

# Exploring Data Augmentation Strategies for Labeled Data Enrichment in Low-Resource Language Fine-Tuning of LLMs

Pettugani Hotragn<sup>(1)</sup>, T.kinshuk sunder reddy<sup>(2)</sup>, Kakarla Mohan Kumar<sup>(3)</sup>

<sup>1</sup>Woxsen University, University in Telangana, India

<sup>2</sup>Woxsen University, University in Telangana, India

<sup>3</sup>Satya Institute of Technology and management, Andhra Pradesh, india

\*\*\*

**Abstract:** As large language models (LLMs) continue to demonstrate remarkable performance across various natural languages processing (NLP) tasks, their adaptability to low-resource languages poses a significant challenge. Limited labeled data availability in such languages hinders the fine-tuning process, impacting the model's ability to capture language nuances effectively. This study delves into the exploration of data augmentation strategies to address the issue of labeled data scarcity in the fine-tuning of LLMs for low-resource languages.

The research investigates diverse data augmentation techniques tailored to linguistic characteristics, aiming to enrich the labeled dataset and enhance model generalization. Augmentation methods such as synonym replacement, paraphrasing, and contextual word insertion are examined to ascertain their effectiveness in mitigating the data sparsity challenge. Furthermore, linguistic constraints specific to low-resource languages are considered to design augmentation approaches that respect language syntax, semantics, and cultural context.

To evaluate the proposed strategies, experiments are conducted using a benchmark dataset representative of the low-resource language under consideration. Comparative analyses are performed to assess the impact of each augmentation technique on the model's performance in various downstream tasks. Additionally, the study explores the trade-offs between increased data diversity and potential introduction of noise, providing insights into the optimal balance for effective fine-tuning.

The findings of this research contribute to the growing body of knowledge on adapting state-of-the-art language models to low-resource languages. The identified data augmentation strategies offer practical insights for researchers and practitioners working on NLP tasks in linguistically diverse and underrepresented languages. By addressing the challenges of labeled data scarcity, this study aims to facilitate the broader application of LLMs in real-world scenarios across a spectrum of languages, fostering inclusivity and linguistic diversity in natural language processing applications.

**Key Words:** Natural language processing (NLP), Machine learning (ML), Large language model (LLM), Word embedding, Deep Learning (DL), Data Augmentation, Label Data

## 1. INTRODUCTION

In recent years, advancements in natural language processing (NLP) have been predominantly driven by large pre-trained language models (LLMs). These models, such as BERT and GPT-3, have demonstrated remarkable capabilities in understanding and generating human-like text. However, their effectiveness often relies on extensive training datasets, which can be a challenge in low-resource languages.

Low-resource languages, characterized by limited amounts of labeled data, present a significant obstacle in fine-tuning LLMs for specific tasks. This scarcity hampers the model's ability to grasp the nuances and intricacies of these languages, hindering its performance on downstream applications.

### 1.1 Motivation

The motivation behind this research lies in the imperative need to enhance the performance of LLMs in low-resource language scenarios. Traditional approaches to fine-tuning involve utilizing the available labeled data, but in cases where the dataset is meager, alternative strategies are necessary. Data augmentation, a technique widely employed in computer vision, has shown

promise in artificially expanding datasets and improving model generalization. However, its application to language data, especially in low-resource contexts, remains an underexplored area.

This research aims to bridge this gap by investigating various data augmentation strategies tailored for labeled data enrichment in low-resource language fine-tuning of LLMs. By doing so, we strive to unlock the latent potential of these models and empower them to better understand and generate high-quality content in languages with limited annotated resources.

### 1.2 Objectives

The primary objectives of this research can be outlined as follows:

1. Investigate and analyze the applicability of existing data augmentation techniques in the context of low-resource languages.
2. Tailor data augmentation strategies to the linguistic characteristics and idiosyncrasies of low-resource languages, ensuring the generated data aligns with the natural language patterns.
3. Systematically evaluate the impact of the proposed data augmentation strategies on the performance of fine-tuned LLMs in low-resource language tasks. This includes assessing improvements in accuracy, robustness, and generalization.
4. Conduct comparative analyses against baseline models that rely solely on the limited labeled data, highlighting the effectiveness and efficiency of the proposed data augmentation techniques.
5. Contribute insights and best practices for data augmentation in the context of low-resource languages, providing guidance for future research endeavors in similar domains.

