Instruction Tuned Models Are Quick Learners with Instruction Equipped Data on Downstream Tasks

by

Himanshu Gupta

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved November 2023 by the
Graduate Supervisory Committee:

Chitta Baral, Chair
Nakul Gopalan
Arindam Mitra

ARIZONA STATE UNIVERSITY

December 2023

ABSTRACT

Instruction tuning of language models has demonstrated the ability to enhance model generalization to unseen tasks via in-context learning using a few examples. However, typical supervised learning still requires a plethora of training data for downstream or “Held-in” tasks. Often in real-world situations, there is a scarcity of data available for finetuning, falling somewhere between few shot inference and fully supervised finetuning. In this work, I demonstrate the sample efficiency of instruction tuned models over various tasks by estimating the minimal training data required by downstream or “Held-In” tasks to perform transfer learning and match the performance of state-of-the-art (SOTA) supervised models. I conduct experiments on 119 tasks from Super Natural Instructions (SuperNI) in both the single task learning / Expert Modelling (STL) and multi task learning (MTL) settings. My findings reveal that, in the STL setting, instruction tuned models equipped with 25% of the downstream train data surpass the SOTA performance on the downstream tasks. In the MTL setting, an instruction tuned model trained on only 6% of downstream training data achieve SOTA, while using 100% of the training data results in a 3.69% points improvement (ROUGE-L 74.68) over the previous SOTA. I conduct an analysis on T5 vs Tk-Instruct by developing several baselines to demonstrate that instruction tuning aids in increasing both sample efficiency and transfer learning. Additionally, I observe a consistent $\sim 4\%$ performance increase in both settings when pre-finetuning is performed with instructions. Finally, I conduct a categorical study and find that contrary to previous results, tasks in the question rewriting and title generation categories suffer from instruction tuning.

DEDICATION

Dedicated to, my loving parents, family and friends for the love, patience, and faith in this short journey and in much more to come...

ACKNOWLEDGEMENTS

I would like to extend my sincere gratitude to my thesis supervisor, Dr. Chitta Baral, for his invaluable guidance. He has supported me in exploring various research problems, experimenting with new ideas, and collaborating with a diverse range of individuals. His unwavering encouragement has enabled me to pursue my research visions with confidence and enthusiasm. I am also thankful to my thesis committee members, Dr. Nakul Gopalan and Dr. Arindam Mitra, for their expert guidance and insightful suggestions throughout my research journey. Furthermore, I wish to express my appreciation to my mentor, Dr. Swaroop Mishra for his valuable feedback, encouragement, and continuous support. I would like to acknowledge the financial assistance provided by ASU and DARPA, as well as the resources made available by ASU HPC and CogInt Lab (Kingkong 1, 2, Godzilla 1, 2). These resources facilitated the execution of my research and the successful completion of this thesis.

My heartfelt thanks go to my roommates, Saurabh Arjun Sawant, Kevin Joseph Scaria, and Siddharth Goyal, for their unwavering support. I am fortunate to have a wonderful circle of friends, including Abhiram Gulanikar, Shreyas Verma, Ujjwala Anantheswaran, Jayant Sravan, Siva Priya Bollineni, and Yamini Nagireddy. I also extend my gratitude to my colleagues Mihir Parmar, Dr. Kuntal Pal, Maitreya Patel, Neeraj Varshney, Dr. Man Luo, and other members of the Cognition and Intelligence Lab, whose support and confidence in me propelled my research forward.

Lastly, I want to express deep love and gratitude to my parents, brother, and all my family members for their unwavering support and encouragement. Their love provided the foundation that sustained me throughout this journey. To everyone mentioned above, and to all those who have offered unconditional love and support, I extend my heartfelt thanks.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

INTRODUCTION

Large language models (LLM) have achieved remarkable performances on several benchmark evaluation suites such as SuperGLUE Wang *et al.* (2019), BIG-Bench Hard (BBH) Suzgun *et al.* (2022), and HELM Liang *et al.* (2022). Research on LLMs has explored their abilities to follow instructions Wei *et al.* (2021); Mishra *et al.* (2022b); Wang *et al.* (2022c) and has developed specialized models for the same (Flan, Instruct-GPT, Tk-Instruct, T0) Ouyang *et al.* (2022); Sanh *et al.* (2021).

1.1 Evolution of Instruction Tuning

Instruction tuning has emerged as a promising approach to enhance the capabilities of large language models (LLMs). Unlike traditional fine-tuning methods that rely on labeled input-output pairs, instruction tuning utilizes natural language instructions to guide LLMs towards desired behaviors Mishra *et al.* (2022b); Wei *et al.* (2021); Sanh *et al.* (2021).

Currently, instruction tuning is defined as follows Zhang *et al.* (2023f): *"Instruction tuning refers to the process of further training LLMs on a dataset consisting of (PROMPT, RESPONSE) pairs in a supervised fashion, which bridges the gap between the next-word prediction objective of LLMs and the users' objective of having LLMs adhere to human instructions."*

Instruction tuning has revolutionized LLM training, enabling them to tackle a broader spectrum of tasks and generalizing more effectively to new situations Ye *et al.* (2021); Khashabi *et al.* (2020). Early instruction tuning methods focused on providing explicit instructions that closely resembled the desired outputs. Flan paper

Wei *et al.* (2021) used the following example for instruction tuning:

"***Instruction***: *Here is a goal: Get a cool sleep on summer days. How would you accomplish this goal?*

Input: *Keep stack of pillow cases in fridge or Keep stack of pillow cases in oven.*

Output: *keep stack of pillow cases in fridge*"

Cross task generalization paper Mishra *et al.* (2022b) defined instruction as instruction to solve a specific task. To quote their work, *"crowdsourcing instructions often elaborate a variety of details about how a task should (and should not) be done"* An example is given below to describe their definition:

*"**Title:** Writing questions that involve commonsense understanding of event duration.*

***Definition:** In this task, we ask you to write a question that involves event duration, based on a given sentence. Here, event duration is defined as the understanding of how long events typically last. For example,"brushing teeth", usually takes few minutes.*

***Emphasis & Caution**: The written questions are not required to have a single correct answer.*

***Things to avoid:** Don't create questions which have explicit mentions of answers in text. Instead, it has to be implied from what is given. In other words, we want you to use "instinct" or "common sense".*

Positive Example

Input: Sentence: Jack played basketball after school, after which he was very tired.

Output: How long did Jack play basketball?

Reason: the question asks about the duration of an event; therefore it's a temporal event duration question.

Negative Example

Input: Sentence: He spent two hours on his homework.

Output: How long did he do his homework?

Reason: We DONOT want this question as the answer is directly mentioned in the text.

***Input:** Sentence: It's hail crackled across the comm, and Tara spun to retake her seat at the helm.*

***Output:** How long was the storm?"*

To point out the differences between the two prompts, Flan paper had Input and

Options as the Prompt and Output as the response, while Cross-Task paper had Title, Definition, Emphasis and Caution, Things to Avoid, Positive, negative examples with explanations as the Prompt.

However, recent advances have introduced more sophisticated techniques that leverage natural language instructions to convey task-specific knowledge and constraints (Iyer *et al.*, 2022; Muennighoff *et al.*, 2022; Chung *et al.*, 2022). These methods, such as the use of prompts, examples, and constraints, have significantly broadened the scope of instruction tuning, allowing LLMs to perform complex tasks that require reasoning, planning, and common-sense knowledge. To give an example from the SuperNI paper Wang *et al.* (2022c), Instruction tuning was changed as follows:

*"**Definition:** Given an utterance and recent dialogue context containing past 3 utterances (wherever available), output Yes if the utterance contains the small-talk strategy, otherwise output No. Small-talk is a cooperative negotiation strategy. It is used for discussing topics apart from the negotiation, to build a rapport with the opponent.*

Positive Example

Input: Context: That's fantastic, I'm glad we came to something we both agree with.

Utterance: Me too. I hope you have a wonderful camping trip.

Output: Yes

Explanation: The participant engages in small talk when wishing their opponent to have a wonderful trip.

Negative Example

Input: Context: Sounds good, I need food the most, what is your most needed item?!

Utterance: My item is food too.

Output: Yes

Explanation: The utterance only takes the negotiation forward and there is no side talk. Hence, the correct answer is No.

***Input:** Context: I am excited to spend time with everyone from camp! Utterance: 'That's awesome! I really love being out here with my son. Do you think you could spare some food?'*

***Output:** Yes"*

We observe that prompts initially contained [Instruction, Input] or [Title, Definition, Emphasis and Caution, Things to Avoid, Positive, negative examples] as the Prompts, SuperNI paper's prompt **evolved** to much detailed prompt. SuperNI's prompt contained Definition, Positive example input, output and explanation, negative example input, output and explanation and then the evaluation instance. Another variant of Instruction tuning is Chain-of-thought finetuning Chung *et al.* (2022) where Prompt is just the instruction but the response contained output label and explanations. An example is given below to showcase this:

" ***Instruction:*** *Answer the following question by reasoning step-by-step.*

Question: *The cafeteria had 23 apples. If they used 20 for lunch and bought 6 more, how many apples do they have?*

Explanation: *The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have*

Output Label: *3 + 6 = 9"*

For this work, we follow SuperNI's pattern of instruction tuning. We give two positive examples and remove explanations from examples. The reason for this is that all the experiments are conducted on T5 based models which do not have enough context length to fit it in the explanations. It was also observed in SuperNI paper that Definitions and two postive examples give the best results for instruction tuning. Table 1.1 summarizes the Instruction tuning approaches:

As instruction tuning continues to evolve, it is poised to play an increasingly crucial role in the development of next-generation LLMs Wang *et al.* (2022b); Ye *et al.* (2022). This approach enables LLMs to learn from natural language instructions, offering them the capability to address a wider range of tasks, adapt to new situations, generalize more effectively, and provide valuable feedback. In doing so, it brings us closer to the realization of truly intelligent language models Bai *et al.* (2022a); Nakano *et al.* (2021); Bai *et al.* (2022b). Instruction tuning on human feedback comes at the

Paper	**Prompt**	**Response**
Mishra *et al.* (2022b)	Title, Definition, Emphasis and Caution, Things to Avoid Positive examples and explanations Negative examples and explanations Input	Output Label
Wei *et al.* (2021)	Input, Options	Output Label
Wang *et al.* (2022c)	Definition Positive examples and explanations Negative examples and explanations Input	Output Label
Chung *et al.* (2022)	Instruction, Question	Explaination, Output Label
Ours	**Defintion, 2 Positive examples** **Input**	**Output Label**

Table 1.1: Table demonstrating evolution of Instruction tuning prompts and responses

expense of performance on a wide array of more traditional NLP tasks Ouyang *et al.* (2022); Glaese *et al.* (2022).

1.2 Motivation

Recent studies in the instruction paradigm demonstrate the generalizability of models that are instruction tuned on training tasks and evaluated using few shot inference Wang *et al.* (2022c); Wei *et al.* (2021), as shown in the first row of Fig. 1.1. Despite this, SOTA performance is obtained by fully supervised finetuning on all available downstream training data as shown in the 4th row of Fig. 1.1.

In real-world situations, there is usually a limited amount of data available for finetuning, which is somewhere between few shot inference and fully supervised finetuning. Given this context, we pose the question - if we use a small amount of the data from these "Held-In" downstream tasks, how quickly could the model learn in the

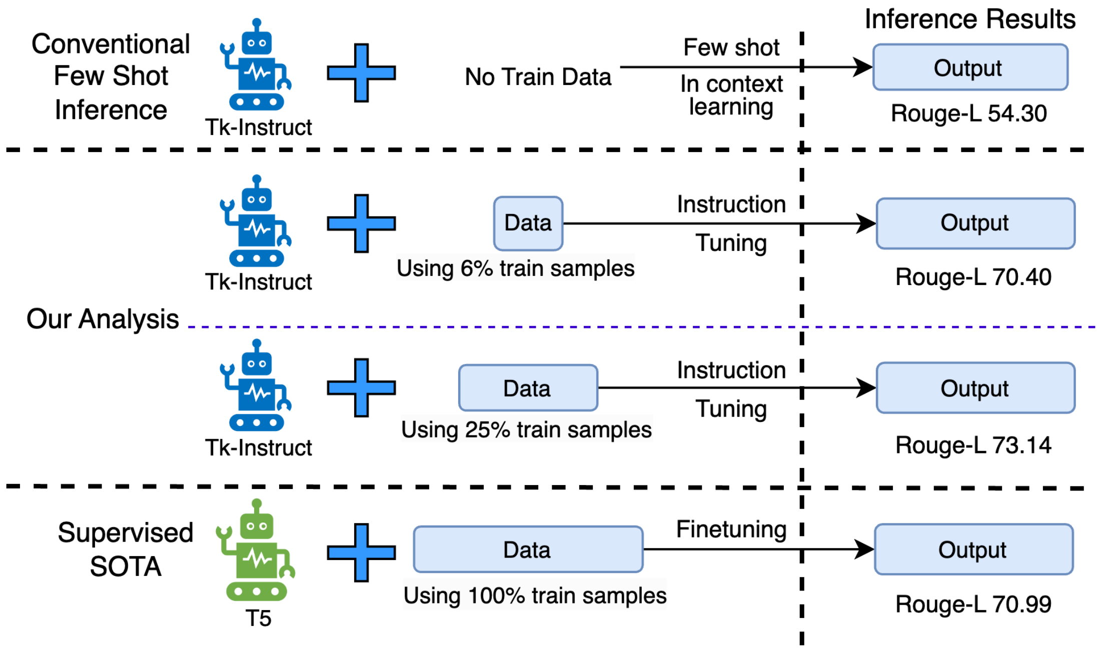

Figure 1.1: Showcasing the difference between the few shot inference, fully supervised finetuning, and our proposed analysis. The first row represents conventional few shot inference using Tk-Instruct which results in a score of 54.30. The fourth row indicates supervised SOTA that uses 100% of downstream train data to finetune T5-3B to get a SOTA score of 70.99. **Our findings** demonstrate the quick learning ability of the instruction tuned model. Using only 6% of downstream train data, Tk-Instruct achieved a score of 70.40. Surpassing SOTA by 2 points with 25% of downstream train data, our results highlight the MTL setting.

instruction paradigm? To answer this, we evaluate the minimal downstream training data required by instruction tuned models to perform transfer learning and match the performance of supervised SOTA models. We experiment on unseen tasks of Super Natural Instructions (SuperNI) Wang *et al.* (2022c), comprising of 119 tasks. We experiment with single-task learning or Expert Model setting (STL), i.e. training 119 task-specific models, and multi-task learning (MTL), where a single model is trained to solve all 119 tasks. We use Tk-Instruct 3B (T5-3B, instruction tuned on 757 tasks of SuperNI) as the instruction tuned model Wang *et al.* (2022c) and use T5-3B Raffel *et al.* (2020) as our non-instruction tuned model. We find that in the STL setting, we achieve competitive results with just 5.91% of the training data (68.34 ROUGE-L) and surpass the supervised SOTA score when using only 25.33% of the entire dataset (71.71 ROUGE-L). In the MTL setting, when using 6% of the train split, we match the SOTA performance (70.40 ROUGE-L), as shown in the 2nd row of Fig. 1. We outperform SOTA by roughly 2% (73.14 ROUGE-L) when using 25% of the train split (3rd row of Fig. 1.) and 3.69% when using 100% of the train split. To the best of our knowledge, we are first to explore the space of sample efficiency in instruction tuned large language models in both STL and MTL setups. Details about the experimental setup are described in §4, and results are described in **§5.**

We analyze the impact of instructions by investigating sample efficiency across diverse ranges, by developing multiple baselines to simulate low resource settings pertaining to training data availability. Our findings highlight sample efficiency achieved through instruction tuning, reaching up to 75%, even in limited training data.

We delve into the impact of instruction tuning as an initial pre-finetuning step. We develop two baselines (for both STL and MTL setups) employing pre-finetuning without instructions. These baselines undergo further finetuning on the downstream training set. Our findings demonstrate an increase in the performance of Tk-Instruct over the baselines by 3% and 5% in the STL and MTL setups, respectively. This highlights the impact of instructions during pre-finetuning in terms of facilitating transfer learning. We finally perform a category-wise analysis to investigate the impact of instruction tuning on different task categories. Our findings reveal that tasks falling under the textual entailment category demonstrate the most substantial improvements through instruction tuning. On the other hand, tasks related to question rewriting and title generation exhibit challenges and limitations when subjected to instruction tuning.

Contribution:

1. **We show that an instruction tuned model using just 6% of downstream train data matches the performance of a supervised SOTA model:** This work demonstrates that instruction tuning can enable language models to quickly learn new tasks with very limited training data for held-in tasks. Specifically, they show that the instruction tuned model Tk-Instruct matches state-of-the-art performance on the SuperNI benchmark when trained on just 6% of the available data per task.

2. **We find that the instruction tuned models perform up to 3% better than the SOTA when instruction tuned with 100% of the data:** When trained on the full SuperNI dataset, Tk-Instruct outperforms the previous SOTA (a supervised finetuned T5-3B model) by 3.69% in terms of ROUGE-L score. This highlights the effectiveness of instruction tuning for improving model performance given sufficient data.

3. **To investigate scenarios with significantly limited downstream train data, we conduct a comprehensive analysis by constructing multiple baselines:** To simulate low-resource scenarios, we construct several baselines using reduced amounts of downstream training data. For example, in the single-task learning setting, they show Tk-Instruct surpasses the SOTA after being trained on just 25% of the full dataset. These experiments demonstrate the sample efficiency benefits of instruction tuning.

4. **We show the impact of our method on various categories of tasks:**

> This work analyzes the impact of instruction tuning across different task categories in SuperNI. They find certain tasks like textual entailment benefit greatly from instruction tuning. However, some task types like question rewriting and title generation perform worse with instruction tuning compared to standard finetuning. This highlights the variability in effectiveness across tasks.

In summary, the key results are that instruction tuning enables efficient learning from limited data, outperforms standard finetuning given sufficient data, demonstrates sample efficiency vs baselines, and has variable impacts across different task categories. The analysis provides insights into the strengths and limitations of instruction-based learning.

1.3 Structure of Thesis

This work with explaining what is instruction tuning and its importance in downstream task especially with respect to sample efficiency. Chapter 2 offers an overview of the state of research in this domain and its dominant paradigms. In chapter 3 describes single task learning, multi task learning and instruction tuned modelling. Chapter 4 details the proposed methodology and describes the baselines that were created to compare with the proposed approach. The chapter also highlights the datasets and evaluation metrics that were used for our experiments. Chapter 5 discusses results of out experiments and comparison with different baselines. This chapter also provides a comprehensive analysis of our method in different settings. The final chapter summarizes key results and findings, while detailing existing limitations and possible avenues of future research.

Chapter 2

LITERATURE REVIEW

Multi-task learning using LLMs (Language Models) has consistently shown performance benefits over task-specific learning Mishra *et al.* (2022b); Ye *et al.* (2021); Lin *et al.* (2022); Chen *et al.* (2022); Yang *et al.* (2022a); Chen *et al.* (2021a); Zhang and Chai (2021); Chung *et al.* (2022). Instruction-based learning has emerged as a promising paradigm in LLMs Malkiel and Wolf (2021); Shao *et al.* (2021); Ouyang *et al.* (2022); Kang *et al.* (2022); Liu *et al.* (2022); Schick and Schütze (2022); Menon *et al.* (2022); Anderson (2022); Su *et al.* (2022a), with recent studies exploring various aspects such as dialogue generation Gupta *et al.* (2022b), multimodality Xu *et al.* (2022b), chain of thought Wei *et al.* (2022), distributed training Jang *et al.* (2023), and federated learning Zhang *et al.* (2023b). Moreover, the effectiveness of Prompts and Instructions Ma *et al.* (2023); Zheng *et al.* (2023) has been demonstrated in low-resource settings Le Scao and Rush (2021), and different variants of prompting, including Scratchpad Nye *et al.* (2021), Majority Voting, Reframing Mishra *et al.* (2022a), Least-to-Most Prompting Zhou *et al.* (2022b), and Question Decomposition Khot *et al.* (2020), have proven effective across various tasks. Instruction-based techniques have also shown efficacy in different applications, such as NER Wang *et al.* (2022a), program synthesis Kuznia *et al.* (2022), style transfer Reif *et al.* (2021), tabular question answering Luo *et al.* (2022), relation extraction Chen *et al.* (2021b), and biomedical applications Parmar *et al.* (2022). However, the majority of the existing works have primarily focused on zero/few-shot inference scenarios Ivison *et al.* (2022a); Gu *et al.* (2022).

We do a thorough literature review of the recent instruction tuning papers and highlight the distinctions between them. We categorize the papers based on the following categories and give a broad overview about them. We also distinguish them from our work by highlighting the broad differences in the research problem those papers are solving vs the issue tackled by us.

2.1 Initial formulation

The initial set of papers apart from the ones mentioned in §1.1 initially formulated instruction tuning. Their evaluation was always on unseen or held-out tasks and sample efficiency was not explored.

1. **Finetuned Language Models Are Zero-shot Learners Wei *et al.* (2021):** The authors propose instruction tuning that involves fine-tuning LLMs on a diverse set of natural language processing (NLP) tasks, which are expressed through natural language instructions. The authors carried out instruction tuning on a 137-billion-parameter pre-trained language model known as LaMDA-PT. They fine-tuned LaMDA-PT on over 60 NLP tasks that were articulated using natural language instruction templates. The resulting model FLAN, was then evaluated on previously unseen task types to assess its zero-shot learning capabilities. FLAN improved the performance compared to its unmodified counterpart and even outperformed the zero-shot capabilities of the more extensive 175-billion-parameter GPT-3 model on 20 out of 25 evaluated tasks. These findings suggest that instruction tuning is a simple yet highly effective approach for enhancing the zero-shot task generalization of large language models. The success of instruction tuning appears to depend on several factors, including the number of fine-tuning datasets, the scale of the model, and the use of natural language instructions. The combination of instructions and few-shot examples can further boost FLAN's performance.

