Enhancing Question Classification with Augmented Data

Peleg Eliyahou
Ben Gurion University
pelegel@post.bgu.ac.il

Stav Dratwa
Ben Gurion University
stavdr@post.bgu.ac.il

July 2023

Abstract

Short text classification is a challenging aspect of Natural Language Processing. Traditional text classification does not consider some traits that appear in numerous real-world applications, such as short text. Since short texts are typically only one to two sentences long, they lack context and therefore pose a challenge for text classification.

Github repository:
`https://github.com/pelegel/DL_Project_Group3`

KEYWORDS

Text Classification, Question Classification, Supervised Learning, Natural Language Processing, Data Augmentation

1 Introduction

Question classification (QC) is a classic problem in Natural Language Processing (NLP). The task is to assign predefined categories to a given question. In order to get better performances in the question classification task, it is usually required to increase the dataset to get better performances, which is expensive and requires more time to develop new labeled records. However, using new approaches that increase the available dataset can improve performance with the limited training data. In this project, we propose an approach that uses data augmentation as a tool to generate additional training instances. Data augmentation involves creating new samples by applying specific transformations or modifications to the existing data.

By leveraging data augmentation, the goal is to increase the diversity and quantity of training instances, potentially resulting in improving the performance of the QC model.

We will implement our proposed approach on the TREC-6 datasets. The goal is to predict the label for any given question from 6 different classes.

2 Related Work

Traditional text classifiers mainly rely on machine learning techniques such as nearest neighbours, naive Bayes, decision tree and support-vector machines [1]. In recent years, with the emergence of deep learning, many NLP tasks, including text classification, are dominated by complex neural models such as BERT.

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained language representation model developed by Google AI Language. It is designed to understand the context and meaning of words in natural language processing (NLP) tasks [2]. BERT has had a significant impact on the field of NLP and has been widely adopted for various tasks, such as text classification, named entity recognition, question answering, and more, thanks to its ability to capture contextual information and produce high-quality representations of text.

2.1 BERT for Text Classification

The BERT-base model consists of an encoder with 12 Transformer blocks, each equipped with 12 self-attention heads and a hidden size of 768. When provided with a sequence containing no more than 512 tokens, BERT generates a representation for the sequence. This sequence typically consists of one or two segments, with the initial token always being [CLS], containing a special classification embedding. Another special token, [SEP], serves to separate segments [3].

For text classification tasks, BERT utilizes the final hidden state, denoted as h, of the first token [CLS] as the representation for the entire sequence. To predict the probability of a label c, a simple softmax classifier is added atop BERT:

$$p(c|h) = \text{softmax}(Wh) \qquad (1)$$

Here, the task-specific parameter matrix W is employed. To optimize the model, we simultaneously fine-tune all the parameters from BERT and W by maximizing the log-probability of the correct label.

3 Methodology

In this project, we want to check whether data augmentation will improve the accuracy in the question classification task. We used BERT model over TREC-6 dataset and used the published code to get the initial accuracy of the model, without further improvements.

Next, we performed data augmentation processes to produce new training records to use in the model. This produces noise in the data that potentially better reflects the real world.

3.1 EDA

EDA is an acronym for Easy Data Augmentation that performs four data augmentation techniques on literal datasets in order to increase their size and create new records that will potentially improve learning performances [3]. It consists of several data augmentation techniques, such as synonym replacement, random insertion and deletion, and random swap. Each of those techniques changes the original training records and creates new training instances out of the available data, that are similar to the original ones, but with a certain added noise.

Those four techniques are used at once, while each training sentence goes through the four augmentation techniques, creating a new training record from the four augmentations.

In order to check whether increasing the data will result in better performance, we performed the EDA data augmentation at four different levels: one, two, four, and eight new training records out of each existing training record.

3.1.1 Synonym replacement

One of the most natural choices in data augmentation is to replace words or phrases with their synonyms [3]. To get the synonyms of each word in the training data, we used WordNet from nltk.corpus python library. WordNet is a lexical database and semantic network for the English language. It is a large-scale lexical resource that provides information about word meanings, relationships between words, and lexical properties. One of the features this library allows is to get the synonyms of a word. Using these synonyms, we randomly choose a word from the original question and replaced it randomly from the synonyms received. This allows to produce of new training records by using words with similar meanings instead of the original wording.

