT QA model likely does not know / focus on. After question tokens are re-ordered by their importance, DistilBERT QA model may focus on answer spans corresponding to the important tokens first, which may more likely to be right. It is unclear whether DistilBERT's model has direct concepts of "token importance" at this point, but at least from evaluation results, introducing such a concept brings improvement. This observation also rationalizes why other data augmentation methods usually consult various systems for language knowledge. Each system likely introduces something new to the evaluated model and helps improve it in a different way. As such, hand-crafted augmentations that use linguistics rules do have their strength.

- DistilBERT mlm and DistilBERT QA model's feature spaces are too similar. We find an example, where context phrase "dyed yellow hair" is rewritten to "colored yellow hair" and the question is "what color did she use to color her hair". At first glance this may help improve the model's answer. In reality, either way the model can find the correct span. The idea is simple - both DistilBERT mlm and QA model know "color" and "dye" are synonyms, so at best the rewrite magnifies the signal but it won't add new knowledge. If there are synonym replacements that DistilBERT question answering does not know and can benefit from, DistilBERT mlm won't know either. This can potentially be changed with additional fine-tuning of DistilBERT mlm, but given our data size, we cannot change the mlm's parameters in significant ways. Alternatively, we can try using BERT mlm to augment data for another model that is not BERT based, or vice versa, to make sure augmentation can indeed bring new information to the system.

- `Token reorderer` breaks question structures which is a more drastic perturbation. By contrast, BERT models are known to be overly stable to semantic perturbation. Adversarial training research has shown that BERT does not change its answer even if the relevant context's meaning is completely reversed by introducing antonyms and negation words [3]. In the original design, we recognize that this will be a flaw in our rewrite results but think it is acceptable - if the QA model is agnostic to semantic changes but still can find the answer span given the surrounding context and structure, it still has some merits. After all, when performing a question answering task, even humans identify answers by surrounding structures and potentially miss details that render their answers wrong. However, just because BERT is agnostic to semantic details, replacement only augmentation is very weak - our experiment shows that even replacing 15% tokens won't affect the model's performance much. One potential improvement is that we can introduce operations other than token replacement, such as insertion and deletion, which may result in more drastic structural changes and improve model's robustness.

6 Conclusion

Our project shows that `token reorderer`, a semantic-based augmentation method using Universal Sentence Encoder, and `bert context rewriter`, an automatic augmentation method using DistilBERT mlm, both can improve the baseline DistilBERT QA model's performance on out-of-domain dataset. The former, however, is more effective in generalizing the model by removing its domain-specific features. Therefore, our answer to the question in the title – can BERT do the work for you – is: not quite yet. Careful design of augmentation rules based on semantic and linguistics knowledge is still important in exposing a neural model's robustness issues and improving them. It may still be possible to utilize BERT mlm to do data augmentation, but before we can understand how to extract any specific linguistics knowledge from the model better, researchers still need to be highly involved in designing how to best combine its strength with other systems. To make this conclusion more complete, we can do further work to verify the limitations we hypothesize in the analysis section. Namely, we can experiment using `bert context rewriter` to augment the data for another QA model that is not based on BERT, and we can introduce more structural mutations, including inserting and removing tokens, to `bert context rewriter`. We hope our work will shed light on the strength and weaknesses of BERT mlm in the context of automatic data augmentation and inspire future researchers to find how to best incorporate BERT mlm into their data augmentation techniques.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[2] Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. Deep text classification can be fooled. *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, Jul 2018.

[3] Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems, 2017.

[4] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.

[5] Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder, 2018.

[6] Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. Adversarial attacks on deep learning models in natural language processing: A survey, 2019.

[7] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers, 2018.

[8] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is bert really robust? a strong baseline for natural language attack on text classification and entailment, 2020.

[9] Siddhant Garg and Goutham Ramakrishnan. Bae: Bert-based adversarial examples for text classification, 2020.

[10] Yicheng Wang and Mohit Bansal. Robust machine comprehension models via adversarial training, 2018.

[11] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[12] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans, 2020.

[13] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, 2016.

[14] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. Newsqa: A machine comprehension dataset, 2017.

[15] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

[16] Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. Duorc: Towards complex language understanding with paraphrased reading comprehension, 2018.

[17] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations, 2017.

[18] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension, 2017.