2. **Multitask Prompted Training Enables Zero-shot Task Generalization Sanh *et al.* (2021):** The authors propose an approach to enhance the zero-shot learning capabilities of large language models (LLMs) through explicit multitask learning. The authors introduce a system that enables the conversion of a wide range of natural language tasks into a human-readable prompted format. This system is used to create a multitask mixture of natural language processing (NLP) datasets encompassing various tasks. Subsequently, they fine-tune a pre-trained encoder-decoder model on this diverse set of tasks. The evaluation of their model, referred to as T0, includes a variety of zero-shot generalization tasks, such as natural language inference, coreference resolution, sentence completion, and word sense disambiguation. The T0 model surpasses several other models in performance, even those models that are up to 16 times larger. Furthermore, the T0 model is assessed on a subset of tasks from the BIG-Bench benchmark and consistently outperforms other models, including those up to six times its size.

3. **ZeroPrompt: Scaling Prompt-Based Pretraining to 1,000 Tasks Improves Zero-Shot Generalization Xu *et al.* (2022a):** Scaling Prompt-Based Pretraining to 1,000 Tasks Improves Zero-Shot Generalization" presents an approach to enhance the zero-shot generalization capabilities of large language models (LLMs). ZeroPrompt's effectiveness is evaluated on a diverse set of zero-shot generalization benchmarks and it consistently outperforms existing state-of-the-art prompt-based pretraining methods across all these benchmarks.

4. **OPT-IML: Scaling Language Model Instruction Meta Learning through the Lens of Generalization Iyer *et al.* (2022):** The paper tackles a significant challenge LLMs must comprehend and act upon natural language instruc-

tions to accomplish specific tasks. The authors introduce OPT-IML to enhance LLMs' generalization ability to tackle previously unseen tasks and domains. The core concept involves optimizing the IML process, which encompasses fine-tuning LLMs on a collection of tasks described via instructions. This optimization involves considerations such as instruction-tuning decisions and model size.

2.2 Enhancements

These papers are next generation of Instruction tuning refinements that explore in-context learning enhancements, increased token length for language models, enabling language models to follow complex instructions, instructions alignment, instruction tuning for black box models etc. Again, these works do not explore sample efficiency and evaluate the proposed approaches on held-out or unseen tasks.

1. **In-Context Instruction LearningYe *et al.* (2023):** LLMs are typically trained on vast amounts of text data but often lack explicit guidance on how to follow specific instructions accurately. To tackle this issue, authors introduce "In-Context Instruction Learning" (ICIL). Fine-tuning, which is often resource-intensive and time-consuming, has been a common practice to adapt LLMs to specific tasks. ICIL presents an alternative that eliminates the need for fine-tuning, making the process more efficient and cost-effective. ICIL uses a single fixed prompt, which is employed for evaluating all tasks. This fixed prompt is constructed as a concatenation of cross-task demonstrations. It serves to provide the LLM with the necessary context and information to understand the task and generate appropriate responses. The approach is designed to enhance instruction-following capabilities and zero-shot task generalization performance.

2. **ChatPLUG: Open-Domain Generative Dialogue System with Internet-Augmented Instruction Tuning for Digital Human Tian *et al.* (2023):** The paper introduces an open-domain generative dialogue system, ChatPLUG, specifically designed for digital human applications. It presents an approach called Internet-Augmented Instruction Tuning (IAT) that involves two key steps: A diverse collection of dialogue tasks encompassing knowledge, personality, multi-turn memory, empathy, and more is gathered. This ensures that ChatPLUG can excel in various aspects of digital human interactions. ChatPLUG is then fine-tuned using the collected instruction data, which is guided by natural language instruction templates. External knowledge from internet searches is integrated into the tuning process, addressing the challenge of knowledge hallucinations.

3. **LongForm: Optimizing Instruction Tuning for Long Text Generation with Corpus Extraction Köksal *et al.* (2023):** The paper introduced "LongForm," which focuses on optimizing instruction tuning for long text generation by leveraging corpus extraction. This approach comprises two key steps: Corpus Extraction and Instruction Tuning. In the Corpus Extraction phase, a diverse set of human-written documents is meticulously selected from existing corpora. These documents are then paired with augmented instructions generated by LLMs. This augmentation process allows LLMs to understand and follow instructions more effectively. Following Corpus Extraction, the generated corpus examples and task-specific examples are employed to fine-tune LLMs. This fine-tuning process enhances LLMs' capability to produce long, coherent texts that align with the provided instructions, thus addressing the challenge of generating consistent and high-quality long-form text. LongForm outperforms

state-of-the-art methods on a range of long text generation tasks, including story generation and text summarization.The LLMs fine-tuned with LongForm are not only coherent and consistent with instructions but also of high quality.

4. **WizardLM: Empowering Large Language Models to Follow Complex Instructions Xu *et al.* (2023):** The paper introduces WizardLM designed to empower LLMs to follow complex instructions. This novel approach comprises two key steps: ”Instruction Evolution” and ”Instruction Fine-tuning.” WizardLM leverages a process known as ”Evol-Instruct” to iteratively enhance a set of initial instructions, gradually increasing their complexity and diversity. This evolution process leverages the capabilities of an LLM to rewrite and augment instructions based on specific prompts. As a result, it generates a rich and diverse dataset of complex instructions, providing the LLM with exposure to a wide array of instruction types. Following the Instruction Evolution process, the evolved instruction data is utilized to fine-tune an LLM, enhancing its ability to follow complex instructions and adapt to varying conditions. This fine-tuning procedure equips the LLM to learn and generalize effectively across a broader spectrum of instruction types and task complexities. WizardLM empowers LLMs to understand and execute complex, multi-step instructions effectively through the use of evolved instruction data. Evol-Instruct successfully generates complex and diverse instructions, significantly expanding the scope of instruction data accessible for LLM training.

5. **Generation-driven Contrastive Self-training for Zero-shot Text Classification with Instruction-tuned GPT Zhang *et al.* (2023e):** Traditional text classification methods often rely on labeled data for each class, which can be resource-intensive and time-consuming to obtain. To mitigate this limi-

tation, the authors proposed a novel approach named GenCSTR (Generation-driven Contrastive Self-training) that leverages advanced techniques and large language models (LLMs) for zero-shot text classification. GenCSTR involves two steps: Initially, a GPT-based LLM is fine-tuned using a dataset containing text descriptions along with corresponding instructions. This fine-tuning process equips the model with the ability to understand and classify text based on provided instructions. Essentially, this phase serves as the foundation for the model's subsequent zero-shot classification capabilities. The instruction-tuned LLM is tasked with generating pseudo-labels for unlabeled data by classifying each text based on the given instructions. These pseudo-labels are subsequently employed to further train and refine the model. By iteratively self-training and utilizing generated labels, the model gradually becomes adept at classifying new, previously unseen text. This process enhances the model's zero-shot classification capabilities, as it becomes more adept at working with unannotated data.

6. **Improving Cross-Task Generalization with Step-by-Step Instructions: Wu *et al.* (2023b)** The paper introduces Cross-task generalization, the ability of LLMs to apply their learned knowledge to tasks they were not specifically trained for, is a critical challenge, and this research addresses it by introducing a novel approach involving step-by-step instructions. The paper recognizes the limitations of conventional LLM training, which predominantly involves exposing models to a vast amount of unstructured text data. While this training equips the models with a vast vocabulary and language understanding, it often falls short in enabling them to generalize effectively to new and diverse tasks. The absence of explicit task-specific guidance hampers their ability to

decompose and understand the underlying structure of tasks that require step-by-step processes. To tackle this challenge, the authors propose the integration of step-by-step instructions during LLM training. These instructions provide a detailed breakdown of how to perform a task, offering specific procedures and guidance. By introducing this form of supervision, LLMs can gain insights into the structural composition of tasks, learning not just what to do but also how to do it. The paper conducts a empirical evaluation of the approach on multiple cross-task generalization benchmarks, including GLUE, FewGLUE, and XTREME-ZSL. The results reveal that LLMs trained with step-by-step instructions outperform those trained without such instructions. This empirical evidence underscores the effectiveness of this approach in improving cross-task generalization.

7. **Aligning Instruction Tasks Unlocks Large Language Models as Zero-Shot Relation Extractors: Zhang *et al.* (2023d)** The paper presents an improvement in the field of relation extraction (RE) using large language models (LLMs). Traditional LLM-based approaches for RE typically demand labeled data for each relation type, which is often costly and labor-intensive to obtain. The authors introduce the concept of instruction alignment, unlocking the potential of LLMs as zero-shot relation extractors without requiring explicit relation annotations. Instruction alignment means aligning instruction descriptions with corresponding relation labels. This alignment allows LLMs to establish a connection between instructions and relations, enabling them to understand and extract relations without the need for explicit relation annotations. The evaluation of this approach on benchmark datasets, including TACRED, fewGLUE, and XTREME-ZSL, demonstrates its effectiveness. LLMs trained with instruc-

tion alignment consistently outperform state-of-the-art methods, especially in zero-shot relation extraction scenarios. Instruction alignment reduces the need for extensive manual annotation of relation data, making LLMs more practical for relation extraction tasks.

8. **The CoT Collection: Improving Zero-shot and Few-shot Learning of Language Models via Chain-of-Thought Fine-Tuning: Kim *et al.* (2023)** The paper addresses a critical challenge that LLMs inability to reason step-by-step through unseen tasks. The authors propose Chain-of-Thought (CoT) fine-tuning to enhance the zero-shot and few-shot learning capabilities of LLMs. CoT fine-tuning involves training LLMs on a dataset of instructions paired with CoT reasoning processes. Each instruction is accompanied by a chain of thought, providing explicit steps and justifications on how to execute the task. The authors also developed the CoT Collection, a comprehensive dataset comprising instruction-response pairs and CoT reasoning processes for a wide array of NLP tasks. The dataset is designed to augment the reasoning abilities of LLMs in the context of zero-shot and few-shot learning. The paper reports the positive impact of CoT fine-tuning on various benchmarks, including GLUE, FewGLUE, and XTREME-ZSL. LLMs trained with CoT fine-tuning consistently outperformed state-of-the-art methods on these benchmarks, showcasing the efficacy of this approach in enhancing zero-shot and few-shot learning.

9. **InstructZero: Efficient Instruction Optimization for Black-Box Large Language Models: Chen *et al.* (2023b)** The paper addresses the challenge of optimizing instructions for large language models (LLMs) that are considered black-box models, meaning their internal workings are not directly accessible or modifiable. Optimizing LLMs' performance for specific instruction-following

tasks, especially when the internal mechanisms are not transparent, presents a significant challenge.InstructZero is introduced as an approach to optimizing instructions for black-box LLMs. What sets InstructZero apart is its efficient use of Bayesian optimization to search for optimal instructions for a given task, even in situations where direct access to the LLM's inner workings is not available. InstructZero's methodology involves an iterative process of generating candidate instructions, evaluating their performance on the black-box LLM, and using the feedback obtained to guide the search for better instructions. Notably, this evaluation of candidate instructions is performed using a proxy LLM, which is an open-source LLM that can be accessed and modified directly. The authors conducted evaluations of InstructZero across various instruction-following benchmarks, including Dialogue, Information Retrieval, and Summarization. The results demonstrated that InstructZero outperformed random search and other baseline methods. This underscores the effectiveness of the Bayesian optimization approach for instruction optimization, particularly in situations involving black-box LLMs.

10. **One Embedder, Any Task: Instruction-Finetuned Text Embeddings: Su *et al.* (2022b)** The paper introduces INSTRUCTOR, which is a single embedder with capability to generate text embeddings customized for a diverse range of downstream tasks and domains without necessitating additional training. The process through which INSTRUCTOR accomplishes this involves the following key steps: Firstly, a wide array of tasks (in this instance, 330 diverse tasks) is annotated with instructions. These instructions offer guidance on how the text should be embedded in the context of each task. For example, an instruction might be: "Represent the medical content of the following

discharge documents." Secondly, INSTRUCTOR is then trained on this multitask mixture, incorporating the contrastive loss to ensure that the generated text embeddings are highly informative and specifically relevant to the intended task. This phase leverages the instructions provided for each task. Lastly, the authors conduct a thorough evaluation of INSTRUCTOR's performance across a comprehensive set of 70 embedding evaluation tasks. Importantly, 66 of these tasks are entirely unseen during the training phase. The evaluation spans tasks related to classification, information retrieval, semantic textual similarity, text generation, and more.

11. **HINT: Hypernetwork Instruction Tuning for Efficient Zero-Shot Generalisation Ivison *et al.* (2022a):** The paper introduces HINT, which leverages the innovative concept of hypernetworks to enhance the adaptability of LLMs to new tasks without the need for extensive retraining. HINT operates by incorporating hypernetworks, which are lightweight neural networks designed to generate parameters for other neural networks, into the LLM architecture. These hypernetworks, as the name suggests, are instrumental in fine-tuning the parameters of an LLM based on task-specific instructions.

12. **The Flan Collection: Designing Data and Methods for Effective Instruction Tuning: Longpre *et al.* (2023)** The paper introduces flan collection which is a comprehensive endeavor aimed at facilitating the development and evaluation of instruction tuning methods. It comprises a large-scale dataset and a suite of methods that hold the promise of making LLMs adept at following instructions and executing tasks as precisely as specified. The Flan Collection stands out as a rich and varied compendium of instruction-task pairs, spanning a wide spectrum of NLP tasks. These tasks encompass a range

of applications, including question answering, summarization, and translation. Notably, the instructions are crafted with meticulous attention to detail, ensuring they are clear, concise, and contextually informative. This level of precision in instruction design provides LLMs with the requisite contextual understanding for task execution. The impact of the Flan Collection becomes evident when various instruction tuning methods are applied and evaluated, including well-known techniques such as prompt-based learning, few-shot learning, and zero-shot learning.

13. **Exploring the Benefits of Training Expert Language Models over Instruction Tuning:Jang *et al.* (2023)** The paper delves into the comparative effectiveness of two prevalent approaches in training large language models (LLMs) for specific tasks. These approaches are ”multitask learning,” where a single LLM is trained on a diverse set of tasks, and ”expert learning,” which involves training separate LLMs, each specialized in a single task. The rationale behind this study is to determine which of these approaches is more effective in terms of performance on unseen tasks, data efficiency, and generalization capability. Studies show that ”expert LLMs,” those trained on a single task, consistently outperformed ”multitask LLMs” across various unseen tasks. This observation highlights the benefits of task-specific expertise in the context of LLMs. The expert LLMs demonstrated superior performance, especially in situations where specific nuances of individual tasks play a pivotal role. The evaluation of these approaches was carried out on two benchmark datasets, the Super-NaturalInstructions (SNLI) benchmark and the Unified-SKG (USKG) benchmark. The results show that expert LLMs consistently outperforming multitask LLMs. The achievement of higher accuracy in both

benchmarks underscores the advantages of task-specific expertise.

2.3 Alignment

These research work focuses on alignment of language models via instruction tuning. Alignment methods include instruction tuning with human provided examples or going for reward free instruction relabeling. Their evaluation was always on unseen or held-out tasks and sample efficiency was not explored.

1. **Training Language Models to Follow Instructions with Human Feedback Ouyang *et al.* (2022):** This research introduces a novel methodology for training language models to follow instructions using human feedback. The core approach involves fine-tuning a pre-trained language model with a dataset of human-provided demonstrations of the desired behavior and further enhancing this fine-tuned model through reinforcement learning from human feedback. The motivation for this work stems from the fact that LLMs often generate outputs that may be untruthful, toxic, or not aligned with the user's intent. To address this issue, the authors leverage human feedback as a means of guiding LLMs to produce more accurate, safe, and contextually relevant responses. They create a dataset of human-written demonstrations that illustrate the desired output behavior when given specific prompts. This dataset serves as the foundation for training supervised learning baselines. To further refine the fine-tuned models, a dataset of human-labeled comparisons is collected, allowing the creation of a reward model that can predict which model output the human labelers would prefer. This reinforcement learning component plays a crucial role in improving model behavior, making it more aligned with human intent and preferences. The authors evaluate their approach across a wide spectrum of tasks and demonstrate that fine-tuning LLMs using human preferences signifi-

cantly enhances model behavior and performance. Notably, their InstructGPT model, fine-tuned with human feedback, outperforms 175B GPT-3 model in human evaluations, despite having significantly fewer parameters. This improvement also extends to enhancing truthfulness and reducing the generation of toxic outputs, while minimizing performance regressions on public NLP datasets.

2. **The Wisdom of Hindsight Makes Language Models Better Instruction Followers Zhang *et al.* (2023g):** The paper titled attempts to make LLMs effective instruction followers by introducing "Hindsight Instruction Relabeling" (HIR). HIR is a novel algorithmic approach designed to improve the alignment between language models and instructions. Instead of relying on complex training pipelines for reward and value networks, HIR adopts a reward-free strategy. It leverages the wisdom of hindsight by converting feedback into instructions. This conversion allows the model to receive more direct and unambiguous guidance about its performance. HIR operates in two distinct phases: an online exploration phase and an offline training phase. These phases are alternated until the model converges and aligns effectively with the instructions provided. The paper demonstrates the effectiveness of HIR by extensively evaluating it on 12 challenging BigBench reasoning tasks. The results indicate that HIR outperforms baseline algorithms, making it a compelling choice for improving instruction-following capabilities in language models.

2.4 Benchmarks

These set of papers propose various benchmarks to to highlight some of the drawbacks in current instruction tuning approaches.

1. **GPTScore: Evaluate as You Desire Fu *et al.* (2023):** The paper addresses the evaluation of text generation quality. Accurate evaluation metrics are crucial

for assessing the capabilities of LLMs and ensuring their effectiveness in various natural language processing (NLP) tasks. Traditional evaluation metrics often rely on manually labeled datasets, which are labor-intensive and costly to create. GPTScore distinguishes itself by utilizing the zero-shot learning capacities of GPTs to evaluate generated texts based on natural language instructions. These instructions can specify different aspects of the desired text, such as coherence, relevance, and informativeness. GPTs, being adept at understanding natural language, then generate a score for the text based on their interpretation of these instructions. The authors conducted a comprehensive evaluation of GPTScore on a diverse set of text generation tasks and datasets. These tasks encompassed summarization, machine translation, dialogue response generation, and more. GPTScore achieved higher correlations with human judgments of text quality, underlining its reliability.

2. **TABLET: Learning From Instructions For Tabular Data Slack and Singh (2023):** The paper addresses the problem to utilize instructions for tabular data. The authors focus was to adapt LLMs to structured and organized data in tabular format. TABLET stands for "TABular Learning from Explicit and Texual instruction," and it comprises a benchmark of 20 diverse tabular datasets, each annotated with instructions that exhibit variability in terms of phrasing, granularity, and technicality. These instructions are designed to guide LLMs on how to work with the data in the tables effectively. This benchmark provides a standardized platform to evaluate LLMs' ability to learn from instructions and perform tasks on tabular data. The researchers conducted comprehensive evaluations of several state-of-the-art LLMs, including Flan-T5-11b and ChatGPT, using the TABLET benchmark. The key findings can be sum-

marized as follows: LLMs can adapt to structured data with the appropriate guidance. LLMs exhibit sensitivity to the phrasing and granularity of the provided instructions. The choice of words and level of detail in instructions can impact the performance and response of LLMs. LLMs struggle with tasks that require higher-order reasoning and inference in the context of tabular data.

3. **Resources and Few-shot Learners for In-context Learning in Slavic Languages Štefánik *et al.* (2023):** The paper addresses the issue of lack of benchmark for Slavic languages due to their rich morphology and complex syntax. To tackle this challenge, benchmark development and the creation of few-shot learners for Slavic languages. This benchmark comprises a dataset with over 100,000 dialogue turns in Russian and Polish, each containing an instruction and a corresponding response. Importantly, this dataset is enriched with various linguistic annotations, such as part-of-speech tags, dependency labels, and named entities. This makes it a valuable resource for developing, testing, and evaluating in-context learning systems tailored to Slavic languages. The research extends beyond benchmark creation to the development of few-shot learners tailored to Slavic languages. These few-shot learners are designed to effectively acquire knowledge from a limited number of examples, making them particularly useful in scenarios where extensive training data is scarce. The few-shot learners in this study are based on diverse architectural paradigms, including recurrent neural networks, transformers, and memory networks. The notable achievement is that these learners demonstrate high accuracy, even with as few as 16 examples per instruction. This finding opens the door to efficient learning in resource-constrained environments and expands the practical utility of NLP systems in Slavic languages.

4. **Poisoning Language Models During Instruction Tuning Wan *et al.* (2023):** The paper shines a light on a vulnerability that these models face poisoning attacks. Poisoning attacks entail the injection of malicious instructions or data into the training process of LLMs. These poisoned instructions are strategically designed to manipulate the behavior of the model. The paper demonstrates that adversaries can introduce poisoned examples into the training data of instruction-tuned LLMs, effectively enabling them to manipulate model predictions when certain trigger phrases or conditions are met in the input text. The research also indicates that larger models are more vulnerable, and this has far-reaching implications considering the trend toward ever-larger models in the field.The paper highlights the limitations of existing defenses, such as data filtering or reducing model capacity. While these measures provide some level of protection, they are not foolproof and come at the cost of reduced test accuracy.

5. **Panda LLM: Training Data and Evaluation for Open-Sourced Chinese Instruction-Following Large Language Models Jiao *et al.* (2023):** The paper addresses the need for comprehensive and diverse training data for Chinese LLMs and presents a robust evaluation framework. One of the issues addressed by this research is the lack of diversity in existing Chinese LLM datasets. Often, these datasets are constrained to narrow instruction types, limiting the models' ability to generalize and handle real-world, diverse instruction formats. Panda LLM serves as a solution to this problem by providing a vast dataset with over 6 million instruction-response pairs. This dataset spans a broad spectrum of domains and task types, including dialogue, information retrieval, and summarization. The diversity in the data not only enables models

to better understand and follow instructions but also enhances their performance across a range of real-world applications. The paper introduces a robust evaluation framework that covers various aspects of instruction understanding and response generation. Metrics include those for accuracy, fluency, coherence, as well as specialized metrics for specific instruction types, such as dialogue engagement and summarization quality.