3.1.2 Random Deletion

One simple method for producing new records is by randomly deleting words from the original records. We randomly deleted words from the original training records, augmenting new records that will be used to train the model.

3.1.3 Random Insertion

In this part, we randomly chose a word out of the original training records and added its synonym in a randomly chosen position.

3.1.4 Random Swap

This augmentation is producing new training records by swapping the position of words from the original training question. This reproduces new training records with noise, that reflects a new wording option of the sentence.

3.2 Back Translation

Another data augmentation technique used in NLP tasks is back translation. This technique takes the original sentence, translates it to a different target language, and then translates it back to the original language. The purpose of back translation is to generate synthetic training data for machine learning models. By translating a sentence into a different language and then translating it back, the original sentence can be reconstructed with some variations or noise. This technique helps in augmenting the training data and introducing diversity, which can improve the generalization and performance of the models.

For the back translation, we used "mtranslate" Python package to translate each training record twice: first from the source language (English) to the chosen target language (Hebrew), and then backward. We chose to translate the training records to Hebrew because of the major differences in syntax and grammar between the two languages, potentially creating new training records that are different from the original ones.

4 EXPERIMENTS

In this section we will present the dataset used in our study, then we will describe the different augmentations steps and our procedure. Finally, we will describe the experimentation's measures.

4.1 Dataset

We evaluate our augmentation methods on the six-class version of the TREC dataset [2]. TREC is dataset for question classification consisting of open-domain, fact-based questions divided into broad semantic categories. The dataset contains in total 6 coarse classes: abbreviation, entity, description, human, location, and numeric. It provides short texts, with an average text length of 10.06 tokens. Compared to other document-level datasets, TREC dataset is sentence-level, and there are fewer training examples for it.

We chose to perform our analysis on this certain dataset because in recent times, new chatbots based language models are launched, and they are trained to conversation and questions answering, such as Chat-GPT. This increases the interest and need for good question classification models, and TREC dataset is used for this question classification purpose.

4.2 Procedure

First, we ran the model for the original dataset, without any augmentation. The accuracy for this dataset was 97.2% as stated in the original article [4]. This high accuracy is challenging to significantly improve.

After we got the initial accuracy, we wanted to check whether the different augmentation techniques will result in better performances. To check this assumption, we created new dataset files, each from a different augmentation over the training set.

1. **EDA-1** - Augmented dataset where each of the original training records was augmented with one new additional training record using EDA.

2. **EDA-2** - Augmented dataset where each of the original training records was augmented with two new additional training records using EDA.

3. **EDA-4** - Augmented dataset where each of the original training records was augmented with four new additional training records using EDA.

4. **EDA-8** - Augmented dataset where each of the original training records was augmented with eight new additional training records using EDA.

5. **Back-translation** - Augmented dataset where each of the original training records was augmented with a new record received from back translation to Hebrew. This dataset added only the back-translated records that were different from the original record.

6. **Back-translation + EDA1** - Augmented dataset that combines EDA1 with back transtation. We took the augmented data from EDA1 and then performed back translation over the training instances.

4.3 Evaluation Metric

Accuracy is a metric commonly used in experiments to measure the performance of classification. It is typically calculated as the ratio of correctly classified instances to the total number of instances in the dataset.

5 Results

Dataset	Training set size	Accuracy	Loss
No Augmentation	5452	97.2%	0.146
EDA-1	10904	97%	0.169
EDA-2	16356	97%	0.196
EDA-4	27260	96.6%	0.21
EDA-8	49068	96.1%	0.22
Back Translation	8853	97.7%	0.106
Back Translation +EDA-1	18210	97.4%	0.196

Table 1: Qusetion Classification results

The table presents the results of different data augmentation techniques applied to the question classification task. The dataset is divided into several subsets, and each subset is augmented using specific augmentation methods. The table shows the training set size, accuracy, and loss achieved by the models trained on these augmented datasets. It is important to note that the test set size and records remained consistent at 500 samples for all datasets, allowing a fair comparison of model performance.