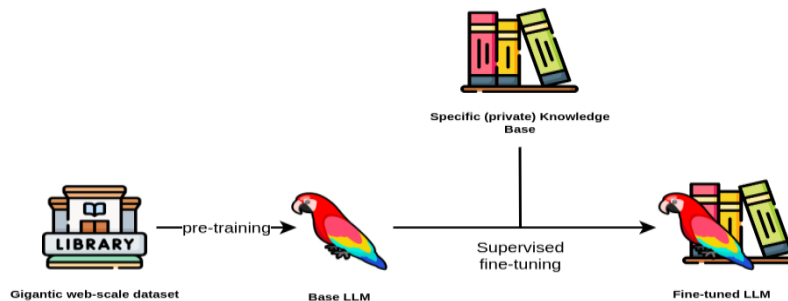


Figure 1: Data Augmentation via Fine-Tuning

## 2. LITERATURE REVIEW

### 2.1 Challenges in Low-Resource Language Fine-Tuning:

Low-resource languages present unique challenges in fine-tuning language models (LLMs). Limited amounts of labeled data hinder the performance of models trained on mainstream languages when applied to these less-represented languages. Challenges include the scarcity of annotated datasets, domain-specific vocabulary, and the need for specialized pre-trained embeddings. Addressing these challenges is crucial for making natural language processing (NLP) applications more inclusive and effective across diverse linguistic contexts.

### 2.2 Importance of Data Augmentation in NLP:

Data augmentation is a key strategy to overcome data scarcity in NLP tasks, especially in low-resource language scenarios. By artificially increasing the diversity and volume of training data, data augmentation helps LLMs generalize better to various linguistic nuances and domains. This is essential for improving the robustness and performance of models when applied to real-world applications in low-resource language settings.

### 2.3 Existing Data Augmentation Techniques for Language Models:

Several data augmentation techniques have been proposed and applied successfully in the context of language models. These include traditional techniques such as synonym replacement, word deletion, and insertion. More advanced methods involve leveraging pre-trained contextual embeddings, such as BERT or GPT, to generate semantically coherent augmented examples. These techniques aim to enhance the model's ability to handle diverse linguistic patterns and effectively transfer knowledge from resource-rich to low-resource languages.

### 2.4 Gap Identification:

Despite the advancements in data augmentation techniques, there exist gaps in current strategies, particularly concerning the enrichment of labeled data in low-resource language fine-tuning. Identifying and addressing these gaps are crucial for achieving optimal performance in NLP tasks for underrepresented languages. Common gaps may include the lack of diversity in augmented examples, challenges in maintaining semantic coherence, and issues related to the transferability of knowledge from resource-rich languages to low-resource languages.

## 3. METHODS FOR DATA AUGMENTATION

Data augmentation strategies are techniques used to artificially increase the size of a training dataset by creating modified or new instances of the existing data. This is particularly useful when dealing with limited labeled data.

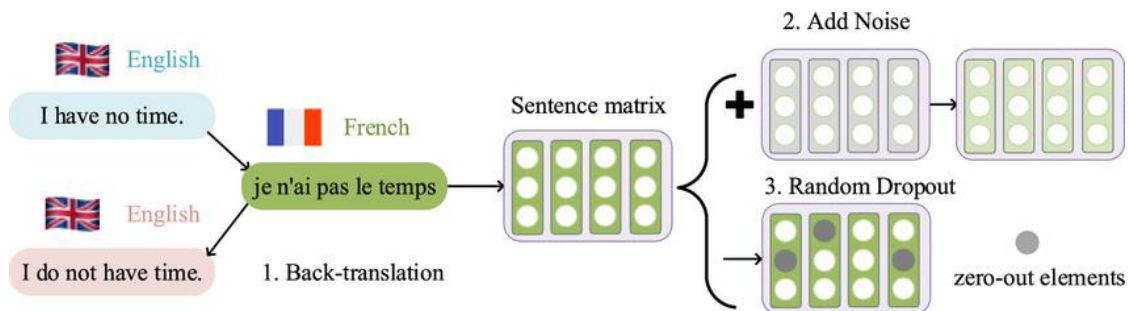


Figure 2 : Data Augmentation Methods

Here are the data augmentation strategies mentioned in your context:

### 3.1 Synonym Replacement

This strategy involves replacing words in the text with their synonyms while maintaining the overall context. Synonym replacement can be performed using pre-built synonym dictionaries or word embeddings. It helps in diversifying the vocabulary of the dataset.

### 3.2 Back-Translation

Back-translation involves translating sentences from the target language to a different language and then back to the original language. This introduces linguistic variations and helps generate new instances of the text. Commonly used languages for back-translation include English, as it often has well-established translation models.

### 3.3 Sentence Splitting and Shuffling

Sentence splitting involves breaking down sentences into smaller fragments, and shuffling rearranges these fragments to form new sentences. This strategy introduces variability in sentence structures and order, contributing to the enrichment of the dataset.

### 3.4 Morphological Variations

Morphological variations involve altering the morphological structure of words in the text. This can include changes in verb tense, noun forms, or other morphological features. Introducing such variations helps the model generalize better to different forms of words.

