Mini-DA: Improving Your Model Performance through Minimal Data Augmentation using LLM

Shuangtao Yang[*]**, Xiaoyi Liu**[*]**, Xiaozheng Dong**[*]**, Bo Fu**
Lenovo Knowdee (Beijing) Intelligent Technology Co., Ltd., Beijing, China
{yangst, liuxy, dongxz, fubo}@knowdee.com

Abstract

When performing data augmentation using large language models (LLMs), the common approach is to directly generate a large number of new samples based on the original dataset, and then model is trained on the integration of augmented dataset and the original dataset. However, data generation demands extensive computational resources. In this study, we propose Mini-DA, a minimized data augmentation method that leverages the feedback from the target model during the training process to select only the most challenging samples from the validation set for augmentation. Our experimental results show in text classification task, by using as little as 13% of the original augmentation volume, Mini-DA can achieve performance comparable to full data augmentation for intent detection task, significantly improving data and computational resource utilization efficiency.

1 Introduction

Data is the lifeblood of deep learning models, and the availability of high-quality data is crucial for achieving strong model performance. However, acquiring such data can be a challenge, particularly in scenarios where data is limited or unavailable. Moreover, human annotation, a common method for obtaining labeled data, is known to be financially expensive and time-consuming. As such, data augmentation techniques have become increasingly important, especially in scenarios where data is limited.

Data augmentation has been studied for a long time in various domains, with rule-based method, data interpolation techniques, and model based approaches explored (Feng et al., 2021; Hedderich et al., 2021). While these traditional data augmentation methods have shown effectiveness, the rapidly evolving field of large language models (LLMs) has ushered in a new era of augmentation methods for natural language processing tasks. With their remarkable ability to generate human-like text, LLMs have enabled generative data augmentation techniques that can create more diverse and realistic synthetic samples, potentially leading to improved model performance. However, as highlighted in the comprehensive survey by (Ding et al., 2024), the generation of extensive augmented datasets can cause significant expenses due to the demands of considerable computational resources, especially for SOTA models.

To address the limitation of data augmentation with LLMs, we propose Mini-DA, a novel framework that aims to maximize the benefits of LLM-based data augmentation while minimizing the associated costs. The key innovation of Mini-DA lies in its ability to leverage the prediction result of the target model on the validation set during k-fold cross-validation to identify "challenging samples" that the model struggles to predict correctly. Then, for these difficult samples, a instruction-tuned large language model is used to generate synthesized data based on a given query and its label. This process is repeated iteratively, collecting augmented data until the model's performance on the test set stabilizes. Through our experiments on two datasets for the intent detection task, we demonstrate that by focusing augmentation efforts on a limited number of difficult samples, Mini-DA significantly reduces the augmentation volume compared to full data augmentation, leading to substantial savings in computational resources while still producing comparable performance.

2 Related Work

2.1 Pre-LLM Data Augmentation

Data augmentation has been widely studied before the advent of LLMs. Various approaches were in-

[*] These authors contributed equally to this work

Proceedings of the Fifth Workshop on Data Science with Human-in-the-Loop (Language Advances), pages 25–30
June 20, 2024 ©2024 Association for Computational Linguistics

