

Towards reliability and interactive debugging for large language models

Bhargavi Paranjape

A dissertation

submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington
2024

Reading Committee:
Hannaneh Hajishirzi, Co-Chair
Luke Zettlemoyer, Co-Chair
Zachary Chase Lipton

Program Authorized to Offer Degree:
Computer Science and Engineering

© Copyright 2024

Bhargavi Paranjape

University of Washington

Abstract

Towards reliability and interactive debugging for large language models

Bhargavi Paranjape

Co-chairs of the Supervisory Committee:

Hannaneh Hajishirzi

Computer Science and Engineering

Luke Zettlemoyer

Computer Science and Engineering

Large language models (LLMs) have permeated our everyday lives and are used in critical decision-making scenarios that can affect millions of people. Despite their impressive progress, model deficiencies may result in exacerbating harmful biases or lead to catastrophic failures. In this thesis, we present and advance a series of important considerations for reliable model deployment. Beyond improved accuracy on new and complex tasks, users want more transparent models that *explain their predictions* and are *robust* to data biases or distributional shifts. They also want to be equipped to interact with these models to better understand and debug them. We present a variety of training and inference techniques toward building these aspects of reliability into models. We particularly focus on techniques that address challenges of scale and lack of human supervision, for models ranging from classifiers with limited interaction potential to massive LLMs that can communicate with humans and external tools.

In the first part of this thesis on advancing explainability for LLMs, we introduce a novel information-theoretic objective to train models to generate explanations that are concise, comprehensible and faithful to model predictions. We also introduce a contrastive prompt-based approach to explain model predictions on common-sense reasoning tasks, that can also be leveraged by users to probe model behavior. We focus on distributional robustness for LLMs in the second part of this thesis. We develop a novel optimization technique, that discovers error-prone data slices for users to examine, and trains a robust classifier to improve

performance on rare data slices. We also develop an open-sourced framework for fine-grained attribution of hallucinations in model generated text to underlying pre-training data. In the third part, we present a framework for automatically decomposing unseen composite tasks that require multi-step reasoning and external-system interaction, and delve into how the framework supports user debugging. Overall, this thesis presents a range of optimization, inference, and evaluation methods that make progress toward better explainability, robustness, and interactive debugging of large language models.

Acknowledgements

An innumerable number of people have been incredibly helpful during my PhD. Acknowledging everyone in a short section is going to be a challenge.

First and foremost, I'd like to thank my advisers Luke Zettlemoyer and Hannaneh Hajishirzi. Apart from being insightful and brilliant, their defining traits as advisers were their personal attention and kindness. Despite leading groups at multiple institutions, both Luke and Hanna always made themselves available to discuss any and every aspect of my research and professional life. I am grateful to Hanna for helping me build strong fundamental research skills. I am especially grateful for her ability to help convert research ideas into well-rounded projects and high-caliber paper submissions. Luke has helped me think about bold ideas, find the sweet spot between impact and novelty, and communicate my ideas clearly and concisely. While reading my papers for this thesis, I was able to chart how Luke's paper-writing techniques have become ingrained into how I present ideas! Together, they complemented each other in their advising styles and their guidance has helped me grow as an independent researcher. Without their generosity, my PhD would have been immensely stressful.

I have been lucky to have had mentors who were willing to dig into the details of research with me. I worked closely with Luke's senior PhD students, Mandar Joshi and Julian Michael, for the first couple of research projects. Both of them strongly influenced my taste and approach in research problems. I enjoyed brainstorming ideas over coffee with Mandar and debating how relevant computational semantics is to modern NLP with Julian. I have enjoyed brainstorming my rough research ideas and discussing technical problems with the absolutely brilliant and kind members of Luke and Hannah's lab—Sewon Min, Yizhong Wang, Akari Asai, Gabriel Ilharco, Tim Dettmers, Gary Liu, Ellen Wu, Sachin Mehta, Margaret Li, Victor Zhong, Terra Blevins, Suchin Gururangan, Artidoro Pagnoni, and Weijia Shi.

A large part of my PhD has also involved industry internships, which have been extremely fulfilling and instrumental in helping me become a more effective researcher. My first internship experience at Meta, hosted by Marjan Ghazvininejad, was a wonderful foray into research in an industry lab. My internship experience at Google with Ian Tenney and Matthew Lamm was equally wonderful. Ian, who I worked with very closely, was an exceptional mentor, a meticulous researcher, and helped me navigate the complex work environment at Google. I loved the nurturing and innovative culture at AI2 while working on multiple projects with Pradeep Dasigi, Vivek Srikumar, Yanai Elazar, Hao Peng, Khyati Chandu, and Sarah Weigrefe. My most grueling but intellectually rewarding internship was working with Marco Tulio Ribeiro, Sameer Singh, and Scott Lundberg at MSR. Marco helped me think more critically and clearly about research problems, and gave me invaluable advice on how to approach research more methodically. My last internship with Koustuv Sinha at Meta was challenging because of the concurrent job-search, and Koustuv was an excellent mentor who helped me navigate both goals.