#### Additional Considerations:

- 1) **Parameter Tuning:** Specify any parameters or thresholds used in these augmentation strategies.
- 2) **Balance:** Ensure that the augmented dataset maintains a balance in the distribution of classes to prevent biases in model learning.
- 3) **Tool and Libraries:** If specific tools or libraries were used for these augmentations, mention them. For example, NLTK, SpaCy, or specific translation services for back-translation.
- 4) **Evaluation:** Consider adding a brief discussion on how the effectiveness of these augmentation strategies was measured or validated, and whether there were any observed challenges or limitations.

### 4. FINE-TUNING PROCEDURE

Remember to adapt and tailor these strategies based on the characteristics of your low-resource language and the specific goals of your fine-tuning task. Additionally, provide sufficient details to enable reproducibility and transparency in your research. In this section, you describe the step-by-step process involved in fine-tuning Language Models (LLMs) on your dataset. Here is a template to help structure this section:

- 1) **Choice of Pre-trained Model:** Specify the pre-trained language model architecture chosen for fine-tuning. This could be a widely used model like BERT, GPT, RoBERTa, etc.
- 2) **Dataset Preparation:** Outline how the dataset was prepared for fine-tuning. This includes any preprocessing steps such as tokenization, handling missing values, and addressing class imbalances.
- 3) **Fine-tuning Objective:** Clearly state the objective of fine-tuning. For example, if you're fine-tuning for a specific downstream task like sentiment analysis or named entity recognition, provide details.
- 4) **Hyperparameter Selection:** Specify the hyperparameters chosen for fine-tuning. This includes learning rate, batch size, the number of training epochs, etc. Discuss any rationale behind the choice of these hyperparameters.
- 5) **Model Modification (if any):** Mention any modifications or adjustments made to the pre-trained model architecture for the specific needs of your task. This might include adding task-specific layers, adjusting hidden layer sizes, or using domain-specific embeddings.
- 6) **Training Process:** Describe the training process, including how the model was initialized, the optimization algorithm used (e.g., Adam), and any regularization techniques applied.
- 7) **Validation and Early Stopping:** Explain how model performance was monitored during training. Describe the criteria used for early stopping to prevent overfitting.
- 8) **Computational Resources:** Specify the computational resources used for fine-tuning, including GPU specifications or any distributed training setups.
- 9) **Fine-tuning Tools and Frameworks:** Mention the tools and frameworks used for fine-tuning. Common choices include TensorFlow, PyTorch, or Hugging Face Transformers library.
- 10) **Handling Low-Resource Challenges:** Discuss any specific challenges related to working with low-resource languages and how these challenges were addressed during fine-tuning.

- 11) **Model Evaluation:** Briefly explain how the fine-tuned model was evaluated. Specify the evaluation metrics used to assess its performance on the validation or test set.
- 12) **Results Analysis:** If available, provide a brief analysis of the results, highlighting any interesting findings or challenges encountered during the fine-tuning process.