6. **STORYWARS: A Dataset and Instruction Tuning Baselines for Collaborative Story Understanding and Generation Du and Chilton (2023):** The paper addresses the challenges of collaborative storytelling. Collaborative storytelling involves multiple authors collectively contributing to a narrative, making it a complex task for NLP models that require an understanding of human collaboration, narrative structure, and creative expression. To tackle these challenges, the authors introduce the STORYWARS dataset, which comprises over 40,000 collaborative stories created by a diverse community of 9,400 authors on an online platform. This dataset covers a wide range of genres and includes rich information, such as annotations, ratings, and author profiles. The paper designs 12 task types, split into 7 understanding tasks and 5 generation tasks. These tasks range from identifying authors and genres to generating story continuations and assessing story quality. This comprehensive suite of tasks offers a thorough evaluation platform for NLP models focusing on collaborative storytelling. The paper introduces INSTRUCTSTORY, an instruction-tuned model designed to excel in the context of collaborative story understanding and generation. Instruction tuning involves fine-tuning a pre-trained language model with specific instructions that guide the model towards desired behaviors and tasks. INSTRUCTSTORY is evaluated across three learning scenarios: fully-

supervised, few-shot, and zero-shot. INSTRUCTSTORY outperforms baseline models, demonstrating the potential of instruction tuning in this domain.

7. **Dynosaur: A Dynamic Growth Paradigm for Instruction-Tuning Data Curation Yin *et al.* (2023):** The paper a pivotal challenge in the realm of large language models (LLMs). While LLMs have showcased remarkable capabilities in a wide range of natural language processing (NLP) tasks, their performance in instruction-following tasks is often hampered by the quality and diversity of the training data. This limitation is primarily due to the difficulty and costs associated with collecting and curating large datasets. The authors propose "Dynosaur," a novel dynamic growth paradigm for curating instruction-tuning data. Dynosaur utilizes a multi-stage process that actively refines and augments the training data for instruction-following tasks, leading to continuous improvements in data quality and diversity over time. The Dynosaur approach comprises three main stages: Firstly, initial Data Collection In this phase, an initial dataset of instruction-response pairs is amassed from diverse sources, including human-generated instructions, crowd-sourced annotations, and pre-existing dialogue datasets. Secondly, active learning techniques are applied to select the most informative and challenging instruction-response pairs, which are then subjected to human review and feedback. This iterative process helps enhance the quality and diversity of the training data. Lastly, data augmentation techniques are employed to generate new instruction-response pairs based on the existing dataset. This step significantly expands the dataset's size and diversity. The updated dataset is regularly assessed to measure its impact on the performance of LLMs in instruction-following tasks.

8. **M3IT: A Large-Scale Dataset towards Multi-Modal Multilingual In-**

struction Tuning Li *et al.* (2023b): The paper addresses the challenges associated with large language models (LLMs) concerning tasks that require combining information from both textual and visual modalities. LLMs often struggle with understanding and following instructions that involve both text and visual elements. Additionally, many LLMs are predominantly trained on monolingual data, limiting their capacity to handle instructions in multiple languages. The authors introduced the Multi-Modal Multilingual Instruction Tuning (M3IT) dataset, a large-scale resource aimed at improving the performance of LLMs in multi-modal and multilingual instruction-following tasks. M3IT is a substantial collection of 2.4 million instances of instruction-response pairs, encompassing 40 meticulously curated datasets. M3IT has instructions written in 80 different languages, making it a multilingual resource. Furthermore, the dataset spans a diverse range of task types, including image captioning, visual question answering, and visual dialog. This diverse, multi-modal, and multilingual nature sets it apart. M3IT was constructed through four key stages: manual instruction writing, dataset pre-processing, quality checks, and dataset translation. These steps culminated in a dataset that encompasses a rich variety of tasks and languages.

2.5 PEFT methods

These set of papers explore parameter efficient instruction tuning methods and use fewer layers for finetuning LLMs. These papers propose a change in model architecture for ensuring models with larger parameters can be finetuned with a fixed set of compute. These methods are faster to run compared to fully supervised finetuning/ instruction tuning as fewer parameters are being updated. These works are targeting efficiency but they do that in terms of model compute and not focus on instruction tuning. Our work focuses on sample efficiency of instruction tuned models. While using our work also result in reduced compute, our methods does a full backward pass at the model and use all the layers. PEFT and sample efficent instruction tuned models can be used concurrently.

1. **A Comparative Study between Full-Parameter and LoRA-based Fine-Tuning on Chinese Instruction Data for Instruction Following Large Language Model Sun *et al.* (2023):** The paper did a comparative study to investigate two different fine-tuning approaches: full-parameter fine-tuning and LoRA-based fine-tuning. Their goal was to determine the most effective method for developing instruction-following LLMs for Chinese language. The study involved comparison between full-parameter fine-tuning and LoRA-based fine-tuning, leveraging a benchmark dataset that encompassed diverse instruction-following tasks. The study's findings show that LoRA outperformed full-parameter fine-tuning on several metrics. It demonstrated enhanced accuracy in following instructions, increased task completion rates, and improved the naturalness of responses. These improvements are attributed to LoRA's unique ability to efficiently capture and utilize specific information that is relevant to instruction-following tasks.

2. **LLaMA-Adapter V2: Parameter-Efficient Visual Instruction Model Gao *et al.* (2023):** Large language models (LLMs) have demonstrated significant capabilities in various NLP tasks, they often require extensive computational resources and large model sizes, which limit their applicability in resource-constrained environments and real-time applications. LLaMA-Adapter V2 is a parameter-efficient visual instruction model. This model is designed to enable LLMs to follow visual and textual instructions effectively while maintaining a

compact model size. LLaMA-Adapter V2 enhances parameter unlocking, allowing more learnable parameters in the LLaMA base model. This distributed learnability enables the model to better grasp and utilize instruction-following guidance without solely relying on additional adapters.The model incorporates an early fusion strategy, where visual tokens are introduced into the early layers of the LLM. This approach facilitates the seamless integration of visual information and instruction comprehension, leading to improved performance. LLaMA-Adapter V2 utilizes a joint training paradigm to optimize different groups of learnable parameters, including the LLaMA base model, adapters, and vision encoder. This balanced approach ensures efficient learning and contributes to the model's overall success. The authors conducted extensive evaluations of LLaMA-Adapter V2 across various visual instruction following benchmarks, including SNLI-VE, Flickr30k Entities, and VQA-X. The findings of these evaluations demonstrate that LLaMA-Adapter V2 outperforms state-of-the-art methods in these benchmarks while having a significantly smaller parameter footprint.

2.6 Prompt Engineering

These set of papers explore different set of prompts that can be used while instruction tuning.

1. **Prompt Consistency for Zero-Shot Task Generalization Zhou *et al.* (2022a):** The paper introduces a novel approach called Prompt Consistency, highlighting to improve LLMs' zero-shot capabilities. Prompt Consistency operates on the principle of making LLMs' predictions consistent across a diverse set of prompts for a given task. This approach encourages LLMs to not just respond effectively to individual prompts but to truly understand the underly-

ing structure and intent of tasks, thereby promoting better generalization and avoiding overfitting to specific prompts. The findings demonstrate that Prompt Consistency outperforms other state-of-the-art methods that aim to improve zero-shot performance, such as prompt-based learning and multi-task learning. The practical significance is noteworthy because it shows the potential to significantly enhance the zero-shot generalization of LLMs, which has been a challenge in the field. The method's practicality extends to cases where only a few labeled examples are available for a specific task.

2. **Instruction Induction: From Few Examples to Natural Language Task Descriptions Honovich *et al.* (2022b)**: The paper introduces a method that prompts an LLM to generate natural language instructions based on input-output examples provided. The key innovation here lies in the ability to derive coherent and contextually relevant task descriptions from just a handful of examples. The authors conducted a rigorous evaluation of the method across a spectrum of NLP tasks, including summarization, question answering, and code generation. The findings demonstrate that Instruction Induction is not only capable of generating natural language instructions but that these instructions are characterized by accuracy and informativeness.

3. **reStructured Pre-training Yuan and Liu (2022):** This paper shows that reformating the training data into a more efficient and informative structure, creates an enhanced learning environment for LLMs. By doing so, LLMs can effectively learn from this structured data and subsequently exhibit superior performance across a variety of NLP tasks.

2.7 Synthetic Data Creation

These sets of papers used LLMs or instruction-tuned LMs to create synthetic data samples for particular sets of tasks. Their evaluation was always on unseen or held-out tasks and sample efficiency was not explored.

1. **Instruction Tuning with GPT-4 Peng *et al.* (2023):** The paper self-supervised learning capabilities of GPT-4, to generate instruction-following data that facilitates the fine-tuning of LLMs. The primary idea is: GPT-4 is employed to generate instruction-following data by providing it with examples of task instructions and the desired outputs. GPT-4 learns from these examples to generate new instructions for various tasks. The generated instruction-following data is subsequently used to fine-tune an LLM. This fine-tuning process enhances the LLM's ability to follow instructions and perform specific tasks, eliminating the need for extensive manual fine-tuning. The approach empowers LLMs to better follow instructions, broadening their utility in tasks that require explicit and precise guidance, such as chatbots, customer service, and task-oriented assistants. It streamlines the adaptation of LLMs to new tasks. This leads to significant reductions in computational costs and time associated with manual fine-tuning. The approach also makes LLMs more adaptable to real-world scenarios where following instructions is essential. This expansion of applicability supports a broader range of NLP tasks.

2. **Controlled Text Generation with Natural Language Instructions Zhou *et al.* (2023):** LLMs using their ability to generate text that adheres to specific constraints has remained a challenge. This issue arises from the fact that LLMs are primarily trained on extensive unlabeled text data, which does not inherently guide them on how to produce text that meets specific requirements. The

authors present InstructCTG (Controlled Text Generation). The core premise of InstructCTG is to employ natural language instructions to precisely specify the desired constraints for text generation. This methodology comprises several steps: Initially, natural texts are annotated to indicate the linguistic and extra-linguistic constraints they satisfy. This annotation process is accomplished through a combination of established natural language processing (NLP) tools and simple heuristics. The annotated constraints are then transformed into instructions using verbalization techniques. These instructions are subsequently combined with the original natural language sentences to create a corpus that forms the basis for weakly supervised training data. The pre-trained LLM is fine-tuned on this augmented dataset, which now includes the original text and their corresponding instructions. This fine-tuning process allows the LLM to understand how to generate text that complies with the specified constraints. The authors assessed the performance of InstructCTG across several text generation tasks, encompassing various constraint types, including style transfer, summarization, and paraphrase generation. Their findings indicated that InstructCTG surpassed the performance of state-of-the-art methods in these tasks, effectively demonstrating its potential in controlled text generation.

3. **Unnatural Instructions: Tuning Language Models with (Almost) No Human Labor Honovich *et al.* (2022a):** The paper introduces a vast dataset of creative and diverse instructions that are generated with virtually no human involvement. It involves prompting an LLM with three seed examples of instructions and then eliciting a fourth instruction. This process is iteratively repeated, resulting in a dataset of over 64,000 instruction-input-output triplets. To further expand the dataset's diversity and coverage, the LLM is prompted

to rephrase each instruction, ultimately leading to a total of approximately 240,000 examples. Despite the presence of some inherent noise in the generated instructions, the models trained on Unnatural Instructions exhibit performance that rivals models trained on manually curated datasets. It surpasses the effectiveness of state-of-the-art models like T0++ and Tk-Instruct across various benchmark tasks.

4. **Self-Instruct: Aligning Language Model with Self-Generated Instructions Wang *et al.* (2022b):** The paper titled "Self-Instruct: Aligning Language Model with Self-Generated Instructions" aims to align LLMs with instructions by leveraging the model's own generation capabilities. Self-Instruct operates through a series of three main steps: Firstly, the LLM is tasked with generating instructions, input examples, and corresponding output examples for various tasks. This step taps into the LLM's inherent generative capacity. Secondly, the generated instructions, input examples, and output examples undergo a filtration process. This step is designed to eliminate low-quality or redundant data, ensuring that the dataset used for instruction tuning is of high quality. Lastly, the filtered data is then employed for fine-tuning the original LLM. This fine-tuning process is a critical step in enhancing the model's ability to follow instructions effectively. The findings revealed that Self-Instruct significantly outperforms existing methods for instruction-following. This includes methods like supervised instruction tuning, zero-shot learning, and prompt-based learning.

5. **TarGEN: Targeted Data Generation with Large Language Models Gupta *et al.* (2023b):** TarGEN is a multi-step prompting strategy for generating high-quality synthetic datasets utilizing large language models (LLMs).

It is seedless, meaning that it does not require specific task instances, which broadens its applicability beyond task replication.TarGEN works as follows:

- Define the target dataset. This includes specifying the task, the desired data distribution, and any constraints.
- Generate prompts. TarGEN uses a variety of techniques to generate prompts that are likely to produce high-quality synthetic data, such as using templates, examples, and constraints.
- Generate data. TarGEN uses an LLM to generate synthetic data based on the prompts.
- Filter and label the data. TarGEN filters the generated data to remove low-quality instances and labels the remaining data.
- Self-correct the data. TarGEN uses a method called self-correction to empower LLMs to rectify inaccurately labeled instances during dataset creation, ensuring reliable labels.

TarGEN was evaluated on eight tasks from the SuperGLUE benchmark. Models trained on datasets generated by TarGEN performed approximately 1-2% points better than those trained on original datasets. When incorporating instruction tuning, the performance increased to 1-3% points better on synthetic data than on original data. TarGEN also has a number of advantages over other synthetic data generation techniques:

- It is seedless, which means that it does not require specific task instances. This makes it more broadly applicable than techniques that rely on task replication.

- It is able to generate high-quality synthetic data that is similar or better than original datasets in terms of complexity, diversity, and bias.
- It is able to self-correct inaccurately labeled instances during dataset creation, ensuring reliable labels.

TarGEN is a promising new technique for synthetic data generation. It has the potential to reduce the need for human-labeled data and to make machine learning more accessible to a wider range of applications.

2.8 Unsupervised methods

These sets of papers aim to explore instruction tuning without labelled data samples. Their evaluation was always on unseen or held-out tasks and sample efficiency was not explored.

1. **Unsupervised Cross-Task Generalization via Retrieval Augmentation Lin *et al.* (2022)**: The paper introduces a method called ReCross, which is designed to enable unsupervised cross-task generalization in LLMs. Cross-task generalization involves training LLMs to apply their learned knowledge to new tasks, even when the new tasks are significantly different from the tasks they were initially trained on. ReCross, the proposed methodology, leverages retrieval augmentation as a means to enhance the cross-task generalization ability of LLMs. It achieves this by retrieving a small, pertinent subset of training examples from a large upstream dataset. These retrieved examples are then used to augment the training data for a new task, allowing LLMs to learn and perform the new task without the need for explicit supervision. The evaluations reveal that ReCross consistently outperforms other unsupervised methods, including

prompt-based learning and multi-task learning. The observed performance improvements are significant and indicate the potential of ReCross to enhance the cross-task generalization capabilities of LLMs. The research demonstrates that ReCross enables LLMs to perform new tasks without requiring any fine-tuning. This is an advantage over supervised methods, which often necessitate the manual creation of task-specific training data and fine-tuning procedures.

2. **Learning Instructions with Unlabeled Data for Zero-Shot Cross-Task Generalization Gu *et al.* (2022):** The paper addresses a challenge in instruction learning: the need for extensive human-annotated data. UDIT stands out as a powerful solution for zero-shot cross-task generalization, thanks to its ability to learn instructions from unlabeled data. UDIT has the ability to work with a small number of unlabeled examples. This allows LLMs to generalize across tasks without human annotation. It can be used to train LLMs to perform new tasks that lack labeled data, such as generating code from natural language instructions or medical diagnosis.

2.9 Sample Efficiency

These set of papers explore sample efficiency of instruction tuned models. While they explore sample efficiency, we highlight the differences between these papers and our work in detail.

1. **Data-Efficient Finetuning Using Cross-Task Nearest Neighbors Ivison *et al.* (2022b):** The paper introduces a method that leverages the concept of cross-task nearest neighbors to fine-tune LLMs for new tasks efficiently, requiring minimal amounts of labeled data. DEFT's premise involves identifying the nearest neighbors of unlabeled, task-specific examples within a comprehensive

pool of multitask data, augmented with prompts. These nearest neighbors, referred to as cross-task nearest neighbors, are then harnessed to fine-tune the LLM. This process optimally utilizes the knowledge encapsulated in the multitask data to enhance task-specific performance.

2. **Maybe Only 0.5% Data is Needed: A Preliminary Exploration of Low Training Data Instruction Tuning Chen *et al.* (2023a):** The paper addresses the issue of massive amount of data needed for their training, a process that can be both financially and temporally expensive. The authors propose an innovative solution known as Low Training Data Instruction Tuning (LTD Instruction Tuning). LTD Instruction Tuning is centered around fine-tuning an LLM using a strategically chosen subset of the original training data. This subset comprises instructions that are most pertinent to the task, reflecting a move towards data efficiency. LLMs subjected to the LTD Instruction Tuning approach consistently outperform their counterparts trained on the full dataset. The improvements were observed even when deploying 0.5% of the original data.

2.9.1 Distinction from Sample efficiency approaches:

Ivison *et al.* (2022b) uses a small number of unlabeled target task examples and the method retrieves the most similar labeled examples from a large pool of multitask data. This retrieval is done using cross-task nearest neighbors, which finds examples from the multitask data that are most similar to the unlabeled target task examples. A detailed breakdown of the method is as follows: Extract embeddings from the unlabeled target task examples. Index the embeddings from the multitask data. Find the nearest neighbors in the multitask data for each unlabeled target task example. Retrieve the corresponding labeled examples from the multitask data. Our approach does not involves using test samples to extract relevant train samples. This is chosen to simulate a real world scenario where test samples are unavailable for access.

Chen *et al.* (2023a) employs a pruning method where firstly, each sentence is encoded into a numerical representation and then cleaned up to make it easier to analyze. Next, similar sentences are grouped together into clusters. Within each cluster, the most representative sentences are identified. These core samples are then used to improve the performance of large language models (LLMs). Finally, the performance of the LLMs is tested to see how well they have been improved. Their experimental settings are on single task learning/expert model training, a subset of our experimental setup. The pruning method described is also computationally expensive. Our approach aims to randomly select training samples and then finetune then of downstream train data/ held-in tasks. Since we are randomly selecting samples, the approach is easier to implement in a real world setting and generalizes to multiple scenarios.

2.10 Distinction from PEFT

The Few-Shot Parameter-Efficient Fine-Tuning (PEFT) paper Liu *et al.* (2022) also explores conditions for data efficiency (i.e., use of less with SOTA or better results), thus validating our focus on data efficiency; but they do not explore instruction tuning, the focus of our work. In other words, while they suggest many other methods that lead to data efficiency, we show that instruction tuning also leads to data efficiency. Our decision to study instruction tuning is to have a fair comparison with the SuperNI paper and the SOTA model in the associated leaderboard. Parameter-efficient finetuning presents a more practical and beneficial alternative to in-context learning (ICL). As stated in their work: 1. “While the benefits of PEFT address some shortcomings of fine-tuning (when compared to ICL), there has been relatively little focus on whether PEFT methods work well when very little labeled data is available.” 2. “Our goal is therefore to develop a recipe that allows a model to attain high accuracy on new tasks with limited labeled examples while allowing mixed-task batches during inference and incurring minimal computational and storage costs.” Their experiments provide compelling evidence that PEFT outperforms ICL significantly. We agree with this finding and propose to establish it as the new standard way of doing instruction tuning, replacing the conventional method that relies on ICL for unseen tasks. However, our analysis is distinct from their work and strengthens their findings on the following grounds:

- Broader Dataset Categories: PEFT’s coverage focuses on 9 datasets across 4 categories. We conduct our experiments on 119 datasets across 12 categories which show broad applicability of the findings.
- PEFT performs their experiments in multitask learning setup, while we explore a more challenging low-resource setting where a single task is available and supplemented with limited samples; Single task learning (STL) setting.
- We establish strong baselines in both STL and MTL setups, using which we explore the effect of sample efficiency, pre fine-tuning with instructions and comparison with supervised SOTA scores. This contrasts with PEFT, which employs T0, few-shot ICL with T5, and ICL GPT-3 as baselines.

More importantly, the main goal of our work is to improve the popular instruction tuning paradigm and standardize further instruction tuning on downstream tasks instead of relying on ICL on unseen tasks. To the best of our knowledge, we are the first to propose so.

2.11 Novelty Aspect

To understand the novelty aspect of this work, let us revisit the importance of ”instructions” and “instruction tuning”. Instruction tuning is an exciting concept and a novel addition to traditional supervised machine learning. In traditional supervised machine learning, models learn from training data. No additional information is given about what task they are learning. This is very different from when humans learn a task. Humans are told what the task is about (given instructions) and then, in many cases, given examples (training data) to learn. With the use of models that are ”better” at understanding language, instruction tuning emphasizes that during training in addition to the traditional input-output data in the training samples, information about the task (in the form of instructions) should also be given.

The papers Flan Wei *et al.* (2021), Natural Instructions Mishra *et al.* (2022b), Supernatural Instructions Wang *et al.* (2022c), ExT5Aribandi *et al.* (2022), and T0Sanh *et al.* (2021), show some usefulness of instruction tuning. In particular, they show that instruction tuned models are better at performing unseen tasks (but with instructions) in **zero-shot** setting.

We show a different usefulness of instruction tuning. We show that an instruction tuned model when given a new task (with instructions) is able to learn (with similar or better accuracy) with a smaller fine tuning dataset than a traditional model (without instruction tuning) when given the same new task (without instructions). In case of humans, this means that human can learn new tasks with less training examples if they are given information about the task (in terms of instruction). This seems obvious in the human setting, but has not been shown comprehensively in the ML/NLP framework.

More formally, in the earlier works there is a model M instruction tuned on tasks

T1, ... Tn. This model M is evaluated with respect to unseen tasks T1', ... Tm'. When the evaluation is done with respect to T1', ... Tm' no fine tuning is done with respect to the data items associated with T1', ... Tm'.

We make two novel claims in this paper:

- A model M instruction tuned on tasks T1, ... Tn when given unseen tasks T1', ... Tm', can achieve SOTA (where models are trained with the full training set, but do not use instructions) or even do better with respect to these tasks while being fine tuned with only a fraction of the training data in T1', ... Tm'.
- We also show that the model M instruction tuned on tasks T1, ... Tn when given unseen tasks T1', ... Tm' fine tuned with a very small fraction of the training data in T1', ... Tm', dramatically improves over the zero-shot (no-fine tuning) setting.