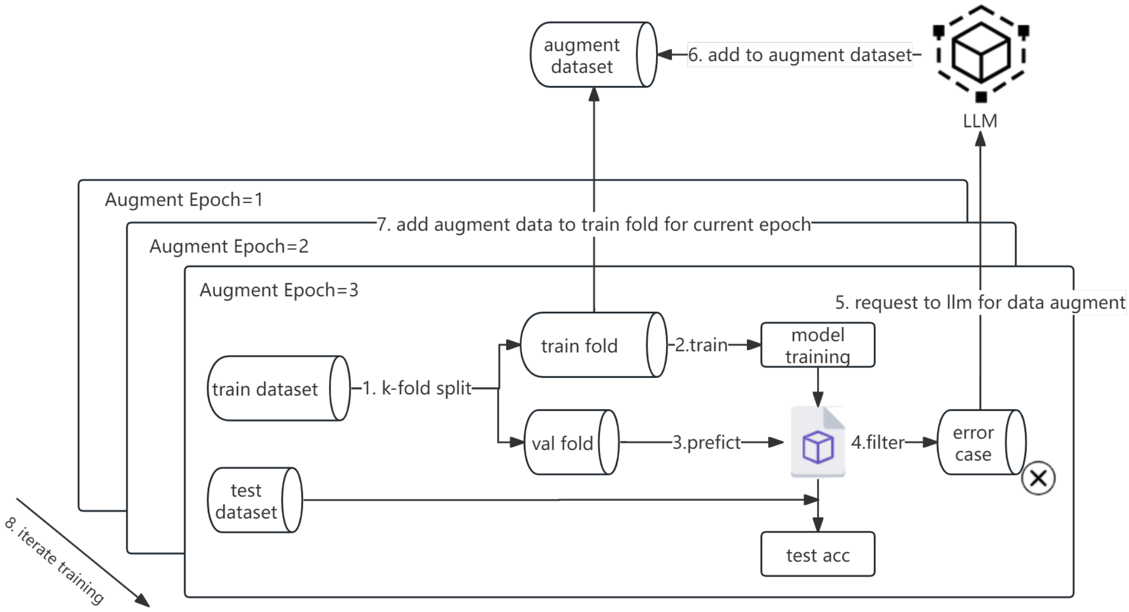

Figure 1: **Mini-DA framework.** The figure shows the iterative augmentation process. (1) One iteration begins by splitting the original dataset into k-folds. (2) Models are then trained on the training folds. (3) The trained models are evaluated on the validation folds, and (4) error cases are selected. (5) A LLM is instructed to perform data augmentation on the selected error samples. (6) The augmented data generated by the LLM is added to the augmented dataset. (7) In the next epoch, the dataset is re-split, and any existing augmented data corresponding to samples in the new training folds is integrated.

vestigated, including rule-based methods like Easy Data Augmentation (EDA) proposed by Wei and Zou (2019). EDA introduced token-level operations such as random insertion, deletion, and swapping. At sentence-level data augmentation, paraphrasing is widely adopted. The most popular one is Backtranslation (Sennrich et al., 2016), which uses Seq2seq and language models to translate a sequence into another language and then back into the original language.

2.2 Data Augmentation with LLMs

With the emergence of Large Language Models (LLMs), data augmentation techniques have undergone significant refinement and innovation. LLMs possess capabilities for generating high-quality, diverse, and contextually relevant text, enabling novel approaches to data augmentation.

One of the most common data augmentation method employing LLMs is to use them as data generators. Chintagunta et al. (2021) utilize powerful models such as GPT-3 to synthesize medical dialogue summaries. By training models on a combination of synthesized and human-annotated data, their approach effectively scales a small set of human-annotated examples to achieve performance comparable to using a significantly larger human-annotated dataset. Møller et al. (2024) employs LLMs to generate examples for specific labels in low-resource classification scenarios, by providing an example and its corresponding label. Lin et al. (2023) uses instruction tuned LLM, GPT-3.5, to generate examples within the context of the training set and subsequently filtered out unhelpful examples. For intent detection task, Sahu et al. (2022) introduces a prompting-based data augmentation using GPT-3, , and demonstrates its effectiveness in improving classifier performance, especially when combined with filtering techniques to address challenges in generating data for closely-related intents.

Another common approach involves using LLMs to reformulate existing data to more diverse variations. These techniques are proved to be particularly valuable in tasks like counterfactual generation, where existing data is transformed into its counterfactual version. For instance, Chen et al. (2023) employs LLMs to generate high-quality counterfactual data on a large scale. CORE (Dixit et al., 2022) also uses GPT-3 for retrieval-augmented generation (RAG), generating counterfactual edits conditioned on retrieved excerpts from the input. These perturbations serve to reduce model bias and enhance performance.

3 Method

In the following section, we describe our proposed iterative LLM-in-the-loop data augmentation approach Mini-DA, as illustrated in Figure 1. At each iteration, we leverage the feedback from the target model to identify difficult examples from validation set and instruct LLM to only augment these selected samples.