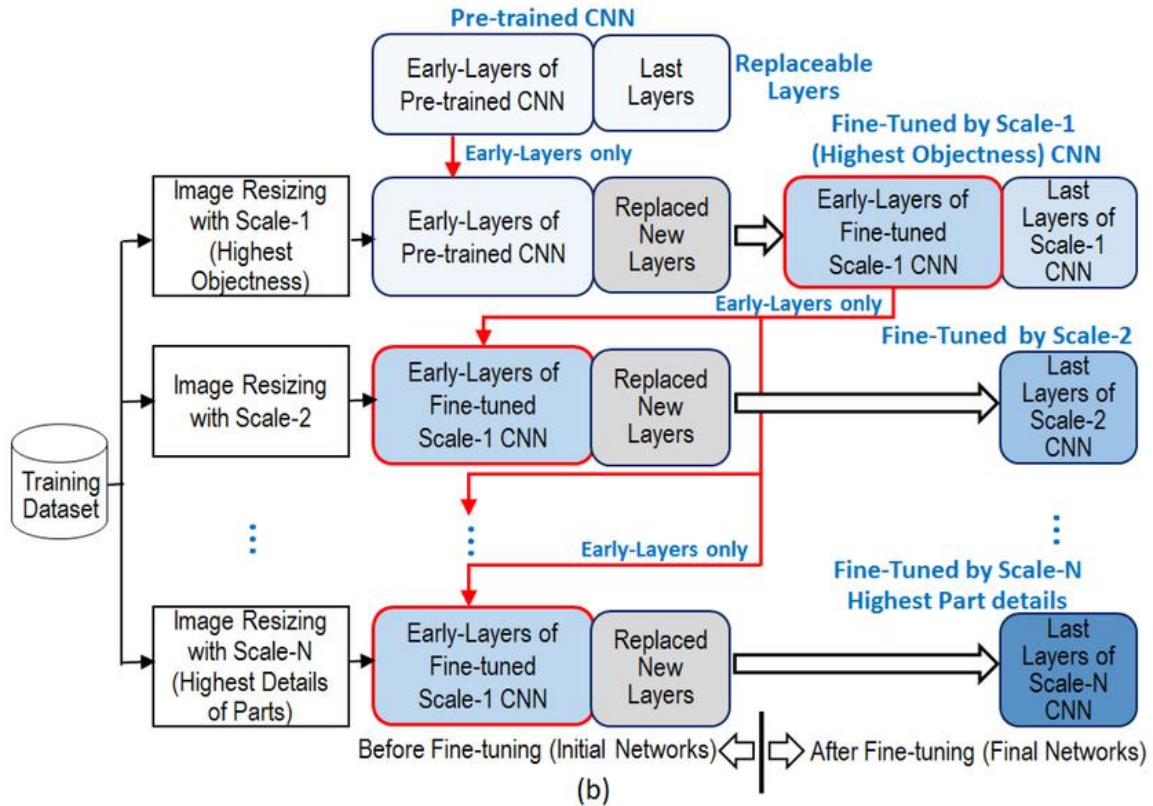


Figure 3 : The sequential fine-tuning process

## 5. METHODOLOGY:

### 5.1 Problem Definition

Consider a task-specific and labeled dataset  $(D_T = \{(x_i, y_i)\}_{i=1}^N)$  comprising text input-output pairs for a given task  $(T)$ . In this context, we have a teacher Language Model (LLM) parameterized by  $(\theta_T)$  and a lightweight student model  $(\theta_S)$  with  $(|\theta_S| \ll |\theta_T|)$ . The objective is to achieve a performance level for  $(\theta_S)$  that is comparable to  $(\theta_T)$  in solving task  $(T)$ . Notably,  $(\theta_T)$  outperforms  $(\theta_S)$  significantly but is more resource-intensive for deployment.

Our approach involves generating synthetic data  $(D_{eT})$  using  $(\theta_T)$  and subsequently fine-tuning  $(\theta_S)$  on the combined dataset  $(D_T \cup D_{eT})$ .

Furthermore, we distinguish between two cases: (i) annotating an unlabeled set of instances  $(\{x_i\}_{i=1}^M)$  such that  $(D_{eT} = \{(x_i, \hat{y}_i)\}_{i=1}^M)$ , or (ii) generating entire input-output pairs  $(D_{eT} = \{(\hat{x}_i, \hat{y}_i)\}_{i=1}^M)$ , where  $(\hat{x}_i)$  and  $(\hat{y}_i)$  denote synthetic input and output sequences, respectively.

Case (i) is applicable in scenarios where an abundance of unlabeled text is available (e.g., scraped from the internet), and acquiring annotations is costly. On the other hand, case (ii) is suitable even in situations where unlabeled instances are not available.

## 5.2 Teacher model

The teacher Language Model (LLM) employed in our study is the 20B GPT-NeoX, which, during our experimental phase, was recognized as state-of-the-art. We adopt the text-to-text framework, where the LLM takes a text query as input and generates text output. For example, in generating dialog responses, the query represents incomplete dialog utterances, and the model produces the response. In classification tasks, the query corresponds to textual input, and the model generates the class label.

In our fine-tuning process, GPT-NeoX is adapted for multi-task instruction-following using various task mixtures from the Super-Natural Instructions dataset. Each prompt consists of a task description prefix and a small subset of training input-output pairs, delineated by [INPUT] and [OUTPUT] tags. The prompt concludes with either [OUTPUT] or [INPUT] based on whether we annotate unlabeled text or generate entirely new examples.

All layers of GPT-NeoX are fine-tuned using the Adam optimizer for up to 5 epochs, employing a cosine schedule with a learning rate of  $1e - 5$ . Our implementation relies on the GPT-NeoX library, which integrates DeepSpeed for distributed training. The experiments are conducted using NVIDIA A100 GPUs from an internal compute cluster.

A crucial hyper-parameter during inference is the generation temperature, influencing the randomness of LLM output. We observe that a low temperature of 0.1 is optimal for annotations, while a high temperature of 0.8 yields the best results for generations due to increased diversity in the samples.

