ZeroShotDataAug: Generating and Augmenting Training Data with ChatGPT

Anonymous ACL submission

Abstract

In this paper, we investigate the use of data obtained from prompting a large generative language model, ChatGPT, to generate synthetic training data with the aim of augmenting data in low resource scenarios. We show that with appropriate task-specific ChatGPT prompts, we outperform the most popular existing approaches for such data augmentation. Furthermore, we investigate methodologies for evaluating the similarity of the augmented data generated from ChatGPT with the aim of validating and assessing the quality of the data generated.

1 Introduction

Data augmentation is a technique to increase the size of the training data available to machine learning models without requiring additional human annotation of data. Increasing the size of training data, provided the additional data is somewhat diverse, is pertinent to enable model generalization especially in low resource tasks. The aim of this paper is to evaluate zero-shot prompting of ChatGPT for data augmentation in the low resource scenario.

Wei and Zou (Wei and Zou, 2019) proposed Easy Data Augmentation (EDA) which is a technique based on word replacement that includes four types of operations: synonym replacement, random insertion, random deletion, and random swap. In synonym replacement, words with similar meanings are substituted for some of the original words in the text. This helps to introduce variations in the text and expand the range of vocabulary. Random insertion involves adding new words to the text, which are not present in the original data. This helps to increase the diversity of the text and can also help models learn to deal with out-of-vocabulary words. In random deletion, words are randomly removed from the text. This can help to simulate situations where some words may be missing in the input and can help the model become more robust to noise. In random swap, two words in the text are randomly swapped. This can help to introduce variations in the text and improve the diversity of the training data.

Back translation (Sennrich et al., 2015) is a common data augmentation technique in Natural Language Processing (NLP). It involves translating a sentence or text from one language to another and then translating it back to the original language using a machine translation model. Back translation can introduce variations in the text which can help to create more diverse and representative data for training NLP models. Researchers have also used pretrained autoencoder transformer models like BERT (Devlin et al., 2018), CBERT (Wu et al., 2019), and BART (Lewis et al., 2019) to augment text data in NLP (Kumar et al., 2020) (Wu et al., 2019). These techniques generally mask some words in the training set and utilize the pretrained models to predict the masked word(s). This could create more diverse data since the predicted words could vary from the original word. The authors include the class labels during finetuning and language modelling to aid the models in predicting the masked words in the context of their labels. Kumar et al. (Kumar et al., 2020) also utilzed an autoregressive pretrained language model, GPT-2 (Radford et al., 2019), to augment data by prompting GPT-2 to complete the sentence given only the first few words of the sentence and the training data label.

Dai et al. (Dai et al., 2023) utilized few-shot prompting of ChatGPT (OpenAI, b) for data augmentation to produce several variations of each sentence in the training sample. The generated sentences are similar in meaning but have different syntax. ChatGPT is a conversational agent that utilizes OpenAI's GPT-3.5 (OpenAI, a)—a large-scale language model trained on a vast corpus of diverse and information-rich web data and

Reinforcement Learning from Human Feedback (RLHF). Few-shot prompting is a technique that enables a language model to perform a new task with only a few examples of training data. Zero-shot prompting is a technique in which a language model is provided with a task description, rather than direct supervision or training data, to perform a specific task. The task description is in the form of a prompt or a question that guides the model on how to generate the desired output. In this paper, we investigate zero-shot prompting of ChatGPT for data augmentation. The main contributions of this paper are:

- Evaluation of zero-shot prompting of ChatGPT for data augmentation on multiple datasets.
- Two methodologies to evaluate the similarity of the data generated from zero-shot prompting of ChatGPT with the training and test sets with the aim of validating and assessing the quality of the data generated.
- Investigation of the marginal performance improvement due to the data generated from different data augmentation techniques.

2 Datasets

The datasets we use to evaluate our data augmentation methodology are popular benchmark natural language understanding datasets that researchers have utilized to evaluate other data augmentation techniques. We evaluate on three text classification datasets:

- **SST-2 (Socher et al., 2013)**: Stanford Sentiment Treebank consists of movie reviews from the Rotten Tomatoes website, where each example in the reviews is labeled with its sentiment polarity (positive or negative).The training set contained 6228 sentences while the test set contained 1821 sentences.
- **SNIPS (Coucke et al., 2018)**: The Spoken Natural Language Interaction for Personal Services dataset consists of annotated spoken queries related to seven intents in the domains of music, weather, and home automation. The training set contained 13084 sentences while the test set contained 700 sentences.
- **TREC (Li and Roth, 2002)**: This is a question classification dataset sourced from the Text Retrieval Conference. It contains six question types (indicating whether the question is about an abbreviation, description, entity, human, location, or numeric value). The training set contained 4906 sentences while the test set contained 500 sentences.