The Mini-DA process can be broken down into the following steps:

1. **Dataset Splitting** If the original dataset does not come with a predefined test set, we first split a portion of the data to create a held-out test set. This test set will be used for monitoring the model's performance and determining convergence during the iterative process. The remaining dataset is then split into k folds. This process employs stratified sampling to ensure that both sets are representative of the underlying data distribution.

2. **Model Training** The target model is trained k times, using one different fold for validation and the remaining k-1 folds are combined for training each time.

3. **Validation Set Prediction** After training, the best models saved from the training stage are then evaluated on their own validation set.

4. **Challenging Case Collection** Error case from prediction of each fold on validation set are collected and identified as challending samples

5. **Selective Data Augmentation** The prediction errors are then input to an LLM with predefined data augmentation prompt, and obtain a set of synthetic examples as augmented data.

6. **Augmented Dataset** Data generated from last step is then added to the augmented dataset, and we maintain a augmented dataset mapping between each original sample and its corresponding augmented data for future use.

7. **Augmented Data Integration** For the next augmenting epoch, the dataset is re-split into new k folds. And k new training and validation set pairs are formed. Before training on the new training sets, we check if any samples in each new training set have corresponding augmented data in augmented dataset. If so, we incorporate those augmented samples into each training set. Each training set should only contain augmented samples that are generated from original data it contains.

a.
You are an experienced data annotator. Please generate five user questions following the requirements below.
1. Focus on the "banking" domain;
2. Should focus on "**{intent_label}**" intent, which represent **{intent_definition}**;
3. The newly generated sentence needs to be semantically similar to sentence: "**{query}**";

b.
You are an experienced data annotator. Please generate five user questions following the requirements below.
1. Focus on the "**{domain_label}**" domain;
2. Should focus on "**{intent_label}**" intent, which represent **{intent_definition}**;
3. The newly generated sentence needs to be semantically similar to sentence: "**{query}**";
4. Newly generated sentences need to be in Chinese;

Figure 2: The prompts used to generate augmented data for a. banking77 dataset and b. ECDT-NLU-2019 dataset

8. **Iterative Process** Steps 2 through 7 are repeated for a predetermined number of epochs or until a convergence criterion is met, which is typically when the model's performance on a held-out test set stops improving across a predetermined number of epochs. At this point, the augmentation of the original dataset is completed.

4 Experiments Setup

4.1 Datasets and Task

To verify the effectiveness of our approach, we conduct experiments on two intent detection datasets, including banking77 (Casanueva et al., 2020) and ECDT-NLU-2019[1].

[1] http://conference.cipsc.org.cn/smp2019/evaluation.html

The original banking-77 is an English dataset in the banking domain, which includes 10,003 training and 3,080 test cases labeled with 77 intent. Since our primary focus is on enhancing the model performance in data limited scenario, we sampled a subset from banking77 for our experiments. We will refer the sampled dataset as banking77-filtered in this paper. Banking77-filtered includes 2,047 training and 693 test cases, which still has 77 intent labels.

The original ECDT-NLU-2019 is a Chinese natural language understanding dataset consisted of multiple tasks, including domain classification, intent detection, and slot filling. We only considered the intent detection task in our experiments. This datasets comprises 2,061 training and 516 test cases with 45 intent labels.

4.2 Models

Since the two datasets we used for our experiments are in different languages, we selected bert-base-multilingual-uncased[2] (Devlin et al., 2018) as our base model for training and prediction.

We use GPT-3.5 Turbo as the large language model to generate augmented dataset. The prompts used to augmented each dataset is illustrated in Figure 2.

4.3 Implementation Details

During the data splitting step, we set $k = 5$ for 5-fold cross-validation. In each augmenting epoch, we train bert-base-multilingual-uncased for 30 training epochs with a batch size of 64, learning rate of $2e - 5$ and the Adam optimizer (Kingma and Ba, 2017).

The stopping criterion for the iterative augmenting process is set to the average accuracy stop improving on test set for 2 consecutive augmenting epochs. For both datasets, we run the augmenting process for a maximum of 10 epochs.

4.4 Baseline Methods

We compare our proposed method with two baseline methods. It is important to note that our primary focus is on proposing an efficient framework for data augmentation by contrasting full-dataset augmentation with selective augmentation. Therefore, we include a basic prompt-based data augmentation method using a LLM as our baseline. However, the augmentation component (step 5) in our framework is modular and can be modified to other augmentation methods with LLM depending on the specific use case.