## 5.3 Student models

For tasks involving text classification, we employ the RoBERTa-Large model, and for text generation tasks, we use the BART-Large model. The publicly available checkpoints of these models are fine-tuned for our specific purposes. In the case of text classification tasks, fine-tuning is carried out for up to 320 epochs using the Adam optimizer, a batch size of 50, and a learning rate of  $1e - 5$ . On the other hand, for text generation tasks, fine-tuning is performed for 5 epochs using the Adam optimizer, a linear learning schedule, a batch size of 32, and a learning rate of  $5e - 5$ . The reported results are based on the model with the best validation loss.

# 6. EXPERIMENTAL RESULTS:

## 6.1 Datasets

**Table 1:** Dataset Statistics

Dataset	Examples		
	Train	Dev	Test
SLURP	11514	2033	2974
RTE	2500	277	300
BoolQ	9427	3270	3245
MultiRC	5100	953	1800
PubMedQA	212300	-	1000
SGD	164982	10000	10000
WebNLG	35426	1667	1779

Table 1 outlines the sizes of the datasets. To calculate the number of fine-tuning samples for the teacher Language Model (LLM), the fractions presented in subsequent tables can be applied to this initial dataset.

### 6.1.1 Text Classification

In the realm of text classification, we present classification accuracies for four distinct tasks. The SLURP (Spoken Language Understanding Resource Package) dataset encompasses multi domain interactions for spoken language understanding, capturing single-turn user interactions with a home assistant. BoolQ comprises naturally occurring yes/no questions pertaining to a provided paragraph. MultiRC (Multi Sentence Reading Comprehension) is a multi-sentence question-answering dataset, challenging models to combine information from multiple sentences.

Dataset	Type	Data amount in %		Dev Acc	Test Acc
		Original	DAFTT		
SLURP	X, Y Y  X	1%	0 %	42.25	43.95
			26 %	54.57	54.25
			78 %	76.14	76.09
	X, Y Y  X	5%	0 %	73.49	71.59
			43 %	77.39	76.89
			78 %	85.00	83.96
	- X, Y Y  X	10%	0%	80.12	80.04
			43 %	82.59	81.91
			43 %	86.13	86.48
-	-	100 %	0 %	88.64	87.70
BoolQ	X, Y Y  X	1%	0 %	62.84	N/A
			31 %	68.96	
			44 %	79.72	
	X, Y Y  X	5%	0 %	62.97	
			31 %	66.02	
			44 %	80.09	
	- X, Y Y  X	10%	0 %	68.29	
			31 %	77.22	
			44 %	81.93	
-	-	100 %	0 %	85.05	
RTE	X, Y Y  X	5%	0 %	60.65	N/A
			80 %	66.79	
			80 %	83.20	
	X, Y Y  X	10%	0 %	65.43	
			80 %	69.68	
			80 %	83.75	
	- X, Y Y  X	20%	0 %	74.73	
			80 %	76.84	
			80 %	85.20	
-	-	100 %	0 %	86.60	

MultiRC	X, Y Y  X	1%	0 %	57.50	8.18
			40 %	63.40	15.11
			254 %	71.46	24.24
	X, Y Y  X	5%	0 %	67.70	18.05
			40 %	72.85	21.86
			254 %	71.51	32.63
	- X, Y Y  X	10%	0 %	70.63	22.35
			40 %	73.88	24.97
			254 %	76.90	34.94
-	100%	0 %	82.12	48.16	

### 6.1.2 Natural Language Generation

For natural language generation tasks, we present Rouge- $\{1,2,L\}$  scores on both development and test sets for two specific tasks. The Schema Guided Dialog (SGD) dataset features task-oriented conversations between a human and a virtual assistant. The WebNLG dataset is dedicated to the data-to-text task, focusing on generating grammatically correct text that verbalizes input triples.

Dataset	Type	Data amount in %		Validation scores			Test scores			
		Original	DAFTT	R-1	R-2	R-L	R-1	R-2	R-L	
SGD	- X, Y Y  X	1%	0 %	21.78	9.29	20.09	21.91	9.40	20.06	
			10%	43.72	25.09	39.84	40.82	22.36	37.00	
				47.44	28.19	43.47	43.37	24.49	39.58	
	- X, Y Y  X	5%	0 %	10%	33.17	16.86	30.32	30.92	14.86	28.09
					45.27	26.39	41.48	41.71	23.25	37.98
					48.99	29.87	45.02	44.60	25.76	40.66
	- X, Y Y  X	10%	0 %	10%	35.48	18.48	32.47	33.40	16.40	30.37
					45.59	30.35	42.62	43.55	24.39	39.15
					48.32	29.14	44.33	43.87	24.69	39.85
	-	100%	0 %		56.61	38.63	52.64	49.61	32.04	45.29

### 6.2 Key Findings

In this segment, we outline the main findings of our study. We calculate data quantities in relation to the size of the original training dataset. For instance, a reference such as "1% original and 10% synthetic data" signifies the creation of the expanded training dataset by combining 1% of data randomly selected from the original training dataset with 10% synthetic data.