One of the most fun parts of my grad school experience was being able to collaborate with some wonderful people on research projects along the way: Qingqing Cao, Srini Iyer, John Thinkstun, Kushal Lakhotia, Asish Ghoshal, Yashar Mehdad, Haoran Li, and Wen-tau Yi. I also immensely enjoyed mentoring undergraduate Tanay Dixit and PhD student Daniel Kim. Witnessing Tanay succeed in publishing his first paper at EMNLP and securing grad school admits was a very rewarding experience. Towards the end of my PhD, I received help with interview prep, introduction to people during my job search, words of kindness, and so much more, from Victor Zhong, Gagan Bansal, Sachin Mehta, Colin Lockard, Shrimai Prabhumoye, Srividya Pranavi Potharaju, Alankar Jain, Monojit Choudhary, Vasu Sharma, and Spandana Gella.

I am also immensely thankful to fellow PhD students at UW and Carnegie Mellon, who made my journey enjoyable—Gagan Bansal, Srini Iyer, Sachin Mehta, Divyansh Koushuik, Sanket Vaibhav Mehta, Danish Pruthi, and Vidhisha Balachandran. Having their support during tough times, including a global pandemic, has been a privilege. My PhD experience was so much smoother because I could always depend on the CSE staff in the event of hiccups. I owe a lot of gratitude to Elise deGoede Dorough, Elle Brown, and Joe Eckert for making sure I was able to navigate rules around internships and graduation. I also want to thank Chiemi Yamaoka-Vismale for taking care of my travel and equipment related issues through my PhD. My thesis committee members Zack Lipton and Shane Steinert-Threlkeld were instrumental in helping me put together my thesis.

While my support system in Seattle kept me going, it's the folks back home in India who were my inspiration all along. Several people in my life have helped me remain patient and energetic throughout grad school—my undergrad friends Swati Saini and Anjali Goyal, and my extended family of aunts, uncles, cousins, and in-laws. Throughout my life, my family has pushed me to become the best version of myself, both personally and professionally. I did not anticipate that the person I would co-author my first PhD paper with would end up becoming my co-author for life. I am grateful to Mandar for his wisdom and support during the tough times in the last couple of years. Pinak has been there for me throughout this journey, inspiring me to achieve my goals and set an example for him. Baba was instrumental in planting the seed for scientific inquiry in me from a very young age. I've grown up wanting to earn a PhD and become a scientist like him. I have been able to share every small fear and shed tears of frustration and sadness before Aai. She has been an emotional pillar for me since I left home at 18. To my parents, I owe a world of gratitude.

DEDICATION

To my family

Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 19 |
| 1.1 | Overview | 19 |
| 1.2 | Background | 21 |
| 1.3 | Common Challenges | 27 |
| 1.4 | Approaches | 29 |
| 1.5 | Thesis Outline | 30 |
| I | Explainability for large language models | 33 |
| 2 | Concise extractive explanations with information bottleneck | 33 |
| 2.1 | Overview | 33 |
| 2.2 | Methodology | 34 |
| 2.3 | Model | 37 |
| 2.4 | Experimental Setup | 38 |
| 2.5 | Results | 39 |
| 2.6 | Discussion | 41 |
| 3 | Contrastive explanations for large language models | 43 |
| 3.1 | Overview | 43 |
| 3.2 | Methodology | 44 |
| 3.3 | Model | 46 |
| 3.3.1 | Contrastive Templates | 46 |
| 3.3.2 | Generating Explanations | 47 |

| | | |
|------------|--|-----------|
| 3.3.3 | Task Model | 48 |
| 3.4 | Experimental Setup | 48 |
| 3.4.1 | Baselines | 48 |
| 3.4.2 | Implementation details | 49 |
| 3.5 | Experimental Results | 50 |
| 3.5.1 | Task Performance | 50 |
| 3.5.2 | Human Evaluation | 50 |
| 3.6 | Discussion | 52 |
| II | Robustness to distributional shifts and hallucinations | 53 |
| 4 | Distributional robustness to spurious correlations | 53 |
| 4.1 | Overview | 53 |
| 4.2 | Method | 55 |
| 4.2.1 | AGRO: Adversarial discovery of Group for Robust Optimization | 56 |
| 4.2.2 | Features for Group Discovery | 57 |
| 4.3 | Experimental Setup | 58 |
| 4.4 | Results | 59 |
| 4.4.1 | Discussion | 60 |
| 5 | Factuality for large language models | 63 |
| 5.1 | Overview | 63 |
| 5.2 | Methodology and Results | 64 |
| 5.2.1 | Claim Generation | 65 |
| 5.2.2 | Evidence Retrieval | 67 |
| 5.2.3 | Claim Verification | 68 |
| 5.3 | Discussion | 72 |
| III | Interaction with external systems | 73 |
| 6 | Tool Augmentation and Complex Reasoning | 73 |

| | |
|--|------------|
| <i>CONTENTS</i> | 11 |
| 6.1 Overview | 73 |
| 6.2 Method | 75 |
| 6.2.1 Task Library | 76 |
| 6.2.2 Tool Library | 77 |
| 6.2.3 Human feedback | 78 |
| 6.3 Experimental Setup | 79 |
| 6.4 Results | 79 |
| 6.4.1 Results on the task library | 80 |
| 6.4.2 Test tasks (cross-task transfer) | 81 |
| 6.5 Discussion | 82 |
| 7 Conclusion | 85 |
| A Appendix | 107 |