Table 2.1 gives an overview of the novelty aspects of the work

Our experiments are meant to simulate a real-world setting where few shot inference is not always necessary and there are some samples available for training. While there have been several studies on sample efficiency in other domains like reinforcement learning Yang *et al.* (2022c); Guo *et al.* (2022); Zhang *et al.* (2021); Yarats *et al.* (2019); Yang *et al.* (2022b); Lagani *et al.* (2021), to the best of our knowledge, no other work has explored the sample efficiency of instruction tuned models in a generalized fashion. We also provide detailed analysis and task-specific insights with respect to various instruction tuning methods.

Paper/ Family of papers	**Explore Sample Efficiency**	**Evaluation on Held-in tasks**	**Evaluation on STL and MTL Setting**	**Category Wise Analysis**	**Use Pruning Methods**
Initial Formulation	No	No	No	No	No
Enhancements on Instruction Tuning	No	No	No	No	No
Alignment	No	No	No	No	No
Benchmarks	No	No	No	No	No
PEFT	No	No	No	No	No
Prompt Engineering	No	No	No	No	No
Synthetic Data Creation	No	No	No	No	No
Ivison *et al.* (2022b)	Yes	No	No	No	Yes
Chen *et al.* (2023a)	Yes	No	No	No	Yes
Longpre *et al.* (2023)	Yes	Yes	No	No	No
Our work (Gupta *et al.* (2023a))	Yes	Yes	Yes	Yes	No

Table 2.1: Table demonstrating differences of our work with existing works. Family of papers refer to the categories that were used to demarcate a collection of similar papers. While Longpre *et al.* (2023) (concurrent work) explores sample efficiency, The experimental setup and the detailed insights are unique to our work.

Chapter 3

PROPOSED METHODOLOGY

In this chapter we describe traditional instruction tuning by starting with traditional single task learning expert modelling followed by multi task learning and finally explaining the proposed approach. We also describe different baselines used for a comprehensive evaluation of our approach.

Terminology used throughout the chapter: For each given task t, we assume that there are input and output instance pairs (X^t, Y^t). Each sample of the task is described by its instruction *inst*.

3.1 Single Task Learning/ Expert modelling

Traditional supervised models learn a mapping function (f_M) between input (x) and output (y) by using a training set of input/output pairs, $(x, y) \in (X^t_{train}, Y^t_{train})$, for a given task t. The model is then evaluated on the test set for the same task, (X^t_{test}, X^t_{test}). In the STL setup, t models are trained for t tasks in an individual fashion.

3.2 Multi Task Learning

In this setup, the training data for all tasks are combined together. The goal of MTL models is to learn a mapping function (f_M) between the input (x) and output (y), such that $f_M(x) = y$, where $(x, y) \in (X^t_{train}, Y^t_{train})$ for all t tasks in a combined way. This model is then evaluated on task-specific instances $(x, y) \in (X^t_{test}, Y^t_{test})$. In contrast to single-task models, a single model is used to solve various tasks in this setup, which allows for generalization.

Proposed Analysis	Single-Task Learning	Multi-Task Learning
Training	$Task_1 \in \left[Inst_{T1}, X^{Train}, Y^{Train}\right] \Rightarrow Model_1$ $Task_2 \in \left[Inst_{T2}, X^{Train}, Y^{Train}\right] \Rightarrow Model_2$ $\ldots.$ $Task_n \in \left[Inst_{Tn}, X^{Train}, Y^{Train}\right] \Rightarrow Model_n$	$\left.\begin{matrix} Task_1 \in \left[Inst_{T1}, X^{Train}, Y^{Train}\right] \\ Task_2 \in \left[Inst_{T2}, X^{Train}, Y^{Train}\right] \\ \ldots. \\ Task_n \in \left[Inst_{Tn}, X^{Train}, Y^{Train}\right] \end{matrix}\right\} Model$
Evaluation	$Model_1\left[X^{Test}_{Task\,1}\right] \Rightarrow Y^{Test}_{Task\,1}$ $Model_2\left[X^{Test}_{Task\,2}\right] \Rightarrow Y^{Test}_{Task\,2}$ $\ldots$ $Model_n\left[X^{Test}_{Task\,n}\right] \Rightarrow Y^{Test}_{Task\,n}$	$Model \begin{bmatrix} X^{Test}_{Task\,1} \\ X^{Test}_{Task\,2} \\ X^{Test}_{Task\,n} \end{bmatrix} \Rightarrow \begin{bmatrix} Y^{Test}_{Task\,1} \\ Y^{Test}_{Task\,2} \\ Y^{Test}_{Task\,n} \end{bmatrix}$

Figure 3.1: Formulation of the proposed analysis. In the Single-task learning (STL) setting, f_{inst} is instruction tuned individually on each downstream dataset. In the Multi-task learning (MTL) setup, all downstream training tasks are combined together, and f_{inst} is instruction tuned on all of them. In both setups, the number of input samples from the downstream train data is varied.

3.3 Instruction tuning

In this setup, the mapping function takes an instruction $inst_t$ along with the input sample to give output as y; $f_M(inst, x) = y$. Instruction tuning can be achieved in both Single-task and Multi-task setups.

Definition: The term "Definition" pertains to the detailed explanation of the task at hand along with specific instructions provided, enabling the model to successful completion of the given task.

Examples:"Examples" refer to the input/output pairs associated with a particular instance of the task. In line with the approach introduced in SuperNI, we incorporate two examples within the instruction prompts.

3.4 Proposed approach

We introduce two datasets for our task: $x_{pre-finetune}$ and x_{train}. These datasets are utilized as the pre-finetuning and downstream training datasets, respectively. By pre-finetuning an LLM f_M using instructions $(inst, x_{pre-finetune})$, we get instruction tuned model f_{inst}. f_{inst} is now instruction tuned on the downstream train data $f_{inst}(inst, x_{train}) = y$. For each experiment, different number of downstream train samples are used. The instruction prompts change according to each downstream task. For **STL** setup, f_{inst} is individually instruction tuned on all tasks of the downstream train data; t models are finetuned for t tasks. Each experiment will consist of t models instruction tuned with a different number of training samples (Column 1 of Fig. 3.1). For **MTL** setup, one dataset is prepared by combining all the tasks of the downstream train dataset together. f_{inst} is instruction tuned on the combined dataset. Similar to the last setting, the experiment will have a different number of training samples to highlight sample efficiency.

3.5 Baselines

To show a detailed analysis of instruction tuned modelling, pre-finetuning and cross-task generalization, we develop different baselines across both setups.

3.5.1 STL baselines

We develop three baselines to compare the proposed modelling paradigm comprehensively. For the first baseline, we pre-finetune the model f_M using $x_{pre-finetune}$ without instructions to get f_{M1}. f_{M1} is now individually finetuned on all t tasks of x_{train} using 5.91% of downstream train data to get $f_{STL-baseline-1}$. For the second baseline ($f_{STL-baseline-2}$), we individually finetune the f_M model on all t tasks with

25.33% of the downstream train set (1000 samples of each task) of x_{train}. We develop the third baseline ($f_{STL-baseline-3}$) by individually finetuning the model f_M on t tasks of x_{train} and use 100% of the downstream training set. Third baseline serves as the supervised SOTA. The rationale for all three baselines is two-fold: First, to compare the baselines with the proposed model with fewer training samples. Second, the baseline is also used to compare the instruction tuned model trained with the same number of samples to observe the relative improvement in performance. Both advantages can be explained through the following example: $f_{STL-baseline-1}$ can highlight the effect of pre-finetuning, demonstrate sample efficiency and can be compared for performance improvement when the same number of samples are used.

3.5.2 MTL baselines

We develop two baselines to compare with the proposed MTL instruction tuned modelling paradigm. For the first baseline, we finetune f_M on 25.33% of the downstream train set x_{train} in MTL setup to get $f_{MTL-baseline-1}$. For developing the second baseline, we pre-finetune the model f_M using $x_{pre-finetune}$ without instructions to get f_{M1}. f_{M1} model is finetuned in MTL setup on all t tasks using 5.91% of the downstream train data (200 samples of each task) of x_{train}. The rationale for creating two MTL baselines is similar to STL baselines; to show sample efficiency, highlight performance improvement when using the same number of samples, and showcase the effect of instructions in pre-finetuning.

3.5.3 Models and Evaluation Metrics

Models: We use Tk-Instruct 3B as the instruction tuned model. For STL setup 952 models (119x8) were trained and 9 models were trained for MTL setup resulting in a total of **961** models for our analysis. All the models were trained on 6x Nvidia

A100 40GB GPUs.

3.5.4 Choosing T5-3B as the model for experiments

The rationale behind this choice is to ensure fair comparison with the SuperNI paper, especially the state-of-the-art (SOTA) model in its corresponding leaderboard since we use the same experimental setup as SuperNI. The SuperNI paper established a state-of-the-art (SOTA) performance benchmark using T5-3B (akin to MTL baseline 2 in our study). Consequently, our decision to opt for Tk-Instruct 3B was a natural extension of their work, ensuring consistent and coherent comparisons. We would like to highlight that our extensive analysis in this paper contains instruction tuning of 961 3B models (L196) involving a large investment of computational resources (note that our benchmark SuperNI is a heavy dataset collection with 1600+ tasks containing 900 english tasks). Expanding our investigation to larger and different architecture models remained beyond the scope of our computational resources. Furthermore, our selection of a 3B model is supported by other instruction tuning papers such as Longpre *et al.* (2023); Liu *et al.* (2022).

3.5.5 Models and Evaluation Metrics

Models: We use Tk-Instruct 3B as the instruction tuned model. For STL setup 952 models (119x8) were trained and 9 models were trained for MTL setup resulting in a total of **961** models for our analysis. All the models were trained on 6x Nvidia A100 40GB GPUs.

Evaluation metric: We consider all the tasks in the dataset as text generation problems and use the ROUGE-L score Lin (2004) to evaluate the generated outputs.

Chapter 4

DATASET

In this chapter we detail about the datasets used in the experiments.

We use seen tasks of SuperNI as the pre-finetuning set consisting of 757 tasks with 100 samples for each task. We used the unseen task set of SuperNI as the downstream train data, consisting of 119 tasks that could be classified into 11 categories. Their statistics are presented in Table 4.2. The Dataset is classified into 11 categories of NLP tasks as shown in Table 4.1. For the sake of clarity, we have clubbed seven categories with fewer tasks into the *others* category. Table 4.3 gives detailed statistics across each category. Since not all the tasks have exactly the same number of samples, we choose the maximum number available if the number of samples is below the threshold. We use 191 samples when finetuning with 200, 1000, and the entire dataset.

4.1 Task Descriptions

4.1.1 Answerability Classification:

Answerability classification is a natural language processing (NLP) task that involves determining whether a given text contains a question that can be answered. This task can be useful in a variety of applications, such as chatbots or information retrieval systems, where it is important to know whether a user's input is a question that can be answered by the system. To perform answerability classification, an NLP model must first be trained on a dataset of texts labeled as either "answerable" or "unanswerable." The model can then be used to classify new texts as either answerable or unanswerable based on their similarity to the texts in the training dataset. The paragraph below gives an example of this category.

Category	Count	Category	Count
Textual Entailment	24	Word Analogy	8
Title Generation	18	Cause Effect Classification	7
Coreference Resolution	14	Dialogue Act Recognition	7
Answerability Classification	13	Keyword Tagging	5
Question Rewriting	11	Overlap Extraction	2
Data to Text	9	Grammar Error Correction	1

Table 4.1: Training sample statistics

Statistics	
# Total Tasks	119
# Total instances in train set	374745
# Total instances in Test Set	11810
# Total instances in pre-finetuning set	75700
Avg len of Train data w instructions	364.97
Avg len of Train data w/o instructions	89.03

Table 4.2: Statistics of the SuperNI dataset. Train and Test set refers to Downstream data.

“***Definition:*** *The answer will be 'yes' if the provided sentence contains an explicit mention that answers the given question. Otherwise, the answer should be 'no'. Instances where the answer is implied from the sentence using ""instinct"" or ""com-*

	Answerability Classification	Coreference Resolution	Data to Text	Question Rewriting	Textual Entailment	Title Generation	Other Categories	Grand Total and percentage
# of tasks	13	14	9	11	24	18	30	**119 (100%)**
100	1300	1370	826	949	2376	1784	2994	**11.5K (3.09%)**
200	2600	2441	1626	1849	4457	3484	5708	**22.1K (5.91%)**
1000	11529	8831	8026	9049	19019	16514	21988	**94.9K (25.33%)**
All	43871	35560	36815	41391	78802	78357	59949	**374.7K (100%)**

Table 4.3: Category-wise statistics of the downstream train data used. We note that since all the tasks have unequal samples, the total samples in each category will be different than # of Tasks*Number of samples. Rows 100, 200, 1000, and all samples represent the sum of the number of samples chosen during each experiment.

mon sense"" (as opposed to being written explicitly in the sentence) should be labeled as 'no'.

Example 1 *Input: Sentence: Jack played basketball for an hour after school, after which he was very tired*

Question: How long did Jack play basketball?

Output: Yes

Example 2 *Input: Sentence: He was born in China, so he went to the Embassy at 1 pm to apply for a U.S. Visa.*

Question: When did he go to Embassy?

Output: Yes Input Now complete the following example- ***Input:*** *Sentence: The Vice President's guidance was we need to take them out. Question: Has he always wanted to take them out? Output: No*"

4.1.2 Coreference Resolution:

Coreference resolution is a natural language processing (NLP) task that involves identifying and linking mentions of the same real-world entities in text. This task is important for understanding the meaning and context of text, as it allows a system to determine that multiple mentions of a word or phrase in a document refer to the same entity. For example, consider the following text: "John went to the store to buy some milk. He needed it for his cereal." In this text, the pronouns "he" and "his" refer to the same person, "John." A coreference resolution system would identify these pronouns as referring to the same entity and link them to the proper noun "John." The paragraph below gives an example of this category.

"***Definition*** *You need to answer a given question containing a blank (). Your answer must be one of the two objects mentioned in the question, for example ""trophy"" and ""suitcase"". Your answer must not contain a word that is not present in the question. Please don't use articles (e.g., the, a) before the answer.*
Example 1: *The trophy doesn't fit into the brown suitcase because _ is too large.*
Output: trophy
Example 2: *Grace was happy to trade me her sweater for my jacket. She thinks _ looks dowdy on her.*
Output: sweater
Input Now complete the following example- ***Input:*** *The goldfish were finally removed from the bag and transferred into the tank, as the _ was a temporary home for them.*
Output: bag"

4.1.3 Data-to-text:

Data-to-text generation is a natural language processing (NLP) task that involves automatically generating human-readable text from structured data. This task can be useful in a variety of applications, such as automated report generation or data summarization, where it is important to present data in a clear and concise manner. An example of data-to-text generation is generating a weather report from data about the current temperature, humidity, and forecast for a particular location. The data might include the following: Temperature: 75 degrees Fahrenheit Humidity: 50% Forecast: sunny A data-to-text generation system could use this data to generate the following text: "The current temperature is 75 degrees Fahrenheit and the humidity is 50%. The forecast for today is sunny." The paragraph below gives an example of this category.

"***Definition*** *In this task, you are given concept set (with 3 to 5 concepts) that contain mentions of names of people, places, activities, or things. These concept sets reflect reasonable concept co-occurrences in everyday situations. All concepts given as input are separated by #. Your job is to generate a sentence describing a day to day scene using all concepts from a given concept set.*
Example 1 *Input: mountain#ski#skier*
Output: Skier skis down the mountain
Example 2 *Input: call#character#contain#wallpaper*
Output: queen of wallpaper containing a portrait called film character.
Input Now complete the following example-
Input: *lake#shore#walk*
Output: Men walk along the shore of the lake"

4.1.4 Textual Entailment:

Textual entailment is a natural language processing task that involves determining the relationship between two text passages. Specifically, it involves determining whether one passage, called the "hypothesis," can be inferred from the other passage, called the "premise." For example: Premise: "The cat is sitting on the couch." Hypothesis: "There is a cat on the couch." In this case, the hypothesis can be inferred from the premise, so the textual entailment relationship is "entailment." The paragraph below gives an example of this category.

“***Definition*** *Definition: In this task, you’re given two sentences. Indicate if the first sentence clearly entails the second sentence (i.e., one can conclude the 2nd sentence by reading the 1st one)Indicate your answer with ’1’ if the first sentence entails the second sentence, otherwise answer with ’0’.*
Example 1 *Input: Sentence 1: No Weapons of Mass Destruction Found in Iraq Yet.*
Sentence 2: Weapons of Mass Destruction Found in Iraq.
Output: 0
Example 2 *Input: Sentence 1: A place of sorrow, after Pope John Paul II died, became a place of celebration, as Roman Catholic faithful gathered in downtown Chicago to mark the installation of new Pope Benedict XVI.*
Sentence 2: Pope Benedict XVI is the new leader of the Roman Catholic Church.
Output: 1
Input Now complete the following example-
Input: *Sentence 1: Since 1987, however, Brazil has taken steps to dramatically reduce the destruction, including stepped-up enforcement and the elimination of tax incentives that led to large-scale land clearing. Sentence 2: In the early 1990s Brazil began to take action to save the rainforest.*
Output: 0”

4.1.5 Question Rewriting:

Question Rewriting is a natural language processing (NLP) task that involves generating a new version of a given question that has the same meaning as the original, but is phrased differently. For example, given the question ”What is the capital of France?”, a question rewriting task might generate the following rephrased question: ”Where is the seat of government for France located?”. The paragraph below gives an example of this category.

"***Definition*** *Given a disfluent sentence, modify the sentence to it to its equivalent fluent form, preserving the meaning of the sentence.*
Example 1 *Input: Who did the Han Chinese want to help the Khitan no I mean the Mongols fight?*
Output: Who did the Han Chinese want to help the Mongols fight?
Example 2 *Input: What part did no I meant how many chapters have coordinating lead authors?*
Output: How many chapters have coordinating lead authors?
Input Now complete the following example-
Input: *What year did a plague-ridden ship land in Norway?*
Output: When did a plague-ridden ship land in Norway?"

4.1.6 Title Generation:

Title generation is a natural language processing (NLP) task that involves creating a title for a given text or topic. This task is often used in content creation and marketing, where an eye-catching title is essential for attracting attention and engaging readers. For example, a title generation task might involve creating a title for an article about the benefits of meditation. Some possible titles might be "5 Reasons Why Meditation is the Key to a Stress-Free Life," "Discover the Surprising Benefits of Meditation," or "Meditation: The Ultimate Tool for Relaxation and Mindfulness." The goal of the title generation task is to generate a title that accurately reflects the content of the article and is compelling enough to encourage readers to click and read more. The paragraph below gives an example of this category.

"***Definition*** *In this task, you're given a paragraph from the research paper and your task is to generate a suitable title for the research paper based on the given paper.*

Under 100 words is a good title length.

Example 1 *Input: The severe acute respiratory syndrome (SARS) epidemic originating from China in 2002 was caused by a previously uncharacterized coronavirus that could be identified by specific RT-PCR amplification. Efforts to control future SARS outbreaks depend on the accurate and early identification of SARS-CoV infected patients. A real-time fluorogenic RT-PCR assay based on the 3-noncoding region (3 -NCR) of SARS-CoV genome was developed as a quantitative SARS diagnostic tool. The ideal amplification efficiency of a sensitive SARS-CoV RT-PCR assay should yield an E value (PCR product concentration increase per amplification cycle) equal to 2.0. It was demonstrated that the 3 -NCR SARS-CoV based RT-PCR reactions could be formulated to reach excellent E values of 1.81, or 91% amplification efficacy. The SARS-CoV cDNA preparations derived from viral RNA extract and the cloned recombinant plasmid both exhibit the identical amplification characteristics, i.e. amplification efficacy using the same PCR formulation developed in this study. The viral genomic copy (or genomic equivalences, GE) per infectious unit (GE/pfu) of SARS-CoV used in this study was also established to be approximate 1200-1600:1. The assay's detection sensitivity could reach 0.005 pfu or 6-8 GE per assay. It was preliminarily demonstrated that the assay could efficiently detect SARS-CoV from clinical specimens of SARS probable and suspected patients identified in Taiwan. The 3 -NCR based SARS-CoV assay demonstrated 100% diagnostic specificity testing samples of patients with acute respiratory disease from a non-SARS epidemic region.*

Output: NHS Wales: Court action if trade deals affect service?

Example 2 *Input: By Jon Welch and Paul MoseleyBBC News Details of health problems, family bereavements and personal issues were sent by the University of East Anglia (UEA) in Norwich to 298 students. Megan Baynes, 23, said she felt ""sick and horrified"" when she realised her details had been shared. The UEA apologised* *""unreservedly"" and said an inquiry had begun. The email contained a spreadsheet listing 172 names and details extenuating circumstances in which extensions and other academic concessions were granted to 42 students. 'Felt sick' It was sent to nearly 300 undergraduates, including Ms Baynes, a former editor of student newspaper Concrete. She is currently awaiting the results of her American Literature and Creative Writing degree, and had been granted extensions for coursework because of an illness suffered by a family member. ""I felt sick at seeing my personal situation written in a spreadsheet, and then seemingly sent to everyone on my course,"" she said. ""My situation was not the worst on there but there are some on there that are so personal. There are people I know and I feel so awful for them and can't imagine how they are feeling."" Theo Antoniou Phillips, UEA Students' Union undergraduate education officer, said: ""This is a shocking and utterly unacceptable data breach that should never have happened."" Jo Swo, the union's welfare, community and diversity officer, said: ""Given the university is supposed to be making mental health a priority, this is a real slap in the face to students who have sought support."" In a statement, a UEA spokeswoman said: ""An email was mistakenly sent to 298 American Studies undergraduates this morning containing details of 42 students with extenuating circumstances. "" This clearly should not have happened and the university apologises unreservedly. The university has launched an urgent enquiry and is contacting all affected students to offer support. ""Anyone needing support should call 01603 592761. The university is informing the ICO (Information Commissioner's Office)."" The ICO has been contacted for comment.*

Output: University of East Anglia in students' personal data breach

Input Now complete the following example-

***Input:** President Donald Trump said Mr Mnuchin had spent his career making money in the private sector and would now work for the taxpayer. Mr Mnuchin pledged to cre-*

ate jobs and combat terrorist financing. Democrats had argued that Mr Mnuchin had made a fortune foreclosing on families during the financial crisis. The top Democrat on the House Financial Services Committee, Maxine Waters of California, called Mr Mnuchin ""the foreclosure king"". His critics have also questioned whether he is too close to the Wall Street banking community, which he will be responsible for regulating. Democrats also complained that Mr Mnuchin had failed to disclose nearly $100m (£79m) in assets when he filed with the Senate Finance Committee. Mr Mnuchin spent 17 years at Goldman Sachs before becoming a hedge fund manager. He later founded a film production company that was behind such box office hits as the X-Men franchise and American Sniper. Mr Trump said Mr Mnuchin would help make the US a ""jobs magnet"". ""He'll work 24 hours a day, I know him. He'll work 28 hours a day if they give him the extra four hours,"" he said. Another former Goldman executive, Gary Cohn, is the director of President Trump's National Economic Council. What do we know about the new treasury secretary's policy plans? Mr Mnuchin hasn't announced a fully fledged plan, but his responses in media interviews and during the Senate debate over his appointment make clear some of his priorities: There are still many policy areas that have not been addressed, including how he will approach trading relations with China, Mexico and other partners.
Output: Trump says Mnuchin will fight for tax cuts and jobs"

4.1.7 Keyword Tagging:

Keyword tagging is the process of assigning specific keywords or labels to a piece of text or document. This task is often used in natural language processing (NLP) to help classify and organize large amounts of text data for various purposes, such as search engines, topic modeling, and sentiment analysis. For example, consider a news article about recent political events in the United States. Keyword tagging for this article might include labels such as ”politics,” ”US politics,” ”election,” ”government,” and ”political parties.” These tags can help identify the main themes and topics discussed in the article, making it easier for users to search for and find similar articles on the same topics.