2.1 Low-Resource Data Scenario

In this research work, we evaluate the impact of data augmentation in the low-data scenario. We follow a similar approach to previous work (Kumar et al., 2020) on data augmentation by randomly subsampling only 10 examples per class on each task for both the training and the development sets.

To improve the model's performance on the low training data scenario, for each task, we incorporate the synthetic data generated by either ChatGPT or the comparison approaches. Subsequently, we assess the models' performance on the entire test set. To address any stochastic variation, we repeat all experiments 15 times.

3 Methods

3.1 Baseline Methods

In this research, we compare zero-shot prompting of ChatGPT with other popular data augmentation methods listed below (see original sources cited following the method names for further details):

- **EDA** (Wei and Zou, 2019): We used the random word replacement techniques that include four types of operations: synonym replacement, random insertion, random deletion, and random swap. We set α (proportion of words replaced) to 0.10 following the original eda research.
- **BackTrans** (Sennrich et al., 2015): We translated the training example from one language to another and then translated it back to the original language using a machine translation model.[1]
- **CBERT** (Wu et al., 2019): First, we utilized BERT's segment embeddings to condition the BERT model on the class labels during finetuning.[2] We then finetuned the model with the masked language model (MLM) objective which randomly masks some words in the sequences and aims to predict the original word using the context. Finally, we used the resulting model to

[1] Google Translate (https://pypi.org/project/googletrans/).

[2] In all baseline experiments that require finetuning, we trained for 20 epochs using AdamW optimizer with a learning rate of 5×10^{-5} and the cross entropy loss.

predict and replace masked words in the training set.

- **BERTexpand** (Kumar et al., 2020): First, we prepended the label to each sequence in the training data and added the labels to the model vocabulary before finetuning. We then finetuned the model with the MLM objective. Finally, we use the resulting model to predict and replace masked words in the training set.

- **BERTprepend** (Kumar et al., 2020): First, we prepended the label to each sequence in the training data without adding the labels to the model vocabulary before finetuning. We then finetuned the model with the MLM objective. Finally, we used the resulting model to predict and replace masked words in the training set.

- **GPT2context** (Kumar et al., 2020): First, we prepended the label to each sequence in the training data before finetuning GPT-2. Next, we finetuned the GPT-2 model on the MLM objective. Finally, we prompted the resulting model to complete the sentences given only the prepended label and the first three words of the training example.

- **BARTword** (Kumar et al., 2020): First, we prepended the label to each sequence in the training data without adding the labels to the model vocabulary. Next, we finetuned the BART model on the denoising reconstruction task where 40% of words are masked and the goal of the model is to reconstruct the original sequence. Finally, we used the resulting model to predict and replace the masked word in each example in the training set.

- **BARTspan** (Kumar et al., 2020): First, we prepended the label to each sequence in the training data without adding the labels to the model vocabulary. Next, we finetuned the BART model on the denoising reconstruction task where 40% of words are masked and the goal of the model is to reconstruct the original sequence. Finally, we used the resulting model to predict and replace the masked spans of words in the training set.

- **ChatGPTfew-shot** (Dai et al., 2023): We used few-shot prompting of ChatGPT for data augmentation to produce paraphrases of each sentence in the training set.

3.2 Prompts for Zero-shot Data Augmentation

In this section, we list the prompts used to generate augmentation examples for each class. The prompts were generated by observing the task, class description and five instances per class of the training data.