Our findings suggest that both data annotations and generations contribute significantly to improvements in performance across all evaluation metrics. Notably, these advantages become more evident as the initial training dataset size decreases.



This observation is intriguing because it implies that a relatively small number of samples are adequate for the successful fine-tuning phase of the teacher model.

### 6.3 Variation in Synthetic Data Amounts

To comprehend the impact of varying the quantity of synthetic data on downstream performance, we explore different augmentation levels for Natural Language Generation (NLG) tasks. Figure 5 illustrates the outcomes for various augmentation amounts when the original training data constitutes 1%. Notably, for Schema Guided Dialog (SGD), we observe performance improvements up to 30% synthetic data. However, for WebNLG, we notice diminishing returns and, in some instances, slightly inferior performance with increasing synthetic data.

### 6.4 Direct Evaluation of Teacher LLMs

The performances of the teacher models, assessed without generating additional data points or fine-tuning a student model, are presented in Figure 8. Intriguingly, the fine-tuned NeoX 20B model surpasses the performance of the fine-tuned davinci-002 model with 175B parameters, even when fine-tuned with default hyperparameters from OpenAI's API.

### 6.5 Comparison Between Teacher LLMs NeoX-20B and GPT-3.5 175B

We explore the effectiveness of the DAFTT framework when paired with different teacher models, comparing NeoX-20B with OpenAI's GPT-3.5 175B (referred to as davinci-002 in their API). Figure 4 illustrates the performance comparison with and without DAFTT. Notably, DAFTT proves most effective in scenarios with minimal initial data, even when the teacher LLM is fine-tuned on that small dataset.

	Type	Data amount		Rouge-L	
		Original	DAFTT	Dev	Test
Dataset		1%	0%	20.09	20.06
	Y  X	1%	10%	43.47	39.58
			20%	45.22	40.89
			30%	46.02	41.29
		1%	10%	39.84	37.00
	X, Y		20%	40.92	37.62
			30%	42.27	39.30
		100%	0%	53.63	46.30
WebNLG		1%	1%	57.59	52.92
	Y  X		2%	58.94	54.50
			3%	59.44	54.44
			4%	59.79	54.42
			5%		
			0%	59.56	54.44
		1%	1%	57.09	52.81
	X, Y		2%	57.00	52.29
			3%	57.08	52.64
			4%	56.92	52.72
			5%	56.92	52.76
		100%	0%	65.73	55.76

Figure 5 displays varying amounts of DAFTT data added to the student model training set. While both SGD and WebNLG tasks benefit from synthetic data, there is an observation of diminishing returns.

Dataset	Type	Data amount		Rouge-L	
		Original	DAFTT	Dev	Test
SGD	-	1%	0%	20.09	20.06
	Y  X	1%	10%	43.47	39.58
	X, Y	1%	10%	39.84	37.00
	Y X; X,Y	1%	5% each	42.46	39.18
WebNLG	-	1%	0%	42.58	39.72
	Y  X	1%	10%	59.60	54.51
	X, Y	1%	1%	56.06	52.98
	Y X; X,Y	1%	5% each	59.41	54.17

Figure 6 compares the combination of Annotation (Y|X) and Generation (X, Y) in equal proportions, performing almost as well as annotating the same amount. The latter assumes access to unlabeled instances, which can be challenging in practical scenarios.

Model	Dev			Test		
	Rouge 1	Rouge 2	Rouge L	Rouge 1	Rouge 2	Rouge L
davinci-002	35.47	17.99	32.35	34.62	17.19	31.46
NeoX	47.44	28.19	43.47	43.37	24.49	39.59

Figure 7 evaluates the downstream performance of the student LM after being fine-tuned on 11% of the original training dataset size, comprising 1% original data and 10% teacher-annotated examples for the SGD task.

Model	Dev			Test		
	Rouge 1	Rouge 2	Rouge L	Rouge 1	Rouge 2	Rouge L
davinci-002	30.71	14.38	28.06	30.54	14.14	27.70
NeoX	38.38	20.84	35.44	37.78	20.13	34.58

Figure 8 showcases the evaluation of the Teacher LLMs on SGD after fine-tuning with 1% training data. The fine-tuned NeoX model consistently demonstrates superior performance on the test sets across tasks, despite differences in model size.

### 6.6 Integration of Annotation and Generation

As discussed in Section 3.2, our observations reveal that annotating existing unlabeled data leads to more substantial performance gains than generating entire input-output pairs. However, in many real-world scenarios, access to unlabeled data may still be more expensive than generating data points entirely from scratch. Therefore, we explore whether combining generated data points with annotated ones in equal proportions can achieve performances closer to a larger set of annotated ones while potentially being more cost-effective in practice. Confirming this hypothesis in the context of 1% original training data, as depicted in Figure 6, we find that when mixing both data sources with 5% each, the performance is nearly equivalent to annotating 10% of the entire dataset.