4.1.8 Overlap Extraction:

Overlap extraction is a natural language processing (NLP) task that involves extracting overlapping text or data from multiple sources. This can be useful for a variety of purposes, such as identifying common themes in different documents, comparing and contrast information, or finding duplicates in a dataset. For example, consider a scenario where you have two news articles discussing the same topic. You might use overlap extraction to identify the common themes or ideas discussed in both articles, such as the main events, people involved, or key quotes. This could help you understand the overall coverage of the topic and identify any discrepancies or differences in the way it was presented by the two sources.

4.1.9 Word Analogy:

Word analogy is a natural language processing task that involves identifying relationships between words based on their meanings and contexts. The goal is to find a word that is similar to another word in a specific way, based on the relationship between the two words. For example, if the task is to find a word that is similar to ”man” in the same way that ”woman” is similar to ”man,” the correct answer would be ”wife.” The relationship between the words ”man” and ”wife” is that they are both terms for a specific type of spouse, with ”man” being the term for a husband and ”wife” being the term for a wife.

4.1.10 Cause Effect Classification:

Cause-effect classification is a natural language processing (NLP) task that involves determining the causal relationships between events or actions described in text. This task can be useful in a variety of applications, such as information extraction and text summarization, where it is important to understand the underlying causes and effects of events described in text. Example: Consider the following two sentences: "The car wouldn't start because the battery was dead." "The child was crying because he fell and skinned his knee." In the first sentence, the cause is "the battery was dead," and the effect is "the car wouldn't start." In the second sentence, the cause is "he fell and skinned his knee," and the effect is "the child was crying."

4.1.11 Dialogue Act Recognition:

Dialogue act recognition is a natural language processing (NLP) task that involves identifying the purpose or intention behind a speaker's words in a conversation. This task can be useful in a variety of applications, such as chatbots or virtual assistants, where it is important to understand the intent behind a user's input in order to respond appropriately. An example of dialogue act recognition is identifying the intent behind the following statement: "Can you pass the salt?" The dialogue act in this statement might be classified as a request, as the speaker is asking the listener to perform an action.

4.1.12 Grammar Error Correction:

Grammar error correction is a natural language processing (NLP) task that involves identifying and correcting grammatical errors in a given text. An example of a sentence with a grammatical error that could be corrected as part of this task is: "I went to the stores to buy some food and clothes." This sentence contains the grammatical error of using the wrong form of the word "store." The correct form should be "store," which is singular, as in "I went to the store to buy some food and clothes."

Dataset Split: Hundred samples per task have been used for testing, and ten samples have been used for validation. These samples are not a part of the training set, and we have ensured that there is no data leakage. Each task contains the same number of samples for the test (100 samples) and validation set (10 samples).

4.1.13 Baselines

We use $x_{pre-finetune}$ and x_{train} in both STL and MTL setups. For creating baselines, both datasets are not equipped with instructions; they contain just input and output for conventional finetuning.

Single Task Learning Baselines

STL Baseline 1: A T5-3B model undergoes pre-finetuning using $x_{pre-finetune}$, which consists of 757 tasks from SuperNI. Subsequently, the model undergoes further finetuning on 200 samples per task from the downstream train data of SuperNI (x_{train}), resulting in 119 models.

STL Baseline 2: Each task from the downstream train data (x_{train}) is used to finetune a T5-3B model (f_M) with 1000 samples per task, resulting in 119 distinct models.

STL Baseline 3 (Supervised SOTA): A T5-3B model (f_M) is finetuned for each task using all available samples from the downstream train data.

Multi Task Learning Baselines

MTL Baseline 1: Similar to STL Baseline 1, we prefinetune a T5-3B $x_{pre-finetune}$ for each task to get the model f_{M1}. f_{M1} is now finetuned on 200 samples per task from the downstream train data (x_{train}) of SuperNI (in an MTL fashion).

MTL Baseline 2: T5-3B model is now finetuned on 1000 samples per task from the downstream train data (x_{train}) of SuperNI (in an MTL fashion).

Chapter 5

RESULTS AND ANALYSIS

In this chapter, we describe single task and multi task baselines results followed by results obtained using our approach. We also compare and and do a category wise analysis of the same.

5.1 Results

The results are presented in two parts, STL and MTL results. Each section contains overall results, Category wise results, and a comparison with the baselines that were defined earlier.

5.1.1 Single Task Learning setup (STL)

Fig. 5.1 shows the rouge score of instruction tuned models when training with different numbers of samples. We see that there is an overall increasing trend as the number of samples increases. From the figure we observe that a max ROUGE-L score of 72.04 is obtained when all samples are used. Figure 5.2 shows the category wise results of the instruction tuned models. From the figure, we see that except for the Answerability Classification category, all the categories have an increasing trend.

50% efficient w.r.t STL baseline 1: STL Baseline 1 is denoted by the red point in Figure 5.1. The score with 100 samples is 65.93, and the score with baseline 1 is 64.29.

Competitive performance using 6% data: STL Baseline 2 is denoted by the yellow point in Figure 5.1). The instruction tuned model uses roughly 23.33% of training samples when trained on 6% data compared to STL baseline 2 which uses

	Answerability Classification	**Coreference Resolution**	**Data to Text**	**Question Rewriting**	**Textual Entailment**	**Title Generation**	**Other Categories**
STL Baseline 1	70.38	70.01	49.75	68.12	71.00	44.71	72.31
STL Baseline 2	78.77	68.61	50.89	71.00	77.78	46.78	75.62
STL Baseline 3	80.36	74.40	52.84	71.11	80.42	48.35	76.82

Table 5.1: STL category wise scores for all three baselines. We see that all the baselines follow a linearly increasing trend as the number of samples increases with baselines 1, 2, and 3 (200, 1000, and all samples used, respectively). However, little improvement is observed in the Question rewriting category w.r.t baseline 2 and 3's ROUGE-L score (71.00 and 71.11, respectively). Another deviation from the standard trend was observed in the Coreference Resolution category where STL baseline 1 had a higher score as compared to STL baseline 2 (70.01 and 68.61 respectively).

25.33% data. The score with 200 samples/task is 68.34, while the score with baseline 2 is 68.91. In comparison with STL baseline 1 with the instruction tuned model (both trained using the 6% data), an increase of $\sim 3\%$ is observed.

Surpassing SOTA with 25% data: The instruction tuned model uses 25.33% of the data compared to STL baseline 3 and gets a score of 71.71, compared to the 70.99 score of the baseline. Comparison with baseline 2 (both trained using 25.33% of the data) yields an increment of 3%. When all samples are used, there is a further increase of 1.04% (72.04 vs 70.99).

Category wise effect of instruction tuning: We observe that Answerability Classification and Title Generation categories scores decrease from instruction tuning as compared to baselines. The best scores from instruction tuning are 75.15 and 44.55 respectively which are significantly lower than the best baseline scores of 80.36 and 48.35. The categories that benefit from instruction tuning compared to the baselines are Coreference Resolution (82.82 vs 74.40) and Data to Text (59.06 vs 52.84).

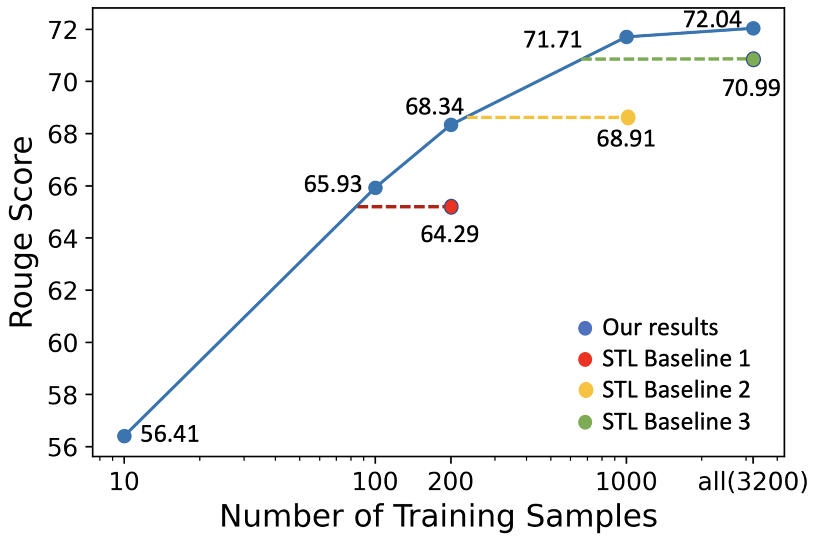

Figure 5.1: Results in the STL setup. and 3, respectively. The horizontal dashed line is marked on the graph to highlight the difference in train data required between the proposed approach and baselines. x-axis is in logarithmic scale.

5.1.2 Multi task setup

Figure 5.3 shows the overall ROUGE-L score of instruction tuned models in the MTL setup. A max ROUGE-L score of 74.68 is obtained when all samples are used, surpassing the SOTA of 70.99. Figure 5.4 shows the category wise results of the instruction tuned models and Table 5.2 showcases the baseline results.

50% efficient compared to MTL Baseline 1: MTL baseline 1 was trained on roughly 6% of downstream train samples. The score with 3% downstream data samples is 66.78, while the score with baseline 1 is 65.63. If we compare the instruction tuned model trained 6% of downstream train samples, there is an increase of roughly 5% points as it reaches a 70.40 rouge score.

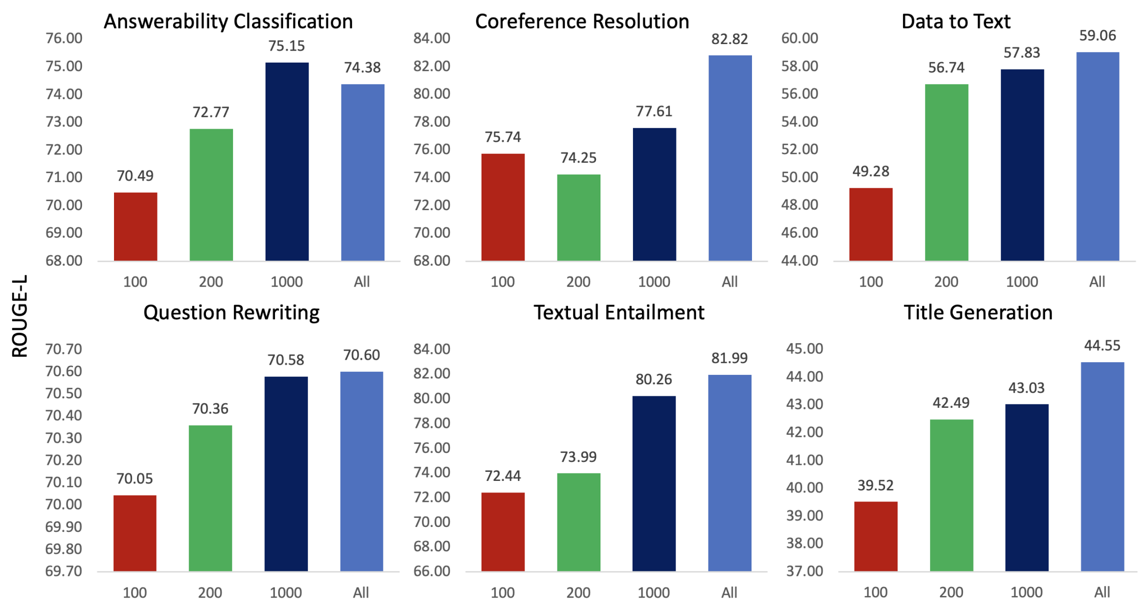

Figure 5.2: Histogram showing category wise results of the proposed approach in **STL** setting. The x-axis shows avg number of training samples. the y-axis shows the rouge-L scores. Most categories follow a conventional trend of performance increase as the number of training samples increases. This trend has an exception in two places. First: Answerability classification score drops when all samples are used after 1000 (75.15 to 74.38). Second: Coreference Resolution score drops when 200 samples are used after 100 (75.74 to 74.25).

	Answerability Classification	**Coreference Resolution**	**Data to Text**	**Question Rewriting**	**Textual Entailment**	**Title Generation**	**Other Categories**
MTL Baseline 1	75.35	75.87	55.85	75.64	65.95	62.19	45.20
MTL Baseline 2	76.89	81.93	54.73	79.80	67.89	67.18	42.19

Table 5.2: MTL baseline category-wise scores. All categories follow an increasing trend as conventional thinking would suggest. The trend is however broken in the Data to Text category and other categories.

Surpassing SOTA with 6% train data: MTL baseline two is denoted by the yellow point in Figure 5.1, and was trained using 25% of downstream train samples.

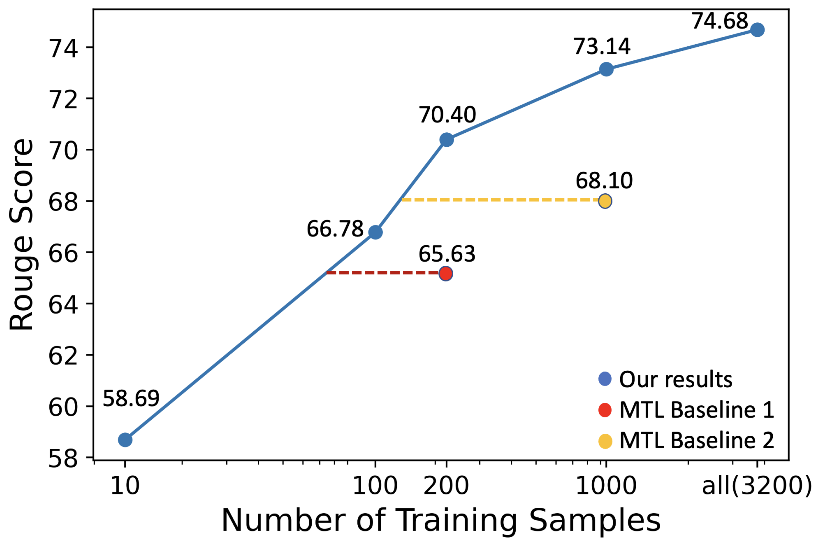

Figure 5.3: Proposed model overall results in MTL setup. The Red and Yellow dots represent MTL Baselines 1 and 2, respectively. The score gap between the proposed approach and the baseline widens as compared to the STL setup.

The instruction tuned model uses $\sim$ 76% fewer samples when trained on 6% of downstream train samples and gets a score of 70.40, while the score with baseline 2 is 68.10. The instruction tuned approach, trained on the same samples as baseline 2, improves by roughly 5% (73.14 vs. 68.10). When all samples are used, a score of 74.68 is obtained, surpassing SOTA by 3%.

Category wise effect of instruction tuning: Contrasting results to STL settings are observed as Question Rewriting and Title Generation categories experience a significant drop (12% and 23% points respectively) as compared to the best baseline scores. There is a significant improvement observed in the Textual Entailment category as the best score improves to 84.16 from 67.89 as compared to the baseline score.

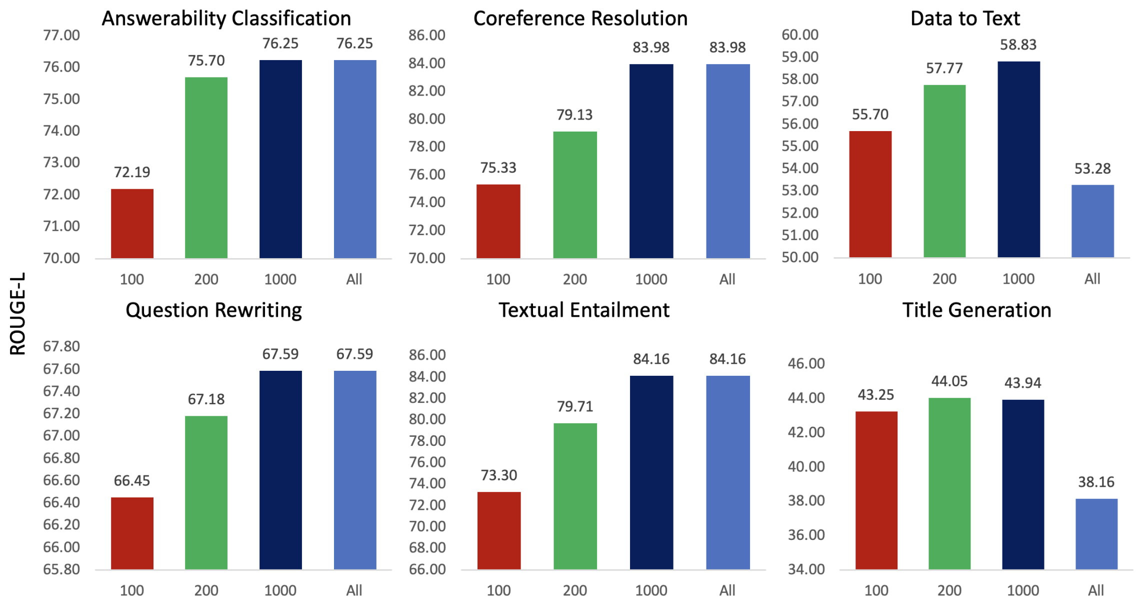

Figure 5.4: Histogram showing category wise results of the proposed approach in the MTL setting. Similar to STL category wise scores, a linear trend is followed but has two exceptions. First: Data to text score drops when all samples are used after 1000 (58.83 to 53.28). Second: Title Generation score drops when all samples are used after 1000 (43.94 to 38.16)

5.2 Analysis

5.2.1 Category Wise Analysis

We analyze the performance across each category in both settings. In the STL setting, we find that the tasks belonging to the coreference resolution and data to text category have a high increase in ROUGE-L score with instruction tuning as compared to baseline approaches (78.23 vs. 71.00 ROUGE-L in coreference resolution and 57.88 vs. 51.16 in Data to Text). Question rewriting performed nearly the same (70.51 vs. 70.07 ROUGE-L) while answerability classification and title generation's score decreased w.r.t baseline (74.10 vs. 76.50 ROUGE-L in answerability classification and

43.36 vs. 46.61 in title generation). In the **MTL** setup, similar findings are observed but across different categories. We find that the tasks belonging to the textual entailment category have the highest increase with instruction tuning compared to the baseline (81.93 vs. 66.91 ROUGE-L). Answerability classification performed nearly the same (75.97 vs. 76.11 ROUGE-L) while question rewriting and title generation's score decreased w.r.t baseline (67.38 vs. 77.71 ROUGE-L in question rewriting and 43.99 vs. 64.68 in title generation).

5.2.2 MTL consistently outperforming STL:

We have performed multiple experiments across instruction tuned modelling settings while keeping the number of training samples the same across different settings. Across each training setup, there is an increase of 1-2% ROUGE-L in MTL setup as compared to STL. Through both settings and all the experiments, it was evident that instruction tuned models perform better in the multi-task setup as compared to the single-task setup.

5.2.3 Sample Efficiency:

Instruction tuned models showcase sample efficiency across both MTL and STL setups. Using multiple baselines, sample efficiency of roughly 50, 75, and 80% are achieved across different spaces in both STL and MTL setups. We also see that when all samples are used in an instruction tuned setting, the overall performance beats SOTA.

5.2.4 Effect of Instructions in pre-finetuning:

STL Baseline 1 and MTL Baseline 1 were pre-finetuned with 757 tasks of the SupperNI dataset but without instructions. They were later finetuned on 119 tasks downstream train data using 6% in STL and MTL fashion. Instructions have a significant effect in pretraining as the instruction tuned model outperformed these baselines by 4 and 5%, respectively, when trained with the same number of samples.

5.2.5 Case Study on Negative results

In title generation tasks (e.g., Task 602, 1356, and 1540), extensive token lengths (1192, 1256, and 781 tokens respectively) exceeding the model's limit (512 tokens) compromised performance (Task 602: 0.01 vs 0.38, Task 1356: 0.03 vs 0.33 and Task 1540: 0.27 vs 0.40). Similarly, in Answerability Classification (Task 233), a lengthy token length (648) adversely impacted its scores (0.63 vs 0.99). Additionally, instruction bias in Task 242 hindered results (0.63 vs 0.99), but substituting a negative example in the instruction prompt significantly improved performance (ROUGE-L score with negative example: 1.00).

5.2.6 Differences of our work from Flan collection and ExT5

We would like to clarify the differences between the experimental settings proposed by our work and those of the aforementioned works - especially where FlanLongpre *et al.* (2023) collection and ExT5Aribandi *et al.* (2022) evaluate on unseen tasks. We refer to the following extracts from Flan and ExT5 respectively. While the flan collection has some similar experiments to ours with their held in tasks, we highlight two important distinctions from their work:

- We conduct our experiments on 119 datasets across 12 categories which show broad applicability of the findings as opposed to just 8 held in tasks.
- We establish multiple strong baselines in both STL and MTL setups, using which we explore the effect of sample efficiency, pre fine-tuning with instructions

and comparison with supervised SOTA scores. Flan used T5-XL-Lm as their baseline.