Dataset: SST-2
Class: Positive
Prompt: Generate 20 sentences that are positive reviews to a movie
Class: Negative
Prompt: Generate 20 sentences that are negative reviews to a movie

Dataset: SNIPS

Class: RateBook
Prompt: Generate 20 sentences in an imperative mood where a human tells a digital assistant to rate a random book and the human provides the numerical rating. Use random book names. Do not mention the name of the digital assistant.
Class: AddToPlaylist
Prompt: Generate 20 sentences in an imperative mood where a human tells a digital assistant to add music to a playlist and the human provides the music name. Use random music and playlist names. Do not mention the name of the digital assistant.
Class: PlayMusic
Prompt: Generate 20 sentences in an imperative mood where a human tells a digital assistant to play a music and the human provides the music name. Use random music and names. Do not mention the name of the digital assistant.
Class: BookRestaurant
Prompt: Generate 20 sentences in an imperative mood where a human tells a digital assistant to book a restaurant and the human provides the restaurant or food name. Use random restaurant and food names. Do not mention the name of the digital assistant.
Class: GetWeather
Prompt: Generate 20 sentences in an imperative mood where a human asks a digital assistant about the weather. Sometimes the human may provide the time and city. Use random city names. Do not mention the name of the digital assistant.
Class: SearchCreativeWork
Prompt: Generate 20 sentences in an imperative mood where a human asks a digital assistant to

find a specific creative work. The creative work could be a movie, tv show, book or game. Use random movie names, tv shows names, books, games. Sometimes the human asks for a specific creative work. Do not mention the name of the digital assistant.
Class: SearchScreeningEvent
Prompt: Generate 20 sentences in an imperative mood where a human asks a digital assistant to find information about a movie or screening in the theater. Sometimes the human asks for a specific movie. Do not mention the name of the digital assistant.

Dataset: TREC
Class: Abbreviation
Prompt: Generate 20 questions asking about the meaning of an abbreviation
Class: Entity
Prompt: Generate 20 questions asking about a random example of a noun or entity. Actually use different nouns or entities in each sentence.
Class: Description
Prompt: Generate 20 sentences that are only "what is" questions that query for a definition.
Class: Human
Prompt: Generate 20 questions about random facts about a person or people in history.
Class: Location
Prompt: Generate 20 sentences that are questions that ask the location of a place in history. Use a different place for each sentence
Class: Numeric Value
Prompt: Generate 20 sentences that are questions about a numeric fact in history

3.3 Evaluating the Similarity of the Generated Data from ChatGPT Versus the Training and Test Data

Since there is a chance that ChatGPT might have been trained on the datasets for some of our tasks, we investigate data contamination in all our datasets. We investigate data contamination by measuring the similarity between the data generated from ChatGPT and the datasets for each of our tasks using two similarity metrics: Cosine(Sentence Embedding) and the BLEU score (each detailed later in this section).

First, for each task, we compared each example generated by ChatGPT with all the examples from the training set and, separately, with the testing set. For each generated example, we selected the maximum similarity score and then averaged these scores over the generated dataset.

	Sentence Embedding	**BLEU Score**
ChatGPTzero-shot to SNIPStest	0.553	0.153
SNIPStrain to SNIP-Stest	0.593	0.394
ChatGPTzero-shot to TRECtest	0.528	0.383
TRECtrain to TRECtest	0.448	0.168
ChatGPTzero-shot to SST-2test	0.600	0.231
SST-2train to SST-2test	0.535	0.243

Table 1: Data augmentation similarity results of ChatGPTzero-shot versus the original training sets relative to the testing sets.

The two metrics used to evaluate similarity were:

- **Sentence Embedding**: We used the MiniLM model (Wang et al., 2020) in the sentence transformer library (SBERT.net) to obtain the embedding of each example. We calculated the cosine similarity between the embeddings of the pair of examples.

- **BLEU Score**: We used the sentence BLEU score function from NLTK library ("Bird and Klein", 2009) to obtain the similarity between the pair of examples. We used up to 3-grams with equal weights for the BLEU score calculations.

Table 1 reveals that the similarity between ChatGPTzero-shot generated data and testing datasets are 3.5% higher on average compared to similarity between the original training and testing sets for sentence embedding similarity. This implies that there is high semantic similarity between the ChatGPTzero-shot generated data and testing datasets which is ideal for data augmentation. On the other hand, using the BLEU score to compare training to testing datasets, ChatGPTzero-shot's BLEU score is on average 0.013 lower than the score between the original training and testing sets. The lower n-gram overlap implies there was little to no data memorization by ChatGPT (this

	Sentence Embedding	BLEU Score
ChatGPTzero-shot to SNIPStrain	0.629	0.284
SNIPStrain to SNIPStrain	0.708	0.646
ChatGPTzero-shot to TRECtrain	0.634	0.594
TRECtrain to TRECtrain	0.622	0.419
ChatGPTzero-shot to SST-2train	0.635	0.455
SST-2train to SST-2train	0.583	0.377

Table 2: Data augmentation similarity results between ChatGPTzero-shot and training sets versus similarity within the training sets.