List of Figures

| | | |
|-----|---|----|
| 1.1 | Solving a math problem by interacting with external systems like OpenAI Codex (Chen et al., 2021) and Python code interpreter. | 26 |
| 2.1 | Our Information Bottleneck-based approach extracts concise rationales that are minimally informative about input, and maximally informative about the label | 34 |
| 2.2 | Semi-supervised experiments showing the task performance for varying proportions of rationale annotation supervision on the MultiRC, FEVER, and Movies datasets. | 40 |
| 3.1 | (1) A commonsense reasoning instance (c, a_1, a_2) is converted into a custom prompt $c_{a_0} \oplus t_{a_1, a_2}$ as input for the explainer PLM (2) The combination of input and explanation $(c_{a_i} \oplus e_j)$ is used by task model to score $a_i \forall i \forall j$. For a_1 and a_2 , scores are aggregated over templates. | 46 |
| 4.1 | Groups discovered by AGRO on CelebA image classification dataset and MultiNLI sentence pair classification dataset. | 54 |
| 6.1 | A run-through of ART on a new task, Physics QA. (A) Programs of related tasks like anachronisms and Math QA provide few-shot supervision to the LLM — related sub-steps and tools in these programs can be used by the LLM for cross-task generalization (shown in purple). (B) Tool use: Search is used to find the appropriate physics formula, and code generation and execution are used to substitute given values and compute the answer (shown in orange). | 74 |
| 6.2 | Human feedback to ART shown for (a) PQA where reasoning steps are added to the program and; (b) Word unscrambling where tool library is augmented with a new lookup tool. | 77 |

List of Tables

| | | |
|-----|---|----|
| 1.1 | Extractive and generative explanations (rationales) for representative NLP tasks | 23 |
| 1.2 | Spurious correlations in natural language inference (MNLI dataset): Premise (P) and hypothesis (H) contain antonym words, correlated with the contradiction class | 25 |
| 2.1 | Task, Rationale IOU F1 (threshold set to 0.1) and Token F1 for our hard-attention Sparse IB approach and baselines on test sets, averaged over 5 random seeds. Gold IOU and token F1 are 100.0. We use 25% training data in our semi-supervised setting (Section 2.3). † Reported Results ^ε We could not reproduce numbers for the Bert-to-Bert supervised method reported in DeYoung et al. (2019). | 39 |
| 3.1 | Examples of commonsense tasks that can be explained using contrastive language and some contrastive explanations authored by in-house annotators. The Fact and Foil are marked in the input. | 45 |
| 3.2 | Test set accuracy on Winogrande (WGRD), PIQA, WSC and Winogender (WGND). ZS is Zero-shot models while FT is fine-tuned models. WSC and Winogender don’t have training data for finetuning. Across all our models, the task model is GPT2-XL for fair comparison with Shwartz et al. (2020) and to make finetuning tractable. | 49 |
| 3.3 | Human Evaluation Results on Winogrande(WGRD) and PIQA. Reported human evaluation results on Self-talk are different from ours, which can be because of moderate levels of agreement (Fleiss Kappa $\kappa = 0.43$). We report average and deviation across 3 different samples of 50 examples. Grammaticality is judged together for all datasets following Shwartz et al. (2020). Only contrastive explanations can be flipped. | 51 |