- Our work simulates a low resource scenario where there are very few labeled samples from unseen tasks available for fine tuning and we aim to extract maximum performance via those samples. We use the phrase “sample efficiency” to refer to the efficient use of available samples.

Chapter 6

CONCLUSION

6.1 Summary

In this study, we have taken a significant step forward in advancing the instruction paradigm by incorporating a small portion of training data commonly available for downstream tasks. By instruction tuning models on the small-scale training sets of downstream tasks, we have observed notable performance benefits for the Tk-instruct model on SuperNI. These findings suggest that instruction tuning can effectively assist a model in quickly learning a task even with limited data.

6.2 Conclusion

The study clearly demonstrates that instruction tuned models like Tk-Instruct can achieve very competitive performance on unseen downstream NLP tasks using just a fraction of the typical training data. For example, in single task learning, Tk-Instruct was able to match the performance of supervised SOTA models using only 25% of the available downstream training data. When allowed to train on the full dataset, it exceeded SOTA performance by over 1%. Similarly impressive results were achieved in the multi-task learning setting. With just 6% of downstream data, Tk-Instruct was able to match SOTA. When trained on the complete dataset, it exceeded SOTA by a sizable margin of 3.7%. The authors conducted a detailed analysis of the sample efficiency advantages of instruction tuning compared to baselines without it. Across various data amounts, the instruction tuned models were 50-80% more efficient in terms of the training data needed to reach a given performance level. An important finding was that pre-finetuning the models with instructions, before any downstream training, significantly improved transfer learning performance. Models pre-finetuned this way achieved 3-5% higher scores compared to those without instruction pre-finetuning. Additionally, the study found that multi-task learning consistently outperformed single task learning by 1-2% when using comparable training data amounts. This aligns with the expected benefits of multi-task learning. However, the advantages of instruction tuning were not consistent across all tasks. It helped most significantly for textual entailment tasks but actually hurt performance on question rewriting and title generation.

6.3 Future Work

The results obtained are specific to the T5-3B model and SuperNI dataset used. Additional experiments should be conducted with larger models such as T5-11B and more diverse datasets beyond SuperNI. This will provide a more robust evaluation of the generalizability of the findings. The study explored performance for up to 100% of the available downstream training data. Further experiments could analyze how performance continues to scale as even more data is added for training. While SuperNI covers a wide range of tasks, evaluating on additional NLP task categories can help strengthen the conclusions regarding sample efficiency. Further analysis is needed to understand why certain tasks like question rewriting did not benefit from instruction tuning. Mitigation techniques can then be developed to handle such cases. Instruction tuning could be combined with other techniques like prompt-based tuning and learning from demonstrations to study any synergistic effects. The approach should be evaluated in low-resource scenarios like multilingual learning, dialect adaptation, and domain-specific language tasks where sample efficiency is critical. Methods to automate the discovery of optimal instructions for new unseen tasks can help unlock the full potential of instruction tuning. The sample efficiency of instruction tuning should be compared head-to-head against other transfer learning techniques on multiple tasks. Drawing connections with human learning can provide insights into improvements in instruction tuning and related techniques for few-shot learning.

REFERENCES

Allamanis, M., M. Brockschmidt and M. Khademi, "Learning to represent programs with graphs", arXiv preprint arXiv:1711.00740 (2017).

Alon, U., M. Zilberstein, O. Levy and E. Yahav, "code2vec: learning distributed representations of code", Proceedings of the ACM on Programming Languages **3**, 1 – 29 (2018).

Anantheswaran, U., H. Gupta, M. Parmar, K. K. Pal and C. Baral, "Edm3: Event detection as multi-task text generation", arXiv preprint arXiv:2305.16357 (2023).

Anderson, R. R., "Vision encoders in visual question answering", (2022).

Aribandi, V., Y. Tay, T. Schuster, J. Rao, H. S. Zheng, S. V. Mehta, H. Zhuang, V. Q. Tran, D. Bahri, J. Ni, J. Gupta, K. Hui, S. Ruder and D. Metzler, "Ext5: Towards extreme multi-task scaling for transfer learning", in "International Conference on Learning Representations", (2022), URL `https://api.semanticscholar.org/CorpusID:260421295`.

Arunkumar, A., S. Mishra, B. Sachdeva, C. Baral and C. Bryan, "Real-time visual feedback to guide benchmark creation: A human-and-metric-in-the-loop workflow", arXiv preprint arXiv:2302.04434 (2023a).

Arunkumar, A., S. Sharma, R. Agrawal, S. Chandrasekaran and C. Bryan, "Lingo : Visually debiasing natural language instructions to support task diversity", Computer Graphics Forum **42**, URL `https://api.semanticscholar.org/CorpusID:258107885` (2023b).

Asai, A., T. Schick, P. Lewis, X. Chen, G. Izacard, S. Riedel, H. Hajishirzi and W. tau Yih, "Task-aware retrieval with instructions", in "Annual Meeting of the Association for Computational Linguistics", (2022), URL `https://api.semanticscholar.org/CorpusID:253581733`.

Bai, Y., A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan *et al.*, "Training a helpful and harmless assistant with reinforcement learning from human feedback", arXiv preprint arXiv:2204.05862 (2022a).

Bai, Y., S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon *et al.*, "Constitutional ai: Harmlessness from ai feedback", arXiv preprint arXiv:2212.08073 (2022b).

Balog, M., A. L. Gaunt, M. Brockschmidt, S. Nowozin and D. Tarlow, "Deepcoder: Learning to write programs", arXiv preprint arXiv:1611.01989 (2016).

Chen, H., Y. Zhang, Q. Zhang, H. Yang, X. Hu, X. Ma, Y. YangGong and J. J. Zhao, "Maybe only 0.5% data is needed: A preliminary exploration of low training data instruction tuning", ArXiv **abs/2305.09246**, URL `https://api.semanticscholar.org/CorpusID:258715090` (2023a).

Chen, J., Y. Zhang, L. Liu, R. Dong, X. Chen, P. Ng, W. Y. Wang and Z. Huang, “Improving cross-task generalization of unified table-to-text models with compositional task configurations”, arXiv preprint arXiv:2212.08780 (2022).

Chen, L., J. Chen, T. Goldstein, H. Huang and T. Zhou, “Instructzero: Efficient instruction optimization for black-box large language models”, ArXiv **abs/2306.03082**, URL `https://api.semanticscholar.org/CorpusID:259075794` (2023b).

Chen, S., K. Crammer, H. He, D. Roth and W. J. Su, “Weighted training for cross-task learning”, in “International Conference on Learning Representations”, (2021a).

Chen, X., X. Xie, N. Zhang, J. Yan, S. Deng, C. Tan, F. Huang, L. Si and H. Chen, “Adaprompt: Adaptive prompt-based finetuning for relation extraction”, (2021b), URL `https://api.semanticscholar.org/CorpusID:233240688`.

Chung, H. W., L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma *et al.*, “Scaling instruction-finetuned language models”, arXiv preprint arXiv:2210.11416 (2022).

Das, S., “Contextual code completion using machine learning”, (2015).

Du, Y. and L. B. Chilton, “Storywars: A dataset and instruction tuning baselines for collaborative story understanding and generation”, in “Annual Meeting of the Association for Computational Linguistics”, (2023), URL `https://api.semanticscholar.org/CorpusID:258686417`.

Efrat, A. and O. Levy, “The turking test: Can language models understand instructions?”, ArXiv **abs/2010.11982** (2020).

Feng, Z., D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang and M. Zhou, “CodeBERT: A pre-trained model for programming and natural languages”, in “Findings of the Association for Computational Linguistics: EMNLP 2020”, pp. 1536–1547 (Association for Computational Linguistics, Online, 2020), URL `https://aclanthology.org/2020.findings-emnlp.139`.

Fu, J., S.-K. Ng, Z. Jiang and P. Liu, “Gptscore: Evaluate as you desire”, arXiv e-prints pp. arXiv–2302 (2023).

Gao, P., J. Han, R. Zhang, Z. Lin, S. Geng, A. Zhou, W. Zhang, P. Lu, C. He, X. Yue, H. Li and Y. J. Qiao, “Llama-adapter v2: Parameter-efficient visual instruction model”, ArXiv **abs/2304.15010**, URL `https://api.semanticscholar.org/CorpusID:258418343` (2023).

Ghosal, D., N. Majumder, A. Mehrish and S. Poria, “Text-to-audio generation using instruction-tuned llm and latent diffusion model”, ArXiv **abs/2304.13731**, URL `https://api.semanticscholar.org/CorpusID:258352270` (2023).

Glaese, A., N. McAleese, M. Trkebacz, J. Aslanides, V. Firoiu, T. Ewalds, M. Rauh, L. Weidinger, M. Chadwick, P. Thacker, L. Campbell-Gillingham, J. Uesato, P.-S. Huang, R. Comanescu, F. Yang, A. See, S. Dathathri, R. Greig, C. Chen, D. Fritz, J. S. Elias, R. Green, S. Mokr'a, N. Fernando, B. Wu, R. Foley, S. Young, I. Gabriel, W. S. Isaac, J. F. J. Mellor, D. Hassabis, K. Kavukcuoglu, L. A. Hendricks and G. Irving, "Improving alignment of dialogue agents via targeted human judgements", ArXiv **abs/2209.14375**, URL `https://api.semanticscholar.org/CorpusID:252596089` (2022).

Gu, Y., P. Ke, X. Zhu and M. Huang, "Learning instructions with unlabeled data for zero-shot cross-task generalization", arXiv preprint arXiv:2210.09175 (2022).

Guo, X., B. A. Li and H. Yu, "Improving the sample efficiency of prompt tuning with domain adaptation", ArXiv **abs/2210.02952** (2022).

Gupta, H., A. A. Gulanikar, L. Kumar and L. B. M. Neti, "Empirical analysis on effectiveness of nlp methods for predicting code smell", in "Computational Science and Its Applications – ICCSA 2021", edited by O. Gervasi, B. Murgante, S. Misra, C. Garau, I. Blečić, D. Taniar, B. O. Apduhan, A. M. A. C. Rocha, E. Tarantino and C. M. Torre, pp. 43–53 (Springer International Publishing, Cham, 2021a).

Gupta, H., T. G. Kulkarni, L. Kumar and N. L. B. Murthy, "A novel approach towards analysis of attacker behavior in ddos attacks", in "Machine Learning for Networking", edited by S. Boumerdassi, É. Renault and P. Mühlethaler, pp. 392–402 (Springer International Publishing, Cham, 2020).

Gupta, H., T. G. Kulkarni, L. Kumar, L. B. M. Neti and A. Krishna, "An empirical study on predictability of software code smell using deep learning models", in "Advanced Information Networking and Applications", edited by L. Barolli, I. Woungang and T. Enokido, pp. 120–132 (Springer International Publishing, Cham, 2021b).

Gupta, H., L. Kumar and L. B. M. Neti, "An empirical framework for code smell prediction using extreme learning machine", in "2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON)", pp. 189–195 (2019).

Gupta, H., S. Misra, L. Kumar and N. L. B. Murthy, "An empirical study to investigate data sampling techniques for improving code-smell prediction using imbalanced data", in "Information and Communication Technology and Applications", edited by S. Misra and B. Muhammad-Bello, pp. 220–233 (Springer International Publishing, Cham, 2021c).

Gupta, H., S. A. Sawant, S. Mishra, M. Nakamura, A. Mitra, S. Mashetty and C. Baral, "Instruction tuned models are quick learners", ArXiv **abs/2306.05539**, URL `https://api.semanticscholar.org/CorpusID:259129868` (2023a).

Gupta, H., K. Scaria, U. Anantheswaran, S. Verma, M. Parmar, S. A. Sawant, S. Mishra and C. Baral, "Targen: Targeted data generation with large language models", arXiv preprint arXiv:2310.17876 (2023b).

Gupta, H., N. Varshney, S. Mishra, K. K. Pal, S. A. Sawant, K. Scaria, S. Goyal and C. Baral, “” john is 50 years old, can his son be 65?” evaluating nlp models’ understanding of feasibility”, arXiv preprint arXiv:2210.07471 (2022a).

Gupta, H., S. Verma, T. Kumar, S. Mishra, T. Agrawal, A. Badugu and H. S. Bhatt, “Context-ner: Contextual phrase generation at scale”, arXiv preprint arXiv:2109.08079 (2021d).

Gupta, P., C. Jiao, Y.-T. Yeh, S. Mehri, M. Eskénazi and J. P. Bigham, “Instructdial: Improving zero and few-shot generalization in dialogue through instruction tuning”, in “Conference on Empirical Methods in Natural Language Processing”, (2022b).

Hase, P. and M. Bansal, “When can models learn from explanations? a formal framework for understanding the roles of explanation data”, in “Proceedings of the First Workshop on Learning with Natural Language Supervision”, pp. 29–39 (Association for Computational Linguistics, Dublin, Ireland, 2022), URL `https://aclanthology.org/2022.lnls-1.4`.

Honovich, O., T. Scialom, O. Levy and T. Schick, “Unnatural instructions: Tuning language models with (almost) no human labor”, ArXiv **abs/2212.09689**, URL `https://api.semanticscholar.org/CorpusID:254853659` (2022a).

Honovich, O., U. Shaham, S. R. Bowman and O. Levy, “Instruction induction: From few examples to natural language task descriptions”, in “Annual Meeting of the Association for Computational Linguistics”, (2022b), URL `https://api.semanticscholar.org/CorpusID:248986755`.

Husain, H., H. Wu, T. Gazit, M. Allamanis and M. Brockschmidt, “Codesearchnet challenge: Evaluating the state of semantic code search”, ArXiv **abs/1909.09436** (2019).

Ivison, H., A. Bhagia, Y. Wang, H. Hajishirzi and M. E. Peters, “Hint: Hypernetwork instruction tuning for efficient zero-shot generalisation”, ArXiv **abs/2212.10315** (2022a).

Ivison, H., N. A. Smith, H. Hajishirzi and P. Dasigi, “Data-efficient finetuning using cross-task nearest neighbors”, ArXiv **abs/2212.00196**, URL `https://api.semanticscholar.org/CorpusID:254125437` (2022b).

Iyer, S., X. Lin, R. Pasunuru, T. Mihaylov, D. Simig, P. Yu, K. Shuster, T. Wang, Q. Liu, P. S. Koura, X. Li, B. O’Horo, G. Pereyra, J. Wang, C. Dewan, A. Celikyilmaz, L. Zettlemoyer and V. Stoyanov, “Opt-iml: Scaling language model instruction meta learning through the lens of generalization”, ArXiv **abs/2212.12017**, URL `https://api.semanticscholar.org/CorpusID:255096269` (2022).

Jang, J., S. Kim, S. Ye, D. Kim, L. Logeswaran, M. Lee, K. Lee and M. Seo, “Exploring the benefits of training expert language models over instruction tuning”, ArXiv **abs/2302.03202** (2023).

Ji, Y., Y. Gong, Y. Deng, Y. Peng, Q. Niu, B. Ma and X. Li, "Towards better instruction following language models for chinese: Investigating the impact of training data and evaluation", ArXiv **abs/2304.07854**, URL `https://api.semanticscholar.org/CorpusID:258180415` (2023).

Jiang, D., X. Ren and B. Y. Lin, "Llm-blender: Ensembling large language models with pairwise ranking and generative fusion", in "Annual Meeting of the Association for Computational Linguistics", (2023), URL `https://api.semanticscholar.org/CorpusID:259075564`.

Jiao, F., B. Ding, T. Luo and Z. Mo, "Panda llm: Training data and evaluation for open-sourced chinese instruction-following large language models", ArXiv **abs/2305.03025**, URL `https://api.semanticscholar.org/CorpusID:258479799` (2023).

Kang, W., S. P. Hernández, J. Wang and A. Malvaso, "Instruction-based learning: A review", Neuropsychologia p. 108142 (2022).

Kehinde, A. J., A. E. Adeniyi, R. O. Ogundokun, H. Gupta and S. Misra, "Prediction of students' performance with artificial neural network using demographic traits", in "Recent Innovations in Computing", edited by P. K. Singh, Y. Singh, J. K. Chhabra, Z. Illés and C. Verma, pp. 613–624 (Springer Singapore, Singapore, 2022).

Khashabi, D., S. Min, T. Khot, A. Sabharwal, O. Tafjord, P. Clark and H. Hajishirzi, "UnifiedQA: Crossing format boundaries with a single QA system", in "Findings of the Association for Computational Linguistics: EMNLP 2020", (2020), URL `https://aclanthology.org/2020.findings-emnlp.171`.

Khot, T., D. Khashabi, K. Richardson, P. Clark and A. Sabharwal, "Text modular networks: Learning to decompose tasks in the language of existing models", arXiv preprint arXiv:2009.00751 (2020).

Kim, S., S. J. Joo, D. Kim, J. Jang, S. Ye, J. Shin and M. Seo, "The cot collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning", ArXiv **abs/2305.14045**, URL `https://api.semanticscholar.org/CorpusID:258841149` (2023).

Köksal, A., T. Schick, A. Korhonen and H. Schütze, "Longform: Optimizing instruction tuning for long text generation with corpus extraction", ArXiv **abs/2304.08460**, URL `https://api.semanticscholar.org/CorpusID:258179256` (2023).

Kuznia, K., S. Mishra, M. Parmar and C. Baral, "Less is more: Summary of long instructions is better for program synthesis", in "Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing", pp. 4532–4552 (Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022), URL `https://aclanthology.org/2022.emnlp-main.301`.

Lagani, G., F. Falchi, C. Gennaro and G. Amato, "Hebbian semi-supervised learning in a sample efficiency setting", Neural networks : the official journal of the International Neural Network Society **143**, 719–731 (2021).

Le Scao, T. and A. M. Rush, "How many data points is a prompt worth?", in "Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies", pp. 2627–2636 (2021).

Lee, Y.-S., R. F. Astudillo, R. Florian, T. Naseem and S. Roukos, "Amr parsing with instruction fine-tuned pre-trained language models", ArXiv **abs/2304.12272**, URL `https://api.semanticscholar.org/CorpusID:258298140` (2023).

Li, B., Y. Zhang, L. Chen, J. Wang, J. Yang and Z. Liu, "Otter: A multi-modal model with in-context instruction tuning", ArXiv **abs/2305.03726**, URL `https://api.semanticscholar.org/CorpusID:258547300` (2023a).

Li, J., Y. Wang, M. R. Lyu and I. King, "Code completion with neural attention and pointer networks", arXiv preprint arXiv:1711.09573 (2017).

Li, L., Y. Yin, S. Li, L. Chen, P. Wang, S. Ren, M. Li, Y. Yang, J. Xu, X. Sun, L. Kong and Q. Liu, "M3it: A large-scale dataset towards multi-modal multilingual instruction tuning", ArXiv **abs/2306.04387**, URL `https://api.semanticscholar.org/CorpusID:259095896` (2023b).

Liang, P., R. Bommasani, T. Lee, D. Tsipras, D. Soylu, M. Yasunaga, Y. Zhang, D. Narayanan, Y. Wu, A. Kumar *et al.*, "Holistic evaluation of language models", arXiv preprint arXiv:2211.09110 (2022).

Lin, B. Y., K. Tan, C. Miller, B. Tian and X. Ren, "Unsupervised cross-task generalization via retrieval augmentation", arXiv preprint arXiv:2204.07937 (2022).

Lin, C.-Y., "ROUGE: A package for automatic evaluation of summaries", in "Text Summarization Branches Out", pp. 74–81 (Association for Computational Linguistics, Barcelona, Spain, 2004), URL `https://aclanthology.org/W04-1013`.

Liu, H., D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal and C. A. Raffel, "Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning", Advances in Neural Information Processing Systems **35**, 1950–1965 (2022).

Liu, Q., F. Zhou, Z. Jiang, L. Dou and M. Lin, "From zero to hero: Examining the power of symbolic tasks in instruction tuning", ArXiv **abs/2304.07995**, URL `https://api.semanticscholar.org/CorpusID:258179750` (2023).

Longpre, S., L. Hou, T. Vu, A. Webson, H. W. Chung, Y. Tay, D. Zhou, Q. V. Le, B. Zoph, J. Wei and A. Roberts, "The flan collection: Designing data and methods for effective instruction tuning", ArXiv **abs/2301.13688** (2023).

Luo, M., S. Saxena, S. Mishra, M. Parmar and C. Baral, "Biotabqa: Instruction learning for biomedical table question answering", arXiv preprint arXiv:2207.02419 (2022).

Ma, X., S. Mishra, A. Beirami, A. Beutel and J. Chen, "Let's do a thought experiment: Using counterfactuals to improve moral reasoning", arXiv preprint arXiv:2306.14308 (2023).

Malkiel, I. and L. Wolf, “Maximal multiverse learning for promoting cross-task generalization of fine-tuned language models”, in “Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume”, pp. 187–199 (Association for Computational Linguistics, Online, 2021), URL https://aclanthology.org/2021.eacl-main.14.

Menon, R., S. Ghosh and S. Srivastava, “Clues: A benchmark for learning classifiers using natural language explanations”, in “Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)”, pp. 6523–6546 (2022).

Min, S., M. Lewis, L. Zettlemoyer and H. Hajishirzi, “MetaICL: Learning to learn in context”, in “Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies”, pp. 2791–2809 (Association for Computational Linguistics, Seattle, United States, 2022), URL https://aclanthology.org/2022.naacl-main.201.

Mishra, S., D. Khashabi, C. Baral, Y. Choi and H. Hajishirzi, “Reframing instructional prompts to GPTk’s language”, in “Findings of the Association for Computational Linguistics: ACL 2022”, pp. 589–612 (Association for Computational Linguistics, Dublin, Ireland, 2022a), URL https://aclanthology.org/2022.findings-acl.50.

Mishra, S., D. Khashabi, C. Baral and H. Hajishirzi, “Cross-task generalization via natural language crowdsourcing instructions”, in “Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)”, pp. 3470–3487 (2022b).