	BLEU Score Similarity		
	> 0.66	Max	% Exs at Max
ChatGPTzero-shot to SNIPStest	0.0%	0.61	1.25%
SNIPStrain to SNIPStest	20.2%	1	1.73%
ChatGPTzero-shot to TRECtest	12.9%	1	3.75%
TRECtrain to TRECtest	2.2%	1	0.24%
ChatGPTzero-shot to SST-2test	0.0%	0.65	1.25%
SST-2train to SST-2test	1.9%	1	0.16%

Table 3: Three statistics for the BLEU score: the percentage of examples where the BLEU score is greater than .66, the maximum BLEU score, and the percentage of examples with the maximum BLEU score

is explored further below). Table 2 reveals that ChatGPTzero-shot generated data is essentially as similar to training data examples as the training dataset is to itself (only an average of 0.5% lower sentence embedding cosine and 0.036 lower BLEU score). This again indicates the high semantic quality of ChatGPTzero-shot generated data with relatively little n-gram overlap, and hence, suggests it is very unlikely there was any memorization of data during pretraining of the language model.

Table 3 further explores the likelihood that ChatGPT simply reproduced any testing data it may have seen during pretraining. For the SNIPS dataset, the original training data had a 0.24 higher BLEU score than did the ChatGPT data when comparing them to the testing data. The most similar generated examples to the testing set had a BLEU score of 0.61 (1.25% of the generated examples); whereas, 1.76% of the original training examples had a BLEU score of 1.0. Together, these statistics imply the generation did not rely on memorization by the underlying pretrained LLM (ChatGPT) for the SNIPS task. For the TREC testing data, the generated training data had a 0.22 higher BLEU score than the original training data comparing to the testing data; while for SST, the original training data had a 0.1 higher BLEU score than did the generated data. For TREC, 3.75% of the generated examples and 0.24% of the original training examples had a BLEU score of 1.0 comparing to the testing dataset. For the generated examples, this very high BLEU score is associated with common very short, questions (specifically, the three questions: "Who invented the telephone?", "What year did the Titanic sink?", and "What is a tsunami?"). Given the small number of such questions, we don't see this as a very worrisome problem. For SST-2, the maximum BLEU score with a testing example for the generated data was only 0.65; whereas, 0.16% of the original training examples had a BLEU score of 1.0.

Therefore the results in Tables 1, 2 and 3 combined revealed that there is little to no data contamination from the data generated by ChatGPT for any of our tasks. The investigation also reveals that the data generated by ChatGPT is on average 32% more similar to the training datasets than the testing datasets. We believe this difference stems from the higher number of examples in the training datasets relative to testing – given we are calculating the maximum similarity between each example and the entire dataset, a larger dataset increases the probability that there will be an example with a higher similarity. Again, these statistics imply ChatGPT did not rely on memorization of the testing data during its pretraining. Hence, results reported in the following sections should be predictive of what others will achieve on new tasks that are relatively similar in nature.

3.4 Model Implementation

We finetuned the pretrained BERT-base uncased model for all three text classification tasks. The computational infrastructure used was the NVIDIA Tesla V100 SXM2 16 GB. The hyperparameters we used to finetune BERT were similar to the hyperparameters used by Kumar et al. (Kumar et al., 2020) for a fair comparison. The hyperparameters were a training time of twelve epochs, batch size of 32, the AdamW optimizer function with a learning rate of 5×10^{-5}, and the categorical cross entropy loss function. During the training the models are saved after each epoch and the best performing model on the development set is used for inference on the testing set. All the experiments utilized a GPU time of about 40 minutes.

4 Results

As seen in Table 4, with zero-shot prompting of ChatGPT for data augmentation, our model achieves an accuracy of 76.3%, 91.7%, and 74.4% for SST-2, SNIPS, and TREC, respectively. Zero-shot prompting of ChatGPT outperformed all existing data augmentation methods on all three tasks. Specifically, our model surpassed the next best model by 9%, 2%, and 4% on SST-2, SNIPS, and TREC, respectively. It is also noteworthy that ChatGPTzero-shot had the lowest variance for both SNIPS and TREC, while being at the median for SST-2.