| | | |
|-----|---|-----|
| 3.4 | Flipped Evaluation Results for contrastive explanation models. Reporting test accuracy across all datasets. % Drop in performance as a result of flipping is indicated in parentheses. | 51 |
| 4.1 | Inputs, labels and known worst-groups in WILDS datasets. | 58 |
| 4.2 | Average and WG performance on WILDS datasets. AGRO is better at multiple worse groups and also consistently improves over ERM, unlike prior work. Best KWG performance bolded and second-best underlined. (*) indicates variance across 3 random seed runs. | 59 |
| 4.3 | Perturbation patterns on worst groups discovered in AGRO. Class flips in () | 61 |
| 5.1 | Claim generation with and without finetuning LLama2. Finetuning helps improve claim recall for longer generations and prevents right-branching effect. | 66 |
| 5.2 | Evaluating claim generation: Instruction tuning of open-models helps bridge the gap and even surpass performance of few-shot prompting with ChatGPT | 67 |
| 5.3 | Comparing different retrieval methods (Rouge-L on top-1 retrieval.) on claim-evidence datasets. | 68 |
| 5.4 | Reasoning data annotation: Annotators can correct both fact verification labels and associated reasoning by evaluating the claim against the reference chunk. | 70 |
| 5.5 | Evaluating attribution: Accuracy of label prediction with majority voting over claims if per-claim label is not available. Retrieved documents are from Pyserini-Wiki. | 70 |
| 5.6 | OpenFactEval for ExpertQA: Claim generation and reasoning-based attribution for individual claims. OpenFactEval labels each claim as Attributable , Contradictory and Extrapolatory | 71 |
| 6.1 | ART performance on BigBench tasks and MMLU tasks. (¹ Human-crafted CoT (Wei et al., 2022b; Suzgun et al., 2022), ⁵ InstructGPT (Ouyang et al., 2022), ⁶ Scaled instruction finetuning (Chung et al., 2022), ⁷ Code-davinci-002 (Chen et al., 2021)). | 80 |
| 6.2 | Improving ART and free-form CoT via human-in-the-loop feedback. (*) indicates that human-in-the-loop improvement was done over automatically generated CoT reasoning for these tasks. Feedback for ART includes correcting sub-steps in programs (“C:”), adding additional sub-steps(“A:”), and defining new tools(“T:”). Note that only five examples were edited for each task. | 82 |
| A.1 | Contrastive Patterns and Examples of outputs generated by the T5-large model. The pattern the PLM completes are marked \implies | 107 |

A.2 Programs in the task library for tasks requiring string manipulation. 108

A.3 Programs in the task library for tasks requiring arithmetic operations. 109

A.4 Programs in the task library for tasks requiring search for world knowledge. 110

A.5 Programs in the task library for tasks requiring code generation or editing. 111

A.6 Programs in the task library for tasks requiring free-form chain-of-thought style reasoning
about logic and linguistics. 112

Chapter 1

Introduction

1.1 Overview

Machine learning and natural language processing have undergone a significant paradigm shift with the rise of models like BERT (Devlin et al., 2019), GPT-4 (OpenAI, 2023), and DALL-E (Ramesh et al., 2021), which are trained on broad data (self-supervision at scale). This paradigm shift has led to tremendous improvements in performance and sample efficiency on a range of downstream tasks. Their wide-ranging capabilities (e.g., language, vision, robotic manipulation, reasoning, human interaction) have resulted in a substantial proliferation of their applications. These models are actively being used in various critical domains, including healthcare, cybersecurity, education, and law enforcement. Large pre-trained models for language and vision are also being used by millions in user-facing applications like search engines and virtual assistants. Every day, millions of users actively engage with models like ChatGPT and Gemini (through BARD) for various tasks, ranging from generating innovative recipes to collaborative pair programming. This interaction is expected to undergo significant expansion in the near future.

The reliable deployment of large language models (LLMs) and other large pre-trained models is of paramount importance due to their significant impact on critical areas of human life (Bommasani et al., 2021). In this thesis, I argue that this escalating interaction between language models and humans necessitates addressing the following primary user concerns that will help engender their trust in the deployed model.

- **Sample-efficient generalization:** Users expect the model to effectively generate accurate responses for a multitude of heterogeneous and previously unseen tasks. These tasks may often encompass intricate

multi-step reasoning and interaction with external systems and tools like search engines and code execution.

- **Transparency about model predictions:** LLMs are black boxes, with billions of un-interpretable parameters. Model predictions should be explainable, in a manner that is human-understandable and faithful to the model’s prediction.
- **Robustness to distributional shifts:** Model pre-training focuses on maximizing the average likelihood of predicting the subsequent token, resulting in an amplification of biases and correlations present in the training data. Users desire models that exhibit robustness in the face of such correlations in the training data distribution, which may otherwise lead to biased predictions, hallucinations, and factuality errors in generated text.
- **Interactive debugging:** Users want enhanced interaction with the model, specifically aimed at troubleshooting i.e. they want to be able to probe the model to identify its areas of weakness and subsequently improve its performance.

This thesis makes progress toward building ML models and frameworks that improve the explainability, robustness, and interactive capabilities in neural models for NLP, with particular emphasis on large language models (LLMs). Specifically, we build models and frameworks that provide reliable explanations for their predictions, are robust to spurious data correlations and distributional shifts, and can tackle complex tasks via external system interactions. Furthermore, our work also advances the understanding of human-in-the-loop interactive debugging along multiple fronts of reliability—interaction with model-generated explanations, interaction with examples containing spurious correlations for model diagnostics, and debugging of LLM-external system interactions.