Mishra, S. and E. Nouri, “Help me think: A simple prompting strategy for non-experts to create customized content with models”, arXiv preprint arXiv:2208.08232 (2022).

Muennighoff, N., T. Wang, L. Sutawika, A. Roberts, S. Biderman, T. L. Scao, M. S. Bari, S. Shen, Z.-X. Yong, H. Schoelkopf *et al.*, “Crosslingual generalization through multitask finetuning”, arXiv preprint arXiv:2211.01786 (2022).

Muennighoff, N., T. Wang, L. Sutawika, A. Roberts, S. R. Biderman, T. L. Scao, M. S. Bari, S. Shen, Z.-X. Yong, H. Schoelkopf, X. Tang, D. R. Radev, A. F. Aji, K. Almubarak, S. Albanie, Z. Alyafeai, A. Webson, E. Raff and C. Raffel, “Crosslingual generalization through multitask finetuning”, in “Annual Meeting of the Association for Computational Linguistics”, (2023), URL https://api.semanticscholar.org/CorpusID:253264914.

Nakano, R., J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saunders *et al.*, “Webgpt: Browser-assisted question-answering with human feedback”, arXiv preprint arXiv:2112.09332 (2021).

Ni, X. and P. Li, “Unified text structuralization with instruction-tuned language models”, ArXiv **abs/2303.14956**, URL https://api.semanticscholar.org/CorpusID:257766560 (2023).

Nye, M., A. J. Andreassen, G. Gur-Ari, H. Michalewski, J. Austin, D. Bieber, D. Dohan, A. Lewkowycz, M. Bosma, D. Luan *et al.*, "Show your work: Scratchpads for intermediate computation with language models", arXiv preprint arXiv:2112.00114 (2021).

Ogundokun, R. O., S. Misra, P. O. Sadiku, H. Gupta, R. Damasevicius and R. Maskeliunas, "Computational intelligence approaches for heart disease detection", in "Recent Innovations in Computing", edited by P. K. Singh, Y. Singh, J. K. Chhabra, Z. Illés and C. Verma, pp. 385–395 (Springer Singapore, Singapore, 2022).

Ouyang, L., J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback", Advances in Neural Information Processing Systems **35**, 27730–27744 (2022).

Parmar, M., S. Mishra, M. Purohit, M. Luo, M. Mohammad and C. Baral, "In-BoXBART: Get instructions into biomedical multi-task learning", in "Findings of the Association for Computational Linguistics: NAACL 2022", pp. 112–128 (Association for Computational Linguistics, Seattle, United States, 2022), URL `https://aclanthology.org/2022.findings-naacl.10`.

Patel, P., S. Mishra, M. Parmar and C. Baral, "Is a question decomposition unit all we need?", in "Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing", pp. 4553–4569 (Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022), URL `https://aclanthology.org/2022.emnlp-main.302`.

Peng, B., C. Li, P. He, M. Galley and J. Gao, "Instruction tuning with gpt-4", arXiv preprint arXiv:2304.03277 (2023).

Prabhumoye, S., M. Patwary, M. Shoeybi and B. Catanzaro, "Adding instructions during pretraining: Effective way of controlling toxicity in language models", in "Conference of the European Chapter of the Association for Computational Linguistics", (2023), URL `https://api.semanticscholar.org/CorpusID:256868688`.

Puri, R. S., S. Mishra, M. Parmar and C. Baral, "How many data samples is an additional instruction worth?", in "Findings of the Association for Computational Linguistics: EACL 2023", pp. 1042–1057 (Association for Computational Linguistics, Dubrovnik, Croatia, 2023), URL `https://aclanthology.org/2023.findings-eacl.77`.

Raffel, C., N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer", (2020).

Raheja, V., D. Kumar, R. Koo and D. Kang, "Coedit: Text editing by task-specific instruction tuning", ArXiv **abs/2305.09857**, URL `https://api.semanticscholar.org/CorpusID:258741409` (2023).

Reif, E., D. Ippolito, A. Yuan, A. Coenen, C. Callison-Burch and J. Wei, "A recipe for arbitrary text style transfer with large language models", arXiv preprint arXiv:2109.03910 (2021).

Sanh, V., A. Webson, C. Raffel, S. H. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, T. L. Scao, A. Raja *et al.*, "Multitask prompted training enables zero-shot task generalization", ICLR 2022 URL `https://arxiv.org/abs/2110.08207` (2021).

Scaria, K., H. Gupta, S. A. Sawant, S. Mishra and C. Baral, "Instructabsa: Instruction learning for aspect based sentiment analysis", arXiv preprint arXiv:2302.08624 (2023).

Schick, T. and H. Schütze, "True few-shot learning with prompts—a real-world perspective", Transactions of the Association for Computational Linguistics **10**, 716–731 (2022).

Shao, L., T. Migimatsu, Q. Zhang, K. Yang and J. Bohg, "Concept2robot: Learning manipulation concepts from instructions and human demonstrations", The International Journal of Robotics Research **40**, 12-14, 1419–1434 (2021).

Slack, D. and S. Singh, "Tablet: Learning from instructions for tabular data", ArXiv **abs/2304.13188**, URL `https://api.semanticscholar.org/CorpusID:258331602` (2023).

Štefánik, M., M. Kadlčík, P. Gramacki and P. Sojka, "Resources and few-shot learners for in-context learning in slavic languages", in "Proceedings of the 9th Workshop on Slavic Natural Language Processing 2023 (SlavicNLP 2023)", pp. 94–105 (2023).

Su, H., J. Kasai, C. H. Wu, W. Shi, T. Wang, J. Xin, R. Zhang, M. Ostendorf, L. Zettlemoyer, N. A. Smith *et al.*, "Selective annotation makes language models better few-shot learners", arXiv preprint arXiv:2209.01975 (2022a).

Su, H., W. Shi, J. Kasai, Y. Wang, Y. Hu, M. Ostendorf, W. tau Yih, N. A. Smith, L. Zettlemoyer and T. Yu, "One embedder, any task: Instruction-finetuned text embeddings", ArXiv **abs/2212.09741**, URL `https://api.semanticscholar.org/CorpusID:254853816` (2022b).

Sun, X., Y. Ji, B. Ma and X. Li, "A comparative study between full-parameter and lora-based fine-tuning on chinese instruction data for instruction following large language model", ArXiv **abs/2304.08109**, URL `https://api.semanticscholar.org/CorpusID:258179474` (2023).

Suzgun, M., N. Scales, N. Schärli, S. Gehrmann, Y. Tay, H. W. Chung, A. Chowdhery, Q. V. Le, E. H. Chi, D. Zhou *et al.*, "Challenging big-bench tasks and whether chain-of-thought can solve them", arXiv preprint arXiv:2210.09261 (2022).

Tian, J., H. Chen, G. Xu, M. Yan, X. Gao, J. Zhang, C. Li, J. Liu, W. Xu, H. Xu, Q. Qian, W. Wang, Q. Ye, J. Zhang, J. Zhang, F. Huang and J. Zhou, "Chatplug: Open-domain generative dialogue system with internet-augmented instruction tuning for digital human", ArXiv **abs/2304.07849**, URL `https://api.semanticscholar.org/CorpusID:258179346` (2023).

Varshney, N., S. Mishra and C. Baral, "It's better to say "i can't answer" than answering incorrectly: Towards safety critical nlp systems", ArXiv **abs/2008.09371**, URL `https://api.semanticscholar.org/CorpusID:221246165` (2020).

Vijayakumar, A. J., A. Mohta, O. Polozov, D. Batra, P. Jain and S. Gulwani, "Neural-guided deductive search for real-time program synthesis from examples", ArXiv **abs/1804.01186** (2018).

Wan, A., E. Wallace, S. Shen and D. Klein, "Poisoning language models during instruction tuning", ArXiv **abs/2305.00944**, URL `https://api.semanticscholar.org/CorpusID:258426823` (2023).

Wang, A., Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy and S. Bowman, "Superglue: A stickier benchmark for general-purpose language understanding systems", Advances in neural information processing systems **32** (2019).

Wang, L., R. Li, Y. Yan, Y. Yan, S. Wang, W. Wu and W. Xu, "Instructionner: A multi-task instruction-based generative framework for few-shot ner", arXiv preprint arXiv:2203.03903 (2022a).

Wang, X., W. Zhou, C. Zu, H. Xia, T. Chen, Y. Zhang, R. Zheng, J. Ye, Q. Zhang, T. Gui, J. Kang, J. Yang, S. Li and C. Du, "Instructuie: Multi-task instruction tuning for unified information extraction", ArXiv **abs/2304.08085** (2023).

Wang, Y., Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi and H. Hajishirzi, "Self-instruct: Aligning language models with self-generated instructions", in "Annual Meeting of the Association for Computational Linguistics", (2022b), URL `https://api.semanticscholar.org/CorpusID:254877310`.

Wang, Y., S. Mishra, P. Alipoormolabashi, Y. Kordi, A. Mirzaei, A. Naik, A. Ashok, A. S. Dhanasekaran, A. Arunkumar, D. Stap, E. Pathak, G. Karamanolakis, H. Lai, I. Purohit, I. Mondal, J. Anderson, K. Kuznia, K. Doshi, K. K. Pal, M. Patel, M. Moradshahi, M. Parmar, M. Purohit, N. Varshney, P. R. Kaza, P. Verma, R. S. Puri, R. Karia, S. Doshi, S. K. Sampat, S. Mishra, S. Reddy A, S. Patro, T. Dixit and X. Shen, "Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks", in "Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing", pp. 5085–5109 (Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022c), URL `https://aclanthology.org/2022.emnlp-main.340`.

Wang, Z., X. Pan, D. Yu, D. Yu, J. Chen and H. Ji, "Zemi: Learning zero-shot semi-parametric language models from multiple tasks", in "Annual Meeting of the Association for Computational Linguistics", (2022d), URL `https://api.semanticscholar.org/CorpusID:252683285`.

Wang, Z., R. Xia and J. Yu, "Unifiedabsa: A unified absa framework based on multi-task instruction tuning", ArXiv **abs/2211.10986**, URL https://api.semanticscholar.org/CorpusID:253735130 (2022e).

Wei, J., M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai and Q. V. Le, "Finetuned language models are zero-shot learners", in "International Conference on Learning Representations", (2021).

Wei, J., X. Wang, D. Schuurmans, M. Bosma, E. H. hsin Chi, F. Xia, Q. Le and D. Zhou, "Chain of thought prompting elicits reasoning in large language models", ArXiv **abs/2201.11903** (2022).

Weller, O., N. Lourie, M. Gardner and M. E. Peters, "Learning from task descriptions", in "Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)", pp. 1361–1375 (Association for Computational Linguistics, Online, 2020), URL https://aclanthology.org/2020.emnlp-main.105.

Wu, M., A. Waheed, C. Zhang, M. Abdul-Mageed and A. F. Aji, "Lamini-lm: A diverse herd of distilled models from large-scale instructions", ArXiv **abs/2304.14402**, URL https://api.semanticscholar.org/CorpusID:258352678 (2023a).

Wu, T. S., M. Terry and C. J. Cai, "Ai chains: Transparent and controllable human-ai interaction by chaining large language model prompts", Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (2021).

Wu, Y., Y. Zhao, Z. Li, B. Qin and K. Xiong, "Improving cross-task generalization with step-by-step instructions", ArXiv **abs/2305.04429**, URL https://api.semanticscholar.org/CorpusID:258556997 (2023b).

Xie, T., C. H. Wu, P. Shi, R. Zhong, T. Scholak, M. Yasunaga, C.-S. Wu, M. Zhong, P. Yin, S. I. Wang, V. Zhong, B. Wang, C. Li, C. Boyle, A. Ni, Z. Yao, D. R. Radev, C. Xiong, L. Kong, R. Zhang, N. A. Smith, L. Zettlemoyer and T. Yu, "Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models", ArXiv **abs/2201.05966**, URL https://api.semanticscholar.org/CorpusID:246016124 (2022).

Xu, C., Q. Sun, K. Zheng, X. Geng, P. Zhao, J. Feng, C. Tao and D. Jiang, "Wizardlm: Empowering large language models to follow complex instructions", ArXiv **abs/2304.12244**, URL https://api.semanticscholar.org/CorpusID:258298159 (2023).

Xu, H., Y. Chen, Y. Du, N. Shao, W. Yanggang, H. Li and Z. Yang, "Zeroprompt: Scaling prompt-based pretraining to 1,000 tasks improves zero-shot generalization", in "Findings of the Association for Computational Linguistics: EMNLP 2022", pp. 4235–4252 (2022a).

Xu, M., F. Qian, Q. Mei, K. Huang and X. Liu, "Deeptype: On-device deep learning for input personalization service with minimal privacy concern", Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. **2**, 4, URL `https://doi.org/10.1145/3287075` (2018).

Xu, Z., Y. Shen and L. Huang, "Multiinstruct: Improving multi-modal zero-shot learning via instruction tuning", ArXiv **abs/2212.10773** (2022b).

Yang, C., J. Pan, X. Gao, T. Jiang, D. Liu and G. Chen, "Cross-task knowledge distillation in multi-task recommendation", in "Proceedings of the AAAI Conference on Artificial Intelligence", vol. 36, pp. 4318–4326 (2022a).

Yang, S., Y. Dong, R. A. Ward, I. S. Dhillon, S. Sanghavi and Q. Lei, "Sample efficiency of data augmentation consistency regularization", ArXiv **abs/2202.12230** (2022b).

Yang, Z., K. Ren, X. Luo, M. Liu, W. Liu, J. Bian, W. Zhang and D. Li, "Towards applicable reinforcement learning: Improving the generalization and sample efficiency with policy ensemble", in "International Joint Conference on Artificial Intelligence", (2022c).

Yarats, D., A. Zhang, I. Kostrikov, B. Amos, J. Pineau and R. Fergus, "Improving sample efficiency in model-free reinforcement learning from images", in "AAAI Conference on Artificial Intelligence", (2019).

Ye, Q., B. Y. Lin and X. Ren, "Crossfit: A few-shot learning challenge for cross-task generalization in nlp", in "Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing", pp. 7163–7189 (2021).

Ye, Q. and X. Ren, "Learning to generate task-specific adapters from task description", in "Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)", pp. 646–653 (Association for Computational Linguistics, Online, 2021), URL `https://aclanthology.org/2021.acl-short.82`.

Ye, S., H. Hwang, S. Yang, H. Yun, Y. Kim and M. Seo, "In-context instruction learning", ArXiv **abs/2302.14691**, URL `https://api.semanticscholar.org/CorpusID:257233109` (2023).

Ye, S., D. Kim, J. Jang, J. Shin and M. Seo, "Guess the instruction! making language models stronger zero-shot learners", arXiv preprint arXiv:2210.02969 (2022).

Yin, D., X. Liu, F. Yin, M. Zhong, H. Bansal, J. Han and K.-W. Chang, "Dynosaur: A dynamic growth paradigm for instruction-tuning data curation", ArXiv **abs/2305.14327**, URL `https://api.semanticscholar.org/CorpusID:258841263` (2023).

Yin, P., G. Neubig, M. Allamanis, M. Brockschmidt and A. L. Gaunt, "Learning to represent edits", ArXiv **abs/1810.13337** (2018).

Yuan, W. and P. Liu, "restructured pre-training", ArXiv **abs/2206.11147**, URL `https://api.semanticscholar.org/CorpusID:249926432` (2022).

Zhang, G., Y. Shi, R. Liu, R. Yuan, Y. Li, S. Dong, Y. Shu, Z. Li, Z. Wang, C. Lin, W.-F. Huang and J. Fu, "Chinese open instruction generalist: A preliminary release", ArXiv **abs/2304.07987**, URL `https://api.semanticscholar.org/CorpusID:258179044` (2023a).

Zhang, J., C. Ni, Z. Yu, C. Szepesvari and M. Wang, "On the convergence and sample efficiency of variance-reduced policy gradient method", in "Neural Information Processing Systems", (2021).

Zhang, J., S. Vahidian, M. Kuo, C. Li, R. Zhang, G. Wang and Y. Chen, "Towards building the federated gpt: Federated instruction tuning", (2023b).

Zhang, J., R. Xie, Y. Hou, W. X. Zhao, L. Lin and J. rong Wen, "Recommendation as instruction following: A large language model empowered recommendation approach", ArXiv **abs/2305.07001**, URL `https://api.semanticscholar.org/CorpusID:258615776` (2023c).

Zhang, K., B. J. Gutierrez and Y. Su, "Aligning instruction tasks unlocks large language models as zero-shot relation extractors", ArXiv **abs/2305.11159**, URL `https://api.semanticscholar.org/CorpusID:258762464` (2023d).

Zhang, R., Y.-S. Wang and Y. Yang, "Generation-driven contrastive self-training for zero-shot text classification with instruction-tuned gpt", ArXiv **abs/2304.11872**, URL `https://api.semanticscholar.org/CorpusID:258297998` (2023e).

Zhang, S., L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu and G. Wang, "Instruction tuning for large language models: A survey", ArXiv **abs/2308.10792**, URL `https://api.semanticscholar.org/CorpusID:261049152` (2023f).

Zhang, T., F. Liu, J. Wong, P. Abbeel and J. E. Gonzalez, "The wisdom of hindsight makes language models better instruction followers", (2023g).

Zhang, Y. and J. Chai, "Hierarchical task learning from language instructions with unified transformers and self-monitoring", in "Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021", pp. 4202–4213 (2021).

Zheng, H. S., S. Mishra, X. Chen, H.-T. Cheng, E. H. Chi, Q. V. Le and D. Zhou, "Take a step back: Evoking reasoning via abstraction in large language models", arXiv preprint arXiv:2310.06117 (2023).

Zhong, R., K. Lee, Z. Zhang and D. Klein, "Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections", in "Findings of the Association for Computational Linguistics: EMNLP 2021", pp. 2856–2878 (Association for Computational Linguistics, Punta Cana, Dominican Republic, 2021), URL `https://aclanthology.org/2021.findings-emnlp.244`.

Zhou, C., J. He, X. Ma, T. Berg-Kirkpatrick and G. Neubig, "Prompt consistency for zero-shot task generalization", in "Conference on Empirical Methods in Natural Language Processing", (2022a), URL `https://api.semanticscholar.org/CorpusID:248496641`.

Zhou, D., N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, O. Bousquet, Q. Le and E. Chi, "Least-to-most prompting enables complex reasoning in large language models", arXiv preprint arXiv:2205.10625 (2022b).

Zhou, W., Y. Jiang, E. G. Wilcox, R. Cotterell and M. Sachan, "Controlled text generation with natural language instructions", ArXiv **abs/2304.14293**, URL `https://api.semanticscholar.org/CorpusID:258352203` (2023).

APPENDIX A

EXTENDED RELATED WORK

LLMs and deep learning methods have been extensively applied across a myriad of downstream tasks for a considerable duration Yin *et al.* (2018); Li *et al.* (2017); Das (2015); Gupta *et al.* (2020, 2021b,c,a); Husain *et al.* (2019); Feng *et al.* (2020); Vijayakumar *et al.* (2018); Arunkumar *et al.* (2023a). Recent studies have harnessed natural language processing (NLP) methods and simple sampling techniques to yield diverse downstream outcomes Xu *et al.* (2018); Alon *et al.* (2018); Allamanis *et al.* (2017); Balog *et al.* (2016); Ogundokun *et al.* (2022); Kehinde *et al.* (2022); Gupta *et al.* (2019). The investigation into the ability of existing LMs to comprehend instructions by Efrat and Levy (2020) has spurred subsequent research endeavors. Notably, Hase and Bansal (2022), Ye and Ren (2021), and Zhong *et al.* (2021) have proposed diverse methods to showcase that language models are adept at following instructions. Weller *et al.* (2020) have devised a framework that concentrates on developing NLP systems capable of solving novel tasks after reading their descriptions. PromptSource and FLAN Wei *et al.* (2021); Sanh *et al.* (2021) were specifically designed to leverage instructions and accomplish zero-shot generalization on previously unseen tasks. Additionally, Parmar *et al.* (2022) demonstrate the effectiveness of instructions in multi-task settings, particularly within the biomedical domain. Mishra *et al.* (2022a) delve into the impact of reframing task instructions on model responses, while Min *et al.* (2022) introduce a framework to enhance understanding in the context of learning. Furthermore, Ouyang *et al.* (2022) propose the InstructGPT model, fine-tuned with human feedback to follow instructions. Gupta *et al.* (2022a) present evidence that augmenting knowledge with instructions aids LMs in better contextual understanding. Wang *et al.* (2022a) develop an instruction-based multi-task framework for few-shot Named Entity Recognition (NER) tasks. Several approaches have emerged to enhance model performance using instructions, including those proposed by Wu *et al.* (2021); Liu *et al.* (2022); Luo *et al.* (2022); Kuznia *et al.* (2022); Patel *et al.* (2022); Mishra and Nouri (2022); Puri *et al.* (2023); Gupta *et al.* (2021d); Anantheswaran *et al.* (2023); Scaria *et al.* (2023); Varshney *et al.* (2020)

Instruction Tuning Applications

1. **InstructDial: Improving Zero and Few-shot Generalization in Dialogue through Instruction Tuning Gupta *et al.* (2022b):** The paper introduces InstructDial, which focuses on enhancing the zero and few-shot generalization capabilities of dialogue systems. InstructDial's methodology revolves around the fine-tuning of pre-trained dialogue models using a limited set of dialogue examples, augmented with natural language instructions. This approach relies on instruction-based learning to improve the performance of dialogue systems, especially in scenarios where traditional supervised methods fall short. The authors have demonstrated that InstructDial surpasses other existing zero and few-shot generalization methods The method's capacity to equip dialogue systems to undertake entirely new tasks without the need for fine-tuning is a substantial advantage over traditional supervised approaches. By achieving this without requiring additional labeled data, InstructDial paves the way for dialogue systems that can adapt and excel in a broader spectrum of tasks, including

those that are novel.

2. **From Zero to Hero: Examining the Power of Symbolic Tasks in Instruction Tuning Liu *et al.* (2023):** The paper delves into an important aspect of training large language models (LLMs) and evaluates the effectiveness of specialized, expert LLMs compared to multitask LLMs. When it comes to training LLMs for specific tasks, two primary approaches emerge: multitask learning, where a single model is trained on a broad range of tasks, and expert learning, where separate models are trained for each specific task. The studies show that expert LLMs consistently outperform multitask LLMs on unseen tasks. This underlines the value of developing task-specific expertise within models. The reduced data and training time requirements of expert LLMs are particularly valuable in scenarios where resources are limited or where model training needs to be expedited.