## 6.7 Qualitative Examination of Generated Text

In Appendix A, we present both annotated instances and entirely generated ones. Through manual inspection and verification, we ascertain that all generated examples are coherent and syntactically correct. However, it's noted that not all instances are factually accurate, a phenomenon often referred to as "hallucination." A potential avenue for future work could involve implementing a post-hoc factuality filtering process for the generated examples.

## 7. RELATED WORK

Several studies explore the realm of data augmentation for textual data by leveraging large teacher models. However, many of these endeavors do not involve the fine-tuning of the teacher Language Model (LLM) on limited data to improve the quality of the generated data. For example, Efrat & Levy investigate a model's ability to follow natural language instructions, including annotating examples from unlabeled datasets. Schick & Schütze focus on synthetic data generation in semantic textual similarity datasets without task-specific training examples. Other works concentrate on tasks such as information retrieval, code generation, and reasoning.

Yoo et al. propose knowledge transfer from LLMs to student models through the generation of synthetic examples and the use of knowledge distillation with soft labels. Wang et al. explore the cost-effective use of GPT-3 as a data labeler to train other models. They find that, for Natural Language Generation (NLG) and Natural Language Understanding (NLU) tasks, using GPT-3 generated labels costs 50% to 96% less than human annotations. Similarly, Ding et al. assess GPT-3's effectiveness as a data annotator on classification and named entity recognition tasks.

Gunasekar et al. and Li et al. follow an approach akin to ours, augmenting pretraining data with synthetically generated documents, such as textbooks and exercises, to train a comparatively small model that outperforms larger ones on coding tasks. However, our focus is specifically on the fine-tuning phase, addressing classification and natural language generation tasks.

In essence, our approach shares similarities with knowledge distillation (KD), where the output logits of the teacher model serve as targets during student model training. However, state-of-the-art LLMs are often accessed through cloud-based commercial APIs, revealing only a truncated output distribution (e.g., top-5 tokens). In contrast, our synthetic data generation approach does not require access to the complete output distribution of the teacher.

## 8. CONCLUSION

This study has showcased that fine-tuning teacher Language Models (LLMs) for both annotating unlabeled instances and generating new data points can effectively enhance the performance of downstream models. Our empirical investigations spanned six tasks, including four in classification and two related to natural language generation. An inherent limitation of our approach lies in the substantial resource requirements associated with fine-tuning a large model. In future endeavors, we aim to further quantify the extent of fine-tuning needed to guide the teacher model towards generating high-quality synthetic data.

## REFERENCES

- [1]. Open LLM Leaderboard - a Hugging Face Space by HuggingFaceH4. URL [https://huggingface.co/spaces/HuggingFaceH4/open\\_llm\\_leaderboard](https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard).
- [2]. Andonian, A., Biderman, S., Black, S., Gali, P., Gao, L., Hallahan, E., Levy-Kramer, J., Leahy, C., Nestler, L., Parker, K., Pieler, M., Purohit, S., Songz, T., Phil, W., and Weinbach, S. GPT-NeoX: Large Scale Autoregressive Language Modeling in PyTorch, August 2021. URL <https://www.github.com/eleutherai/gpt-neox>.
- [3]. Azerbayev, Z., Ni, A., Schoelkopf, H., and Radev, D. Explicit knowledge transfer for weakly supervised code generation, 2022. URL <https://arxiv.org/abs/2211.16740>.
- [4]. Bastianelli, E., Vanzo, A., Swietojanski, P., and Rieser, V. Slurp: A spoken language understanding resource package, 2020. URL <https://arxiv.org/abs/2011.13205>.