Towards explainability, our work focuses on methods that can extract explanations from text (Paranjape et al., 2020) and generate textual explanations for model predictions (Paranjape et al., 2021b), on a diverse range of downstream NLP tasks. In particular, our work on explainability focuses on designing explanation formats that require minimal human supervision, are human-comprehensible, and meet user preferences for explainable AI (such as conciseness and contrast, (Miller, 2018)). In addition, we explore the interactive debugging of model behavior via explanation perturbation in Paranjape et al. (2021b).

LLMs are affected by noisy correlations between text spans in training data (Lovering et al., 2020), leading to hallucinations and factual errors (Wang et al., 2023), predictive biases (Sagawa et al., 2019), and amplification of social biases (Webster et al., 2020). As LLMs typically process large-scale internet

data, detecting such correlations and isolating instances of model failure can be challenging (Wang, 2021). Furthermore, it is important for robustness techniques aimed at addressing these concerns to allow users, who serve as the final decision-makers, to interpret and take action on identified sources of spurious correlations (SCs) and model biases (Eisenstein, 2022). In the second part of this thesis, we propose an optimization technique for distributional robustness to identify groups of examples that potentially contain spurious correlations (Paranjape et al., 2022). We demonstrate how users can perturb examples in these groups to identify recurring biases in models' predictive behavior. Dataset bias also results in hallucinations and factual inconsistencies in model-generated text and attributing (tracing back) these failures to internet-scale training data is a challenge. We develop a detailed attribution report specifically tailored for long text generated by LLMs.¹ This report provides in-depth attribution information for atomic facts (Min et al., 2023) in the generated text, utilizing a web-scale text corpus as the grounding source.

Lastly, LLM interactions have become increasingly complex (Xi et al., 2023). They now exist within an ecosystem that necessitates their communication with external systems, referred to as "tools," particularly while solving intricate multi-step reasoning tasks that current LLMs struggle to handle on their own. These models may engage with APIs (Patil et al., 2023), code compilers and interpreters (Gao et al., 2022), calculators (Schick et al., 2023), and search engines (Press et al., 2022) to address complicated user prompts. Planning and executing appropriate, accurate, and safe tool interactions is a crucial part of *reliable interaction* within this ecosystem. In this thesis, we explore the native planning and tool-use capabilities of LLMs through few-shot in-context learning (Paranjape et al., 2023). Additionally, we explore how these plans can be examined and improved through human-in-the-loop debugging.

In the concluding chapter, this thesis also explores prospective avenues for enhancing the reliability of LLM-based applications. Specifically, we discuss crucial research directions for reliability in models that enable multimodal and embodied interactions, as well as personalized models tailored to individuals and organizations.

1.2 Background

Large language models and their emergent abilities NLP has experienced a major shift in recent years with the introduction of deep neural models for text processing and the availability of large text datasets for

¹ <https://drive.google.com/file/d/1FcGWUf7vOKxPF7EE5zvmgdMLPEGjmsyV/view?usp=sharing>

training. In particular, self-supervised training on web-scale data using language modeling objectives (Dai and Le, 2015) and downstream transfer learning has had a significant impact on the field, resulting in notable advancements in benchmark performance.

This has led to the development of context-aware models like GPT (Radford et al., 2019), ELMo (Peters et al., 2018), and ULMFiT (Howard and Ruder, 2018). The next advancements in self-supervised learning, exemplified by BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019), RoBERTa (Liu et al., 2019), T5 (Raffel et al., 2020), and BART (Lewis et al., 2020), involved the rapid adoption of the Transformer architecture (Vaswani et al., 2017), the incorporation of more powerful deep uni- and bi-directional sentence encoders, and the scaling up of models and datasets.

The proliferation of pre-trained models has led to surprising developments resulting from their scale. GPT-3 (Brown et al., 2020), with its 175 billion parameters compared to GPT-2’s 1.5 billion, enables in-context learning, where the language model can adapt to a specific task by simply providing it a prompt (a natural language description of the task). This emergent property was neither explicitly trained for nor anticipated (Wei et al., 2022a). Beyond scaling, Wei et al. (2021); Sanh et al. (2021) also find that instruction-based finetuning on sufficiently large and diverse downstream tasks further improves models’ emergent few-shot generalization and instruction-following abilities. In addition, scale and instruction-based finetuning also unlock models’ abilities to produce a sequence of intermediate steps before generating the final answer— a *chain-of-thought* (Cobbe et al., 2021; Wei et al., 2022b; Suzgun et al., 2022). Finally, Ouyang et al. (2022); Gozalo-Brizuela and Garrido-Merchan (2023); Touvron et al. (2023) have proposed reinforcement learning from human feedback, further improving the model’s ability to generate responses that are aligned with human preferences and adhere to established safety constraints.