1. Baseline 1: We performed 5-fold cross-validation on the same base model, bert-base-multilingual-uncased, using the original, unaugmented training sets and the same hyperparameters in 4.3.

2. Baseline 2: we performed full data augmentation by generating augmented samples for every instance in the training set using GPT-3.5 Turbo with the prompts specified in Figure 2. We then conducted 5-fold cross-validation, where for each fold, the augmented data generated from that fold's training set was integrated into the corresponding fold's training set, ensuring no data augmented from the validation fold was trained on. The same hyperparameters in 4.3 were employed.

4.5 Evaluation Metrics

Considering the imbalanced class distribution present in the two selected datasets, we utilized accuracy as the evaluation metric to assess and compare the model performance across all methods on both datasets.

5 Result and Analysis

In this section, we present the experimental results obtained by evaluating our proposed Mini-DA method and the two baseline approaches on the selected datasets. We report and analyze the performance of each method in terms of the average accuracy of 5-fold cross-validation. Results are shown in Figure 3, Table 1, Figure 4, and Table 2.

For banking77-filter dataset, results shown in Figure 3, the average accuracy on test set of models trained on the original dataset achieved 80.52% (the dotted green line), while average accuracy models trained on the fully augmented dataset reaches 86.41% (the green line), representing a 5.89% improvement from unaugmented baseline. The red line represents the average accuracy on the test set when using the Mini-DA framework for training set augmentation crossing augmentation epochs. At the second augment epoch, Mini-DA achieved an average accuracy of 86.64%, which is even 0.23% higher than the result obtained using the fully augmented dataset, despite only augmenting 24% of the training data. When progressing to the fifth

[2] https://huggingface.co/google-bert/bert-base-multilingual-uncased

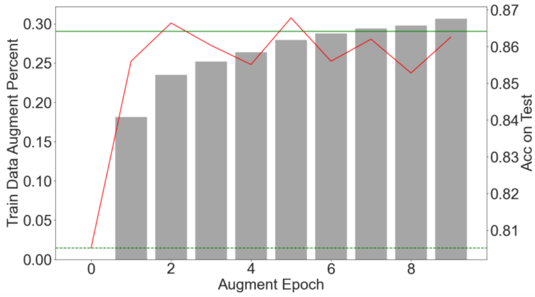

Figure 3: Accuracy on the banking77-filtered test set for the Mini-DA approach (red line) compared to the fully augmented dataset (green line at top) and the original, unaugmented dataset (dotted green line at bottom) across augmentation epochs. The bars indicate the sum of total augmented data added to the training set at each epoch.

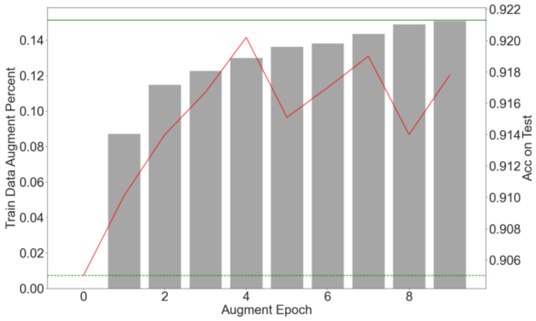

Figure 4: Accuracy on the ECDT-NLU-2019 test set for the Mini-DA approach (red line) compared to the fully augmented dataset (green line at top) and the original, unaugmented dataset (dotted green line at bottom) across augmentation epochs. The bars indicate the sum of total augmented data added to the training set at each epoch.