Model	SST-2	SNIPS	TREC
No Aug	61.0 (4.3)	84.3 (2.5)	49.7(12.0)
EDA	64.3 (5.4)	88.7 (1.9)	65.4 (7.9)
BackTrans.	65.1 (6.0)	89.8 (1.5)	66.4 (7.4)
CBERT	65.2 (6.8)	88.3 (3.1)	65.4(10.7)
BERTexpand	64.5 (6.6)	88.4 (2.4)	65.8 (6.3)
BERTprepend	64.4 (6.2)	88.9 (1.2)	65.2 (9.5)
GPT2context	63.8 (6.8)	89.0 (2.3)	63.3 (9.8)
BARTword	67.1 (6.7)	89.5 (2.2)	64.3 (9.9)
BARTspan	65.7 (7.2)	89.6 (1.3)	70.4 (6.3)
ChatGPTfew-shot	67.6 (5.5)	89.5 (1.5)	67.6 (7.4)
ChatGPTzero-shot	**76.3** (6.5)	**91.7** (0.9)	**74.4** (5.1)

Table 4: Accuracy (and standard deviation) for each data augmentation method.

5 Augmentation Learning Curve

In this section, we investigate how performance is impacted by the number of examples created through augmentation for all data augmentation methods. In Figure 1, we show the average [3] performance over all three datasets (SST-2, TREC, SNIPS) for K augmentation examples per original training example for different data augmentation techniques.[4]

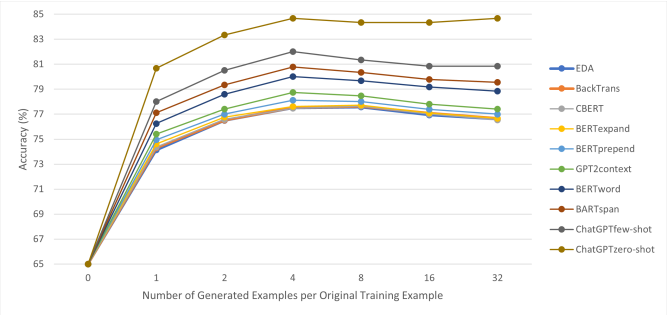

Figure 1: Average accuracy on SST, TREC, and SNIPS as the number of generated examples per original training example increases

As seen in Figure 1, ChatGPTzero-shot outperforms all other data augmentation methods for $K \in \{1, 2, 4, 8, 16, 32\}$. This provides further evidence on the effectiveness of using ChatGPT for data augmentation in low resource scenarios.

6 No Training Data Scenario

In this section, we investigate the performance of our model on SST-2, SNIPS and TREC's test datasets when we use augmented data from zero-shot prompting of ChatGPT without any data from the original training datasets. We use the same prompts described in section 3.2 to generate 20 instances per class. We train the models using the same hyperparameters as in section 3.4. We repeat both steps 15 times to account for stochasticity. Our model obtained a mean accuracy of 0.80, 0.78, and 0.62 on SST-2, SNIPS and TREC's designated test sets, respectively. Interestingly, with no data from the original training set, our model outperforms all existing augmentation methods on SST-2 and within a standard deviation of the best result on TREC. These results further highlight the significance of zero-shot prompting of ChatGPT for

[3] In the Appendix, Figures 2, 3 and 4, show the individual performance of different data augmentation techniques on the SST-2, TREC, SNIPS

[4] For Back Translation, we used K different languages.

data augmentation. Unlike other techniques, our method of data augmentation can be used even in the absence of any training data.

7 Performance on a Post ChatGPT Dataset

To further investigate ChatGPTzero-shot for data augmentation while controlling for the possibility of data contamination, we use a dataset that was both developed and made public after the release of ChatGPT in November 2022. We use the TURKISH EARTHQUAKE dataset (Kaggle.com) which is a tweet classification dataset in the Turkish language sourced from Twitter during the February 2023 earthquake in Turkey. The dataset contains two classes. The positive class consists of tweets from people asking for help during the earthquake, while the negative class consists of tweets that are in the context of the Turkish earthquake but not asking for help. All inputs to ChatGPT were in the English language for consistency with previous experiments. However, in this section ChatGPT was also prompted to respond to every prompt in Turkish. Furthermore, we used the Turkish implementations of BERT (Huggingface.co), and GPT-2 (huggingface.co). For EDA, we used the Turkish WordNet (Software) in place of the English WordNet for the synonym dictionary. For Back Translation, we used Turkish as both source and the target language. We did not run any experiments with BART on the dataset in this section since a Turkish implementation of BART is currently unavailable.