Reliability in large language models The study of reliability in large language models (LLMs) is imperative due to their integration into real-world AI systems, with far-reaching consequences for billions of people. Notably, foundation models such as BERT are utilized by Google search, reaching billions of users. Similarly, OpenAI’s ChatGPT, used by millions, is being integrated into critical applications like healthcare and financial advising chatbots. There are several aspects to the problem of reliable deployment that are closely tied to *social impact* at scale (Bommasani et al., 2021): including the potential exacerbation of social inequities (fairness), and the potential concerns of amplifying disinformation (misuse). Bommasani et al. (2021) identify several technological and research challenges that are directly tied to these societal implications—their inherent lack

of interpretability (explainability) and robustness (including the lack of factuality in generations). In addition to these characterized challenges, the emergent capability of LLMs to interact with external systems like APIs and code executors has also led to another emerging area of concern for reliability—appropriate and safe interaction with external tools and models (Mialon et al., 2023). In this thesis, we particularly focus on important research challenges in explainability, robustness, and interaction with external systems. In the following sections, we briefly discuss seminal and relevant prior work that has made progress towards these challenges, highlighting research gaps that we aim to address in this thesis.

| |
|---|
| Extractive rationale for text prediction tasks |
| Movie review: In this movie, ... Plots to take over the world. The acting is great! The soundtrack is run-of-the-mill, but the action more than makes up for it (a) Positive (b) Negative |
| Generative contrastive explanation for Winograd Schema commonsense reasoning |
| The party was more interesting and uplifting than the funeral because the __ was rigid. Explanation: Parties are for celebrating while funerals are for mourning Explanation: People wear colorful clothes at parties and black at funerals (a) party (b) funeral |

Table 1.1: Extractive and generative explanations (rationales) for representative NLP tasks

Explainability In models where billions of parameters interact in a complex and interconnected manner, generating an accurate, reliable, and comprehensible explanation for their predictions is a significant challenge. Previous methods have focused on post-hoc explanations like LIME and SHAP (Ribeiro et al., 2016; Lundberg and Lee, 2017), or attention analysis (Luong et al., 2015; Wiegrefe and Pinter, 2019). However, these feature importance-based techniques are found to be less trustworthy (Serrano and Smith, 2019; Jain and Wallace, 2019). Methods that *condition* predictions directly on explanations (often, self-generated) are more trustworthy (Jain et al., 2020; Pruthi et al., 2022). Extractive rationalization (Lei et al., 2016), where a smaller sub-span of the input text is earmarked as a *rationale*, is one of the most well-studied of such methods (see Table 1.1 for an example). Supervised techniques for extractive rationalization (Camburu et al., 2018; DeYoung et al., 2019) incur considerable costs in obtaining reliable rationale annotations for every new dataset and task. In this thesis, we focus on the challenging scenario of extracting explanations in a *fully unsupervised manner*. To this end, we use the information bottleneck (Tishby et al., 1999) principle. Information Bottleneck (IB) optimizes for compression of input text/images into minimally informative hidden representations that are also accurately predictive of output labels. This principle has recently been adapted in several downstream

applications like parsing (Li and Eisner, 2019), extractive summarization (West et al., 2019), and image classification (Alemi et al., 2016; Zhmoginov et al., 2019). Alemi et al. (2016) and Li and Eisner (2019) use IB for optimal compression of hidden representations of images and words respectively. We leverage IB to compress *the number* of cognitive units (like sentences) to ensure interpretability of the bottleneck representation (in Chapter 2).

The extractive paradigm can be unfavorable for certain tasks like commonsense question answering, where the given input provides limited context for the task. The Winograd schemas example in Table 1.1 requires reading between the lines and using knowledge about entities like **party** and **funeral**, that is not explicit in the text. To circumvent this, recent work uses a free-form open-ended textual format to *generate* explanations for commonsense reasoning tasks like SNLI (Bowman et al., 2015), Winograd Schemas (Zhang et al., 2020), and CommonsenseQA (Rajani et al., 2019). These methods rely on explicit human supervision, which is inherently ambiguous, incomplete, and consequently, expensive to collect and evaluate on (Camburu et al., 2019b,a; DeYoung et al., 2019). Latacinnik and Berant (2020) use an unsupervised approach to generate free-form explanations as sequences of tokens that are not well-formed sentences. This is because extracting targeted knowledge from massive language models without explicit supervision is challenging. *Prompts* are increasingly being used to extract useful knowledge from PLMs in a targeted manner (Petroni et al., 2020; Richardson and Sabharwal, 2020; Talmor et al., 2020; Donahue et al., 2020; Lin et al., 2019) and in improving prediction on downstream tasks (Brown et al., 2020; Shin et al., 2020; Shwartz et al., 2020). In this thesis, we leverage prompting for explainability. However, unlike Shwartz et al. (2020) that ask follow-up questions in the prompt, we draw from social sciences research in explainability (Miller, 2019, 2018) for prompt design.