Let's explain the results:
No Augmentation: This is the baseline scenario where no data augmentation is applied. The training set size is 5452, and the model achieved an accuracy of 97.2%.

EDA-1, EDA-2, EDA-4, EDA-8: These rows represent the results of applying Easy Data Augmentation (EDA) with different augmentation levels (1, 2, 4, and 8). EDA introduces various textual transformations such as synonym replacement, swap, random insertion, and deletion to create augmented samples.

As we increase the number of augmentations (from EDA-1 to EDA-8), the training set size increases, leading to more diverse training data.

However, there is a trade-off. While EDA-1 and EDA-2 maintain high accuracy (97%), EDA-4 and EDA-8 show a decrease in accuracy (96.6% and 96.1%, respectively). Moreover, the loss increases slightly with higher augmentation levels.

This indicates that EDA might introduce significant noise and make the model less accurate in its predictions, resulting in a slight decline in performance. In addition, the more augmented sentences produced from each original training record, the lower the accuracy received.

Back Translation: This row shows the result of applying Back Translation to Hebrew, which involves translating sentences between languages and back to the original language to create additional samples. Back Translation achieves an accuracy of 97.7% with a low loss of 0.106. It demonstrates that translating and re-translating sentences can be an effective data augmentation method for enhancing model performance, as it improved our initial results.

Back Translation + EDA-1: This row presents the results of combining Back Translation to Hebrew with EDA-1. Here, the training set size is larger than Back Translation alone, and the accuracy is slightly lower (97.4%). The combined use of Back Translation and EDA introduces even more diverse samples, but it also introduces more noise, which might explain the small decrease in accuracy compared to Back Translation alone. Yet, when implying this augmentation technique, we received an improvement in accuracy compared to the original results.

6 Conclusion

We conducted an investigation to enhance question classification performance using augmented data.

Results show that using Back Translation methods yields the highest accuracy and lowest loss. Augmented data plays a crucial role in improving question classification performance because it increases the diversity and variability of the training dataset. By introducing variations in the input data, the model becomes more robust and can generalize better to unseen examples. Back Translation, which involves translating sentences between languages and back to the original language, creates additional training samples with different phrasings, sentence structures, and semantic representations, leading to improved model performance. It would also be interesting to compare back translations to different languages to examine the change in the model performances due to the change in the target language.

However, **EDA (Easy Data Augmentation) with a high number of augmentations (EDA-4 and EDA-8) led to a drop in accuracy and an increase in loss.** While data augmentation is effective in enhancing generalization, excessive augmentations can introduce noise and confuse the model during training.

EDA performs various operations such as synonym replacement, random insertion, and deletion, which can generate unrealistic examples or blur the boundaries between classes.

Consequently, this can negatively impact the model's ability to discriminate between different classes, leading to reduced accuracy and increased loss.

Selecting appropriate data augmentation techniques is crucial for improving question classification models effectively. Future research could focus on exploring more advanced augmentation strategies and fine-tuning hyperparameters to strike a balance between diversity and noise in the augmented data. By understanding the impact of different augmentations, we can develop more reliable and robust question classification models that generalize well to real-world scenarios.

References

[1] Karl, F., & Scherp, A. (2022). Transformers are Short Text Classi-

fiers: A Study of Inductive Short Text Classifiers on Benchmarks and Real-world Datasets. arXiv preprint arXiv:2211.16878.

[2] Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., ... & He, L. (2022). A survey on text classification: From traditional to deep learning. ACM Transactions on Intelligent Systems and Technology (TIST), 13(2), 1-41.

[3] Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019). How to Fine-Tune BERT for Text Classification? Retrieved from http://arxiv.org/abs/1905.05583

[4] Sun, A., & Lim, E.-P. (2001). Hierarchical text classification and evaluation. In Proceedings 2001 IEEE International Conference on Data Mining (pp. 521-528). DOI: 10.1109/ICDM.2001.989560.

[5] Wei, J., & Zou, K. (2019). Eda: Easy data augmentation techniques for boosting performance on text classification tasks. arXiv preprint arXiv:1901.11196.

[6] Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. Advances in neural information processing systems, 28.