- [5]. Bayer, M., Kaufhold, M.-A., and Reuter, C. A survey on data augmentation for text classification. *ACM Computing Surveys*, 55(7):1–39, 2022.
- [6]. Black, S., Biderman, S., Hallahan, E., Anthony, Q., Gao, L., Golding, L., He, H., Leahy, C., McDonnell, K., Phang, J., et al. Gpt-neox-20b: An open-source autoregressive language model. *arXiv preprint arXiv:2204.06745*, 2022.
- [7]. Bonifacio, L., Abonizio, H., Fadaee, M., and Nogueira, R. Inpars: Data augmentation for information retrieval using large language models, 2022. URL <https://arxiv.org/abs/2202.05144>.
- [8]. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [9]. Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.
- [10]. Dai, A. M. and Le, Q. V. Semi-supervised sequence learning, 2015. URL <https://arxiv.org/abs/1511.01432>.
- [11]. Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018. URL <https://arxiv.org/abs/1810.04805>.
- [12]. Ding, B., Qin, C., Liu, L., Bing, L., Joty, S., and Li, B. Is gpt-3 a good data annotator?, 2022. URL <https://arxiv.org/abs/2212.10450>.
- [13]. Efrat, A. and Levy, O. The turking test: Can language models understand instructions? *ArXiv preprint arXiv:2010.11982*, 2020.
- [14]. Gardent, C., Shimorina, A., Narayan, S., and Perez-Beltrachini, L. Creating training corpora for nlg micro-planning. In *55th annual meeting of the Association for Computational Linguistics (ACL)*, 2017.
- [15]. Gehrmann, S., Adewumi, T., Aggarwal, K., Ammanamanchi, P. S., Anuluwapo, A., Bosselut, A., Chandu, K. R., Clinciu, M., Das, D., Dhole, K. D., et al. The gem benchmark: Natural language generation, its evaluation and metrics. *arXiv preprint arXiv:2102.01672*, 2021.
- [16]. Gunasekar, S., Zhang, Y., Aneja, J., Mendes, C. C. T., Del Giorno, A., Gopi, S., Javaheripi, M., Kauffmann, P., de Rosa, G., Saarikivi, O., et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.
- [17]. Hinton, G., Vinyals, O., Dean, J., et al. Distilling the knowledge in a neural network. *ArXiv preprint arXiv:1503.02531*, 2(7), 2015.
- [18]. Ho, N., Schmid, L., and Yun, S.-Y. Large language models are reasoning teachers. *ArXiv preprint arXiv:2212.10071*, 2022.
- [19]. Jin, Q., Dhingra, B., Liu, Z., Cohen, W., and Lu, X. PubMedQA: A dataset for biomedical research question answering. 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)*, pp. 2567–2577, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1259. URL <https://aclanthology.org/D19-1259>.
- [20]. Kaddour, J., Harris, J., Mozes, M., Bradley, H., Raileanu, R., and McHardy, R. Challenges and Applications of Large Language Models, July 2023. URL <http://arxiv.org/abs/2307.10169>. arXiv:2307.10169 [cs].
- [21]. Khashabi, D., Chaturvedi, S., Roth, M., Upadhyay, S., and Roth, D. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of*

the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 252–262, 2018.

[22]. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461, 2019.

[23]. Li, Y., Bubeck, S., Eldan, R., Del Giorno, A., Gunasekar, S., and Lee, Y. T. Textbooks are all you need ii: phi-1.5 technical report. arXiv preprint arXiv:2309.05463, 2023.

[24]. Lin, C.-Y. Rouge: A package for automatic evaluation of summaries. In Text summarization branches out, pp. 74–81, 2004.

[25]. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692, 2019.

[26]. Eng, Y., Huang, J., Zhang, Y., and Han, J. Generating training data with language models: Towards zero-shot language understanding. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), Advances in Neural Information Processing Systems, 2022. URL [https://openreview.net/forum?id=4G1Sfp\\_1sz7](https://openreview.net/forum?id=4G1Sfp_1sz7).



[27]. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P. J., et al. Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res., 21(140):1–67, 2020.

[28]. Rasley, J., Rajbhandari, S., Ruwase, O., and He, Y. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 3505–3506, 2020.

[29]. Rastogi, A., Zang, X., Sunkara, S., Gupta, R., and Khaitan, P. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset, 2019. URL <https://arxiv.org/abs/1909.05855>.

[30]. Schick, T. and Schütze, H. Generating datasets with pretrained language models, 2021. URL <https://arxiv.org/abs/2104.07540>.

**BIOGRAPHIES**

	<p><b>Pettugani Hotragn</b>                  Education Details: 10th standard (8.8/10.0), Intermediate (74.9%), B. Tech (till 6th semester (3.27/4)                  Any Achievements- Participated in hackathon, competitions and internship                  Current pursuing Education - B tech (4th year (7th sem)                  University : woxsen University, University in Telangana, India</p>
	<p><b>T.kinshuk sunder reddy</b>                  Education Details: 10th standard (70%), Intermediate (74.5%), B. Tech (till 6th semester (2.25/4)).                  Any Achievements- Participated in hackathon and competitions.                  Current pursuing Education - B tech (4th year (7th sem)                  University - woxsen University, University in Telangana, India</p>



**Kakarla Mohan Kumar**

Education Details: 10th standard (9.8/10.0), Intermediate (84.5%), B. Tech (till 6th semester (2.988/4)

Any Achievements- Participated in hackathon, competitions and internship

University - Satya Institute of Technology and management, Andhra Pradesh, india

Current pursuing Education - B tech (4th year (7th sem)