Distributional Robustness Distributional robustness is a general paradigm for assessing model generalization to out-of-train distribution examples, which often occur when models are deployed in real-world applications. Distributionally robust optimization (DRO) techniques (Ben-Tal et al., 2013; Duchi et al., 2016) and out-of-distribution data augmentation techniques (Liu et al., 2022a; Plumb et al., 2021; Kaushik et al., 2020) are the most well-studied and commonly employed techniques for distributional robustness. In particular, Group-DRO (Sagawa et al., 2019) is a popular variant of distributionally robust optimization that optimizes for robustness over various types of sub-population (group) shifts: label-shifts (Hu et al., 2018), shift in data-sources or domains (Oren et al., 2019), or test-time distributional shifts due to spurious correlations in training data (Sagawa et al., 2019). Spurious correlations in data are learned by models, resulting in poor

performance on a group of examples where the spurious correlation is absent (see Table 1.2 for an example in the natural language inference task). However, Group-DRO assumes that spurious correlations are known and group memberships for training examples are available for robust optimization. However, this is a strong assumption since it requires knowledge of spurious correlations in training data, which is not always available for large supervised datasets, new tasks, and pre-training data. Recent work that focuses on group discovery (Sohoni et al., 2020; Liu et al., 2021; Nam et al., 2020, 2022; Creager et al., 2021) rely on heuristics and strong assumptions about the dataset or task. Importantly, all these techniques are at least partially supervised, as they assume group memberships are available on a validation set for hyperparameter tuning or model selection. In this thesis, we focus on the *fully-unsupervised* setting for group distributional robustness. A related problem to robust optimization is automatic slice discovery (SD)—identifying underperforming data slices for error analysis of deep-learning models. Slice discovery methods (SDMs) can help determine patterns in model failures and make them human-comprehensible. In this thesis, we combine techniques from slice discovery and distributional robustness to jointly discover groups with spurious correlations and optimize for group distributional robustness.

| Common examples that contain the SC |
|--|
| P: it's slow it's uh there are many better machines on the market right now for H: This is the fastest machine, you won't find a better machine. Label: Contradiction P: 'You killed Derry.' H: Derry is still alive . Label: Contradiction |
| Uncommon examples that don't contain the SC (the distributionally shifted group) |
| P: Stand behind it. H: Stand so that it is in front of you. Label: Entailment P: Get there early in the morning before the crowds arrive H: You should get there shortly after they open to beat the crowds. Label: Entailment |

Table 1.2: Spurious correlations in natural language inference (MNLI dataset): Premise (P) and hypothesis (H) contain antonym words, correlated with the contradiction class

Factuality Long-form text generation has improved dramatically over the last few years, with the advent of closed-source LLMs like ChatGPT, GPT4 from OpenAI, Claude from Anthropic, etc. Open-source LLMs are also catching up fast, as evident with the LLama family of models (Touvron et al., 2023), Mistral (Jiang et al., 2023), etc. As text generations become exceedingly human-like and plausible, a critical objective is to ensure their generations are also factual, since many of these models are reported to produce convincing

hallucinations (Liu et al., 2023; Zhang et al., 2023). Hallucinations arise primarily from a combination of data sparsity (of rare knowledge) and over-representation of common knowledge in pretraining data (Mallen et al., 2022), resulting in text-span correlations that are often spurious. Recognizing these text-span correlations at scale and identifying hallucinations automatically can be a challenge. Consequently, the community needs to continually and critically evaluate model generations for factual correctness *post-hoc* and factuality evaluations can be both costly and time-consuming to do with human annotators. Thus, a concerted focus in the research community is to devise automatic ways to evaluate factuality of generations (Yue et al., 2023; Bohnet et al., 2022; Min et al., 2023; Chern et al., 2023). Automatic evaluation methods have typically employed: 1) retrieval of reference documents from a large external knowledge base, and 2) model-based evaluation, where an external model is tasked to compare the ground truth and model generation, generally by formulating the problem as natural language inference (NLI)(Bohnet et al., 2022; Yue et al., 2023). However, long-form text generations can have multiple facts to verify, and the generated text can have a mixture of correct and incorrect facts, making it hard to quantify factuality as a singular score. Min et al. (2023) propose decomposing the model generation into small atomic units and then performing model-based evaluation of factuality using a powerful *closed-source* LLM as the evaluator. However, usage of closed-source model APIs is not cheap—in our calculations, using FactScore with GPT-4 (including claim generation and evaluation) for a single model generation can cost upwards of 0.5\$. When evaluating model generations at scale, the cost can quickly accumulate to a steep value (5,000\$ for 10k evaluations). Secondly, while scale precludes identifying spurious text-span correlations in pre-training data apriori, we still want to ground hallucinations to the data that is the source of these correlations. In this thesis, we specifically focus on robust attribution of atomic facts in long machine-generated text using *open models* and *grounding on pre-training data*.

```

Solve the following arithmetic word problems, writing out intermediate arithmetic calculations as
python code.
Input: Viola had 167 Bread. Nancy took 137 from him. How much bread does Viola have?
Q1: [generate code] write down the arithmetic or algebra equations as python code
#1:
viola_bought = 167
nancy_took = 137
ans = viola_bought - nancy_took
Q2: [code execute] Execute the python code and get the value of "ans"
#2: 30
Q3: [EOQ]
Ans: 30
  
```

Figure 1.1: Solving a math problem by interacting with external systems like OpenAI Codex (Chen et al., 2021) and Python code interpreter.