3. **Unified Text Structuralization with Instruction-tuned Language Models Ni and Li (2023)** The research paper addresses Text structuralization which encompasses essential tasks like information extraction (IE) and structure formalization. The paper starts by acknowledging that existing approaches to text structuralization often rely on manually annotated datasets and specialized techniques tailored to different IE sub-tasks. These methods can be less efficient, lack generalizability, and require significant effort to adapt to various domains and tasks. This approach uses instruction-tuned language models to extract diverse structures from text. UTS-ILMs employ instructions that specify the desired IE task and structure type, which are added to the text before it is processed by an LLM. The LLM, guided by these instructions, can then effectively extract the relevant information and structures from the text.

   The study evaluates UTS-ILMs using two LLM models, PaLM and OPT, across a comprehensive dataset spanning various text domains and knowledge types. The findings from this empirical investigation yield several key results:

   - Unified Text Structuralization: UTS-ILMs present a unified approach to text structuralization. By utilizing instruction-tuned language models, these models enable LLMs to manage a variety of IE tasks and structure types without the need for specialized and complex methods.
   - Performance Superiority: UTS-ILMs outperform state-of-the-art methods across different IE sub-tasks, including entity recognition, relation extraction, and event detection.
   - Efficiency and Generalizability: UTS-ILMs offer increased efficiency and generalizability. These models reduce the reliance on manually annotated datasets, making them more scalable and adaptable to a variety of domains and knowledge types. The instruction-tuning method allows for quick generalization to new IE tasks and structure types without necessitating extensive additional training or data.

4. **InstructUIE: Multi-task Instruction Tuning for Unified Information Extraction Wang *et al.* (2023)** The paper presents InstructUIE, a framework

for unified information extraction that leverages the self-supervised learning capabilities of large language models (LLMs). InstructUIE is designed to generate instruction-following data for fine-tuning LLMs, thereby addressing the need for specialized techniques and manual annotation. InstructUIE consists of two primary steps: In this phase, an LLM is employed to create instruction-following data. This data generation involves providing the LLM with examples of IE tasks and the desired output structures. The model then learns to generate new instructions based on these examples.Following the generation of instruction-following data, the model is fine-tuned using this information. The fine-tuning process improves the model's capacity to follow instructions and execute a range of IE sub-tasks, including but not limited to entity recognition, relation extraction, and event detection. One of the significant implications is the reduction in the dependence on manually annotated datasets and specialized techniques for different IE sub-tasks. This enhances efficiency and scalability across diverse domains and knowledge types.

5. **AMR Parsing with Instruction Fine-tuned Pre-trained Language Models Lee *et al.* (2023)** The paper addresses the task of improving Abstract Meaning Representation (AMR) parsing using pre-trained language models, particularly focusing on the application of instruction fine-tuning. AMR parsing involves the automatic generation of structured, graph-based representations of sentences, capturing their semantic meaning in a consistent manner.The key idea of this work is that Large language models (LLMs) can be significantly enhanced in their AMR parsing capabilities by leveraging instruction fine-tuning. Instruction Tuning enables LLMs to learn how to follow instructions effectively and perform specific tasks, in this case, AMR parsing. The authors employ FLAN-T5, as the base for their experiments and fine-tune it on a dataset of AMR parsing instructions. The study evaluates the performance of the instruction fine-tuned FLAN-T5 models across three key AMR parsing benchmarks: AMR2.0, AMR3.0, and BioAMR. The study establishes that instruction tuning can significantly enhance the performance of LLMs in the domain of AMR parsing. This technique allows the models to better understand and execute instructions related to this complex task. The fine-tuned FLAN-T5 models, specifically Flan-T5-Large and Flan-T5-XL, achieve new state-of-the-art results across all three AMR parsing benchmarks, namely AMR2.0, AMR3.0, and BioAMR. This highlights the potential of instruction fine-tuning to advance the field.

6. **LaMini-LM: A Diverse Herd of Distilled Models from Large-Scale Instructions Wu *et al.* (2023a)**

   LLMs have their sheer size and substantial computational demands have limited their applicability, especially in resource-constrained environments and devices. In response to this issue, the authors have introduced LaMini-LM, a collection of small-sized and efficient language models.

   LaMini-LM is distilled from ChatGPT. The methodology behind LaMini-LM is comprised of two pivotal steps: In this phase, a substantial and diverse set of dialogue tasks are gathered. These tasks encompass various facets of NLP, including knowledge, personality, multi-turn memory, and empathy. The data for these dialogue tasks is sourced from existing datasets, alongside the generation of new instructions using ChatGPT. The collected instruction data serves as the foundation for the training of a series of small-sized encoder-decoder and decoder-only transformer models. The evaluation of LaMini-LM was carried out using a variety of metrics, both automatic and human-centric, comparing its performance against state-of-the-art Chinese dialogue systems. The results of these evaluations indicate that LaMini-LM outperforms the baseline systems. It exhibits a comparable performance to ChatGPT in open-domain dialogue tasks, showcasing its prowess in text understanding and generation while upholding its smaller model size.

7. **LINGO: Visually Debiasing Natural Language Instructions to Support Task Diversity Arunkumar *et al.* (2023b)**

   LLMs can inherit biases present in the data they are trained on, potentially limiting the diversity of tasks they can perform based on the instructions they receive.

   To address this challenge, the authors introduced LINGO, a visual analytics interface designed to facilitate a task-driven workflow. LINGO serves three primary functions:

   LINGO allows users to visually identify potential biases in natural language task instructions. This is achieved by presenting the task instructions in a visual representation that makes it easier for users to spot linguistic features that could introduce bias.

   The tool allows users to alter or create task instructions that are less biased. This feature enables the refinement of the language used in instructions to ensure it is more neutral and unbiased.

   LINGO supports real-time evaluation of pre-trained models' performance on debiased task instructions, ensuring that the modifications result in improved model behavior.

   In an evaluation involving novice and expert instruction creators and a dataset comprising over 1,600 linguistic tasks across 55 different languages, LINGO showcased its effectiveness. It encouraged the generation of more challenging and unbiased tasks for pre-trained models, characterized by higher linguistic diversity and lower instruction bias.

8. **Text-to-Audio Generation using Instruction-Tuned LLM and Latent Diffusion Model Ghosal *et al.* (2023)**

   The paper introduces TTA that involves the synthesis of realistic audio waveforms from textual descriptions, a task that requires balancing linguistic fidelity and audio quality. Traditional approaches to TTA have often struggled to produce natural-sounding audio that accurately reflects the nuances of the input text. To address these limitations, TTA authors introdced TANGO. This approach has two critical components: an instruction-tuned large language model (LLM) and a latent diffusion model (LDM). The instruction-tuned LLM is designed to capture the semantic meaning of the input text, leveraging its training on a large corpus of text and instructions. It generates an intermediate representation of the desired audio, bridging the gap between text and audio.The LDM, on the other hand, is trained on a large corpus of audio waveforms. It specializes in generating realistic audio by transforming high-dimensional noise into coherent sound. TANGO unifies these two models by feeding the intermediate representation produced by the LLM into the LDM. This integration guides the LDM in generating audio that is consistent with the input text. The LDM gradually refines the audio quality by reducing noise and introducing high-frequency details, resulting in more natural and accurate audio generation.

9. **Towards Building the Federated GPT: Federated Instruction Tuning Zhang *et al.* (2023b)**

   The paper revolves around the need for diverse and high-quality instruction data to train LLMs effectively. Traditional centralized training methods often require a centralized repository of such data, which can be problematic for several reasons, including privacy concerns and the logistics of data collection.

   The proposed solution, FedIT, leverages federated learning to train LLMs using distributed instruction data. This approach is rooted in two key steps. The first involves local instruction tuning, where each device independently refines a copy of the LLM using its locally available instruction data. The second step, model aggregation, merges the insights and knowledge acquired from each device's local instruction data. This collaborative learning approach ensures that privacy is preserved since the underlying data is not shared. Furthermore, it reduces the reliance on a central repository of data, making it a more scalable and versatile approach. To validate the effectiveness of FedIT, the authors conducted evaluations on a range of instruction-following tasks, including dialogue, summarization, and question answering. The results of these evaluations demonstrated that FedIT outperforms traditional centralized training methods.

10. **COEDIT: Text Editing by Task-Specific Instruction Tuning Raheja *et al.* (2023)**:

    The paper introduces an approach to text editing using natural language processing (NLP) models. Text editing is a fundamental step in the writing process, and NLP models can play a crucial role in providing automated editing suggestions to assist human writers. However, traditional NLP-based editing methods often face challenges related to data requirements and generalization to different editing instructions and styles.

    To overcome these challenges, authors propose COEDIT, a text editing model that leverages task-specific instruction tuning. This approach involves fine-tuning a pre-trained language model on a diverse set of task-specific instructions for text editing. These instructions guide the model towards specific editing behaviors, such as simplifying a sentence or changing its style to be more neutral.

    The authors evaluate COEDIT across various text editing benchmarks, including CoLA, GLUE, and WMT datasets. Their findings are significant. First, COEDIT achieves high performance in text editing with minimal labeled data, which is a crucial efficiency improvement, reducing the need for extensive data annotation efforts. Second, it outperforms existing state-of-the-art editing models while being significantly smaller in size. Third, COEDIT exhibits strong generalization capabilities, effectively handling editing instructions and styles it has not seen during training.

11. **Otter: A Multi-Modal Model with In-Context Instruction Tuning Li *et al.* (2023a)** The paper highlights the multimodal issues of LLMS that combine both text and visual information. The authors propose a new multi-modal model called "Otter" that leverages in-context instruction tuning to enhance its ability to follow instructions that span both text and visual domains.

    Otter uses in-context instruction tuning, a technique that involves fine-tuning a model on a dataset comprising instruction-response pairs that encompass both text and visual information.

    Otter builds upon OpenFlamingo, an open-sourced version of DeepMind's Flamingo model, specifically designed for multi-modal tasks. This foundation contributes to Otter's effectiveness in handling various instruction-following benchmarks.

    The paper reports comprehensive evaluations of Otter's performance on a range of multi-modal instruction-following benchmarks, including VQA, GQA, and Visual Dialog. Otter consistently outperformed state-of-the-art methods in these benchmarks, affirming the effectiveness of in-context instruction tuning for multi-modal instruction following.

12. **Recommendation as Instruction Following: A Large Language Model Empowered Recommendation Approach Zhang *et al.* (2023c)** Traditional recommendation systems have long relied on techniques such as collaborative filtering and content-based filtering to provide users with product, movie, music, and content suggestions. However, these methods often struggle to capture the intricate nuances of user preferences and context. In response to these limitations, authors consider recommendation as a task of instruction following by LLMs.

    The core idea of this approach is to convert user preferences and contextual information into natural language instructions. These instructions are then executed by an LLM to generate personalized recommendations. This paradigm effectively bridges the communication gap between users and recommendation algorithms.

    The paper introduces a structured instruction format, encompassing user preferences, intention, task form, and context. To facilitate the evaluation and effectiveness of their approach, the authors designed 39 instruction templates, leading to the generation of a substantial dataset of 252,000 user-personalized instructions, covering a wide range of preferences and intentions.

    To substantiate the superiority of their approach, the researchers instantiated these instruction templates into well-established recommendation tasks and conducted comprehensive experiments on real-world datasets. The results show that their instruction-following approach consistently outperformed competitive baselines, including the formidable GPT-3.5 model.

13. **LLM-Blender: Ensembling Large Language Models with Pairwise Ranking and Generative Fusion Jiang *et al.* (2023)**

    The paper displays that LLMs display variations in performance across tasks and examples. This inconsistency arises from the fact that different LLMs may excel in specific areas while struggling in others.

    LLM-Blender is designed to leverage the diverse strengths of multiple LLMs, leading to a more consistent and superior overall performance.LLM-Blender comprises two primary components: PairRanker employs pairwise ranking to distinguish subtle differences between candidate outputs from multiple LLMs. It uses a specialized pairwise comparison method that simultaneously encodes the input text and a pair of candidate outputs, enabling the system to determine the superior candidate. GenFuser is the generative fusion module. It combines the top-ranked candidates from PairRanker's pairwise comparisons into a fused output. This approach capitalizes on the strengths of the highly ranked candidates while mitigating their weaknesses, ultimately leading to improved output quality. LLM-Blender addresses the problem of inconsistent LLM performance by combining the strengths of multiple LLMs. This ensembling approach results in more reliable and superior output quality. By leveraging the diverse capabilities of multiple LLMs, LLM-Blender can compensate for the individual limitations of each model.

14. **UnifiedSKG: Unifying and Multi-Tasking Structured Knowledge Grounding with Text-to-Text Language Models Xie *et al.* (2022)** The paper presents a framework designed to revolutionize the field of structured knowledge grounding (SKG). SKG tasks involve leveraging structured knowledge to fulfill user requests, encompassing a wide range of applications such as semantic parsing over databases and question answering over knowledge bases. The key challenge addressed by this paper is the heterogeneity of inputs and outputs in SKG tasks, which hinders systematic and compatible research in this domain.

    UnifiedSKG introduces an approach to SKG by unifying 21 different SKG tasks into a common text-to-text format. This unification facilitates systematic research across SKG tasks. The essence of this framework is the transformation of each SKG task into a text-to-text format, where a natural language query or instruction serves as the input, and a structured knowledge representation is the output. For instance, a question answering task can be represented as follows: the input is "What is the capital of France?" and the output is the structured knowledge representation "(country: France, capital: Paris)."

    The UnifiedSKG framework leverages a dataset comprising text-to-text transformations for all 21 SKG tasks. This dataset is constructed by automatically mining existing SKG datasets and converting them into text-to-text transformations. Once the framework is trained, it is used for performing a wide variety of SKG tasks without the need for fine-tuning. By providing a natural language query or instruction, UnifiedSKG can generate the corresponding structured knowledge representation.

    Authors conducted a comprehensive evaluation of UnifiedSKG on various SKG benchmarks. The results are remarkable, as UnifiedSKG outperforms state-of-the-art SKG models on the majority of these benchmarks. Moreover, the framework's ability to facilitate multi-task learning is highlighted, leading to further improvements in performance across most tasks.

15. **In-BoXBART: Get Instructions into Biomedical Multi-Task Learning Parmar *et al.* (2022)**

    The paper is an application of instruction learning for NLP field. The research introduces a novel and unified model for handling a multitude of biomedical tasks, known as In-BoXBART. This approach is trained on a meta-dataset encompassing 32 instruction-based tasks in the biomedical domain, spanning various categories such as classification, question answering, and summarization.

    The unification of these diverse tasks under a single model streamlines the learning process and empowers In-BoXBART to perform an array of biomedical NLP tasks more effectively and efficiently.

    The authors validate the effectiveness of In-BoXBART through evaluations on a spectrum of biomedical NLP benchmarks. In-BoXBART outperforms single-task baseline models by an average margin of around 3%. Furthermore, it exhibits improvements in few-shot learning settings, surpassing single-task baseline models by an average margin of approximately 23%.

16. **Zemi: Learning Zero-Shot Semi-Parametric Language Models from Multiple Tasks Wang *et al.* (2022d)**

    The paper is designed to enhance the zero-shot performance of semi-parametric language models.

    The central concept behind Zemi is the amalgamation of parametric and non-parametric components within a language model. Zemi augments the LLM's parametric core with a retrieval system that can efficiently extract relevant documents from a vast unlabeled corpus. This integration allows Zemi to learn from a dataset of text, significantly broader than what traditional parametric models can handle.

17. **Crosslingual Generalization through Multitask Finetuning Muennighoff *et al.* (2023)**

    The paper introduces a method to enhance crosslingual generalization for large language models (LLMs). This approach fine-tunes an LLM on a diverse set of tasks across multiple languages, utilizing prompts to help the model understand the relationships between these languages.

    The significance of this work lies in its potential to achieve crosslingual generalization without compromising performance on monolingual tasks. Practical applications of MTF include training LLMs for translation tasks without the need for parallel text data, enhancing LLM performance on multilingual tasks, such as cross-lingual question answering, and making LLMs more user-friendly for individuals who speak multiple languages. The experimental results highlighted in the paper showcase the effectiveness of MTF, particularly when applied to models like BLOOM and mT5. The ability to generalize from English tasks with English prompts to non-English languages solely present in the pretraining corpus demonstrates the method's power. The research further explores how finetuning on multilingual tasks with machine-translated prompts, aligned with the language of the dataset, can enhance model performance.

18. **Task-aware Retrieval with Instructions Asai *et al.* (2022)**

    The paper leverages human-written instructions to guide the search behavior of LLMs, making them more adept at understanding user intent and retrieving relevant documents. The authors conducted evaluation of TART, focusing on two zero-shot retrieval benchmarks, BEIR and LOTTE. T They introduce of a new evaluation setup known as $X^2 - Retrieval$. Unlike traditional evaluation setups, $X^2 - Retrieval$ reflects real-world scenarios where diverse domains and tasks are mixed. In this more complex setting, TART continued to significantly outperform competitive baselines, emphasizing its real-world applicability and effectiveness in instruction-guided retrieval.

19. **UnifiedABSA: A Unified ABSA Framework Based on Multi-task Instruction Tuning Wang *et al.* (2022e)**

    Traditional ABSA methods often rely on separate models for different subtasks within ABSA, such as aspect term extraction, sentiment classification, and opinion target extraction. This fragmented approach necessitates substantial labeled data for each subtask, making it time-consuming and resource-intensive. The authors present a unified ABSA framework grounded in multi-task instruction tuning. The core idea behind UnifiedABSA is to employ a single model capable of handling all ABSA tasks while harnessing multi-task learning to exploit shared knowledge across tasks. They incorporate natural language instructions to guide the learning process, making it adaptable to diverse ABSA tasks without the need for extensive parameter updates.

    Using this social media sentiment analysis becomes more effective, enabling the understanding of public sentiment toward specific topics or brands. Moreover, it can enhance customer service improvement by analyzing feedback to identify areas for enhancement.

20. **Improving Cross-task Generalization of Unified Table-to-text Models with Compositional Task Configurations Chen *et al.* (2022)**

    LLMs are designed to handle an array of tasks that involve tables and text, such as table summarization, question answering over tables, and table generation. However, the challenge lies in enabling these models to effectively generalize across this diverse set of tasks, ensuring their versatility and adaptability.

    To tackle this challenge, the authors introduced "compositional task configurations." These configurations consist of a set of prompts that are prepended to the encoder of a unified table-to-text model. These prompts serve the crucial role of explicitly specifying various task-related parameters, including the task type, dataset name, input type, and output type. By providing the model with clear and structured information about the specific task at hand, the compositional task configurations aim to improve the model's cross-task generalization.

21. **MultiInstruct: Improving Multi-Modal Zero-Shot Learning via Instruction Tuning Xu *et al.* (2022b)**

    The paper introduces an approach that leverages instruction tuning to enhance the performance of LLMs in multi-modal zero-shot learning tasks. MultiInstruct provides explicit guidance to LLMs on how to effectively combine information from different modalities, facilitating better task understanding and improved performance. The research team evaluated MultiInstruct across various benchmarks, such as VQA-X, GQA, and ImageCaption.

22. **Exploring the Impact of Instruction Data Scaling on Large Language Models: An Empirical Study on Real-World Use Cases Ji *et al.* (2023)**

    The research paper studies the relationship between the amount of instruction data and LLM performance across a diverse set of real-world tasks.

    The primary findings of this empirical study are:

    - Data Quantity and Performance: The relationship between data quantity and LLM performance. For open-ended generation tasks, the paper highlights that increasing the amount of instruction data leads to continuous improvements in LLM performance.
    - Structured Task Insights: In contrast, for structured tasks like mathematical problem-solving and code generation, the paper notes a plateaued performance. As more instruction data is added, the model's performance curve remains relatively flat. This suggests that for structured tasks, there might be a ceiling of performance that LLMs can reach with a moderate amount of instruction data.
    - Specialized Training and Quality Data: The study emphasizes the importance of data selection and specialized training methods for hard tasks. Instead of merely increasing data size, the focus should shift towards effectively selecting high-quality training data. Additionally, the authors point to the need for training methods that cater specifically to challenging tasks.

23. **Towards Better Instruction Following Language Models for Chinese: Investigating the Impact of Training Data and EvaluationJi *et al.* (2023)**

    The paper studies the challenges faced by large language models (LLMs), particularly in following instructions effectively, with a specific focus on the Chinese language.

    The paper delves into the pivotal role that data quality and diversity play in the performance of instruction following language models (IFLMs). It emphasizes that using carefully curated and diverse training data significantly enhances the ability of IFLMs to follow instructions and execute various tasks.

    The study explores the hurdles related to domain adaptation and suggests strategies to enhance adaptability. By incorporating domain-specific training data and infusing domain knowledge into the IFLMs' training process, the researchers propose methods to improve performance in specific domains.

24. **Chinese Open Instruction Generalist: A Preliminary Release Zhang *et al.* (2023a)**

    The paper presents development LLMs and instruction tuning capabilities for Chinese, which come with their unique linguistic and cultural intricacies.

    COIG does training on a massive dataset that pairs Chinese text with corresponding instructions. This training process is fundamental in endowing COIG with the ability to effectively comprehend instructions, access a wealth of knowledge, and engage in reasoning and inference to fulfill the given task.

    COIG has ability to understand the intent and context of instructions, granting it the capability to generate contextually appropriate responses.

25. **Adding Instructions during Pretraining: Effective Way of Controlling Toxicity in Language Models Prabhumoye *et al.* (2023)**

    The research paper addresses a challenge faced by large language models (LLMs) — their propensity to generate toxic content. To tackle this issue, the authors introduced two novel pretraining data augmentation strategies, MEDA and INST. These strategies involve adding instructions to pretraining samples, providing guidance to the LLM regarding the toxicity of the content. Essentially, the instructions indicate whether the text should be toxic or non-toxic. Through these strategies, the LLM learns to associate toxic language with negative feedback and non-toxic language with positive feedback, thereby enabling it to control and reduce the generation of toxic content.