Method	**Augment Epoch**	**Number of Augmented Data added to Training sets**	**Average ACC on Test set**
No Augmentation		0	0.8052
Full Augmentation		2047	0.8641
Mini-DA	1	372	0.8560
	2	482	0.8664
	3	516	0.8603
	4	541	0.8551
	5	573	**0.8678**
	6	590	0.8560
	7	603	0.8620
	8	610	0.8528
	9	628	0.8626

Table 1: Results of banking77-filter

Method	**Augment Epoch**	**Number of Augmented Data added to Training sets**	**Average ACC on Test set**
No Augmentation		0	0.9050
Full Augmentation		2061	0.9213
Mini-DA	1	180	0.9101
	2	237	0.9140
	3	253	0.9167
	4	268	0.9202
	5	281	0.9151
	6	285	0.9170
	7	296	0.9190
	8	307	0.9140
	9	311	0.9178

Table 2: Results of EDTC-NLU-2019

augment epoch, Mini-DA achieved its optimal performance while a total of 573 training data points were augmented, accounting for 28% of the training set. On the test set, the average ACC reached 86.78%, an improvement of 0.37% compared to the average accuracy using the full augmented data.

On the EDTC-NLU-2019 dataset (shown by Figure 4), we observed similar phenomena. At the fourth augment epoch, the average accuracy on the test set reached 92.02%, which is only 0.11% lower than the result obtained using the fully augmented dataset (92.13%). However, at this point, Mini-DA only augmented 13% of the training data, resulting in a reduction of 1793 GPT-3.5 Turbo requests compared to the full data augmentation approach.

Through these experiments, we can observe that for intent detection scenarios with low resources, Mini-DA effectively combines two stages: fine-tuning on the target model and data augmentation using a LLM. By employing a cross-validation approach to selectively augment difficult samples from the validation set, Mini-DA avoids unnecessary augmentation of correctly predicted samples in the training set, thereby reducing the cost of data augmentation.

6 Conclusion

In this work, we present Mini-DA to efficiently augment intent detection data with LLMs. We design a iterative LLMs-in-the-loop framework that incorporates feedback from fine-tuning stage of target model to generate an augmented dataset. The results demonstrate that with as little as 13% of the augmented data generated, we can achieve comparable performance to full data augmentation on intent detection task in data-limited scenarios. Overall, Mini-DA presents a promising solution for data augmentation which significantly reducing computational costs and improving data efficiency.

For future work, our plan involves conducting comprehensive experiments across various tasks, including but not limited to question answering, text generation, and text retrieval. We believe this approach can be effective in improving model performance across a wide range of tasks in a data-efficient manner.

References

Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on NLP for ConvAI - ACL 2020*. Data available at https://github.com/PolyAI-LDN/task-specific-datasets.

Zeming Chen, Qiyue Gao, Antoine Bosselut, Ashish Sabharwal, and Kyle Richardson. 2023. DISCO: Distilling counterfactuals with large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5514–5528, Toronto, Canada. Association for Computational Linguistics.

Bharath Chintagunta, Namit Katariya, Xavier Amatriain, and Anitha Kannan. 2021. Medically aware GPT-3 as a data generator for medical dialogue summarization. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 66–76, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Bosheng Ding, Chengwei Qin, Ruochen Zhao, Tianze Luo, Xinze Li, Guizhen Chen, Wenhan Xia, Junjie Hu, Anh Tuan Luu, and Shafiq Joty. 2024. Data augmentation using llms: Data perspectives, learning paradigms and challenges. *Preprint*, arXiv:2403.02990.

Tanay Dixit, Bhargavi Paranjape, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. CORE: A retrieve-then-edit framework for counterfactual data generation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2964–2984, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for nlp. *Preprint*, arXiv:2105.03075.

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *Preprint*, arXiv:1412.6980.

Yen-Ting Lin, Alexandros Papangelis, Seokhwan Kim, Sungjin Lee, Devamanyu Hazarika, Mahdi Namazifar, Di Jin, Yang Liu, and Dilek Hakkani-Tur. 2023. Selective in-context data augmentation for intent detection using pointwise V-information. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1463–1476, Dubrovnik, Croatia. Association for Computational Linguistics.

Anders Giovanni Møller, Jacob Aarup Dalsgaard, Arianna Pera, and Luca Maria Aiello. 2024. The parrot dilemma: Human-labeled vs. llm-augmented data in classification tasks. *Preprint*, arXiv:2304.13861.

Gaurav Sahu, Pau Rodriguez, Issam H. Laradji, Parmida Atighehchian, David Vazquez, and Dzmitry Bahdanau. 2022. Data augmentation for intent classification with off-the-shelf large language models. *Preprint*, arXiv:2204.01959.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.