Similar to previous experiments, we executed 15 training iterations to account for stochaticity. In each iteration of training, we used 10 training and 10 validation examples per class. Unlike previous datasets, this dataset does not have a pre-designated train/validation/test split, therefore, on each iteration of training, we chose 10 random examples per class for both training and validation and used the rest of the data as the test set. Similar to previous experiments, for ChatGPTzero-shot, we have generated 150 examples per class and randomly chose 10 examples per class for each iteration of training.

The prompts used to generate augmentation examples for both classes of the Turkish earthquake dataset are:
Class: Positive
Prompt: create 10 examples of tweets asking for help in an earthquake**Class**: Negative
Prompt 1: create 5 examples of tweets related to earthquake in general but not directly asking for help.
Prompt 2: create 5 examples of tweets angry at the government for the bad earthquake response.
Prompt 3: create 5 examples of tweets that contain general unhappiness with the Turkish government.

Model	TURKISH EQ
No Aug	86.6 (2.3)
EDA	87.3 (1.7)
BackTrans.	88.2 (1.3)
CBERT	87.8 (2.7)
BERTexpand	88.9 (2.1)
BERTprepend	88.6 (1.4)
GPT2context	87.4 (2.1)
ChatGPTfew-shot	89.3 (1.2)
ChatGPTzero-shot	**90.1** (1.0)

Table 5: Accuracy (and standard deviation) for each data augmentation method on the TURKISH EARTHQUAKE dataset

As seen in Table 5, the experiments in this section reveal that ChatGPTzero-shot outperforms all existing data augmentation approaches in the low-resource scenario. These results provide more evidence that the performance of ChatGPTZero-shot on SST-2, SNIPS and TREC is not due to data contamination during the training of ChatGPT. These results also provide evidence of the applicability of ChatGPT to low-resource languages.

8 Discussion

This research provides an easy-to-use and intuitive methodology for generating augmented data. In our experiments, our method of augmenting training data for these NLP tasks by zero-shot prompting ChatGPT shows great promise in the low resource scenario, substantially outperforming all of the baseline methods. It should however be noted that, as with other data augmentation methods, zero-shot prompting of ChatGPT for data augmentation does not necessarily improve results where there is a large training dataset.

One of the limitations of previous data augmentation methods is that the quality of augmented data strongly depends on the original training dataset. The quality of data generated from zero-shot prompting of ChatGPT is not limited by the human-annotated training data. This research also provides evidence that data generated from zero-shot prompting of ChatGPT continues learning

with more generated data compared to many existing data augmentation techniques. Furthermore, on SST-2, our model achieved better results using only data from zero-shot prompting of ChatGPT than did other data augmentation approaches that supplemented an initial human-annotated training dataset. This further highlights the effectiveness of zero-shot prompting of ChatGPT as a data augmentation approach.

It is important to note that the quality of augmented data generated by zero-shot prompting of ChatGPT depends on the quality of the prompts. The prompts in this research were human generated based on the task description and observing a few training data instances. While there is a lot of current research in the area of prompt engineering, there are still no task-independent well-established best practices for how to generate effective prompts. This research presents a methodology for evaluating the augmented data generated from large language models. To evaluate the augmented data, we calculated the sentence embedding similarity and BLEU scores of the synthetic examples compared to all the examples in the training and test data. This revealed that there was very little data generated with high similarity scores, making it unlikely that ChatGPT was regenerating data it memorized during its training. Furthermore, we evaluated zero-shot prompting of ChatGPT on a dataset developed after the release of ChatGPT. This provided further evidence that the superior performance of the approach does not stem from ChatGPT being trained on the task's data.

9 Limitations

One limitation of this work is that the quality and relevance of the synthetic data generated by ChatGPT heavily depends on the quality of the prompts, which in this research are all human generated. If the prompts are not reasonably designed by humans it is possible the generated data might not lead to improvements in model performance. It is also possible that data augmentation might not lead to improved model performance when there is already a substantial amount of high-quality training data available. Moreover, the augmentation techniques in this research have been exclusively assessed within the realm of text classification. Hence, it is important to conduct further evaluations across a broader spectrum of natural language processing tasks to ascertain the efficacy of these data augmentation methods in contexts beyond text classification.

10 Ethical Concerns

The use of generative models, like ChatGPT, can introduce or propagate biases present in its training data. Ethical considerations involve identifying and mitigating biases to ensure fairness and equitable representation in the augmented data. For example, it is possible that the language of different culture groups might have subtle differences in regard to how sentiment is expressed and therefore should such issues should be taken into account in a real-world application.

11 Conclusion and Future Work

This paper proposes a novel technique for generating augmented data for machine learning tasks using zero-shot prompting of ChatGPT. The experimental results demonstrate that the proposed method substantially outperforms all the baseline approaches for all four tasks investigated in this research. This indicates our method's potential as a promising data augmentation method in the low resource setting.

The data augmentation approach investigated in this research relies on manually engineering effective prompts for each task which requires some expertise. Future researchers can explore more systematic approaches to prompt engineering especially for tasks that cannot be adequately described within a concise prompt of one to three sentences.

References

Edward Loper "Bird, Steven and Ewan Klein". 2009. *"Natural Language Processing with Python"*. "O'Reilly Media Inc".

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Zihao Wu, Lin Zhao, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, et al. 2023. Chataug: Leveraging chatgpt for text data augmentation. *arXiv preprint arXiv:2302.13007*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep

bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Huggingface.co. Turkey bert. [Accessed: 07.04.2023].

huggingface.co. Turkish gtp2. [Accessed: 07.04.2023].

Kaggle.com. Turkey earthquake relief tweets dataset. [Accessed: 07.04.2023].

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

OpenAI. a. Gpt-3.5. [Accessed: 07.04.2023].

OpenAI. b. Introducing chatgpt. [Accessed: 07.04.2023].

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

SBERT.net. Sentencetransformers documentation. [Accessed: 07.04.2023].

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Starlang Software. Turkish wordnet. [Accessed: 07.04.2023].

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *Computational Science–ICCS 2019: 19th International Conference, Faro, Portugal, June 12–14, 2019, Proceedings, Part IV 19*, pages 84–95. Springer.

12 Appendix

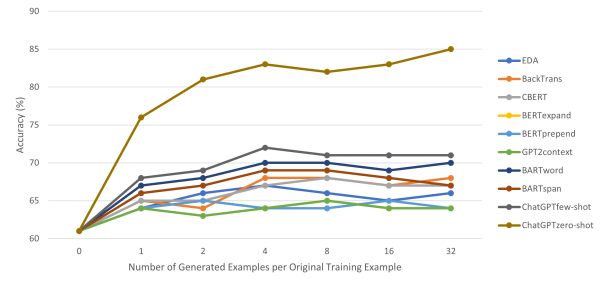

Figure 2: Accuracy on SST as the number of generated examples per original training example increases

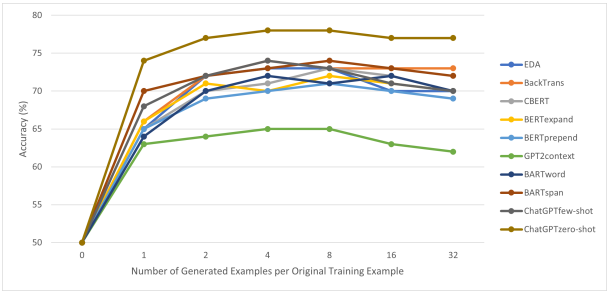

Figure 3: Accuracy on TREC as the number of generated examples per original training example increases

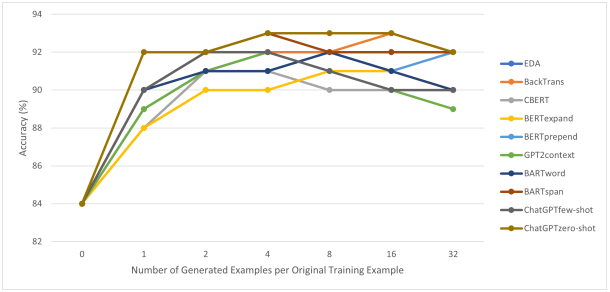

Figure 4: Accuracy on SNIPS as the number of generated examples per original training example increases