External system interaction Recent work has shown that finetuning LLMs on a broad range of public NLP datasets (with prefixed instructions) is an effective technique for in-context learning and cross-task generalization (Mishra et al., 2021; Sanh et al., 2021; Khashabi et al., 2020; Wei et al., 2021) in both the zero-shot and few-shot settings. Emergent in-context learning abilities allow for *chain-of-thought* style prompting, that encourages LLMs to generate intermediate reasoning steps before the final answer, with multiple task-specific variants (e.g. Least-to-most prompting (Zhou et al., 2022), Self-Ask (Press et al., 2022), Ask-me-anything (Arora et al., 2022), Successive prompting (Dua et al., 2022), decomposed prompting (Khot et al., 2022)). Apart from step-by-step reasoning, there is increased interest in LLMs’ emergent abilities to interact with external tools such as search engines, web browsers, calculators, translation systems, and python interpreters (Komeili et al., 2022; Thoppilan et al., 2022; Lazaridou et al., 2022; Shuster et al., 2022; Nakano et al., 2021; Thoppilan et al., 2022; Cobbe et al., 2021; Thoppilan et al., 2022; Gao et al., 2022; Chen et al., 2022c). Figure 1.1 shows an example of the LLM interacting with external systems like code generation and execution to solve a complex math problem using Python code. Most of these approaches either require large amounts of human supervision (Thoppilan et al., 2022; Komeili et al., 2022) or carefully constructed prompts tailored to specific tasks and particular tools. An alternative line of recent work uses self-supervision to teach LLMs to use search, translation, and a calculator Schick et al. (2023)—Toolformer. In contrast, we focus on a setting that does not require any additional training or task-specific prompts. This allows users flexibility both in terms of replacing the underlying LLM (e.g. when a new version of GPT4 is released) and in replacing or adding "new" tools (either general-purpose tools or tools that are important for a specific task of interest).

1.3 Common Challenges

As highlighted above, while there has been substantial progress toward making LLM applications more reliable (explainable, robust, and capable of interacting reliably with external systems), there are some common challenges to existing modeling approaches, learning objectives, and evaluation frameworks, that we seek to address in this thesis.

Scale Scaling of data, model parameters, as well as the number of downstream applications, poses significant challenges to different aspects of reliability:

- **Explainability:** Model-generated explanations have generally been limited for tasks involving short NLP documents. However, LLMs capable of processing longer documents, and capable of multi-step and common-sense reasoning, necessitate exploring explainability for a new set of models and tasks. Similarly, the mechanistic interpretability of billions of parameters (eg. attention analysis techniques like Luong et al. (2015)) can be challenging, requiring the development of a new set of techniques for parameter-scaled models. For models pre-trained on trillions of tokens, tracing predictions back to attributable training data is also a challenge (Akyürek et al., 2022), requiring new frameworks for explainability for models pre-trained over internet-scale data.
- **Distributional robustness:** Robustness is generally explored for tasks with known biases, where assigning training examples to groups and evaluating worst-group performance is relatively easy. Scaling existing techniques for distributional robustness to *new* tasks and datasets is an outstanding challenge. Models trained on internet-scale data (trillions of tokens) also pose a challenge to the exhaustive identification of dataset bias and learned spurious correlations.
- **External system interaction:** Augmented language models that interact with external systems (eg. tools like search engines and code interpreters) to solve complex multi-step reasoning tasks require exploration of reliability in this new interactive paradigm, with an added dimension of scale—the number of external systems that the LLM has to interact with.

Lack of human supervision

- **Explainability:** Soliciting explanation annotations for every new task, to explicitly train models to generate explanations for target tasks, can be very expensive and time-consuming. Soliciting explanations for open-ended common sense reasoning tasks and long-text processing tasks is especially challenging since getting accurate and cohesive human annotations is hard.
- **Distributional robustness:** Identifying biases/spurious correlations in new tasks and datasets requires expert human supervision, which becomes increasingly hard to acquire for large datasets and complex reasoning tasks.
- **External system interaction:** The appropriate order of steps to solve a complex task, choice of tools, and choice of inputs to tools requires expert human annotation, and is challenging to acquire at scale.

User-centric design principles The eventual goal of building reliability is to engender user trust, so it is crucial to take human cognitive biases and preferences into account in designing explanation formats,

robustness techniques, and interaction protocols for LLMs. Conciseness and contrastive nature of explanations are often more useful to humans (Miller, 2019, 2018), biases are easier identified in a group of examples than in isolation (Wu et al., 2019), and a granular assessment of factuality, as opposed to a score, has also shown to be more useful for users to audit machine-generated text (Min et al., 2023). These cognitive biases and preferences should be taken into account while designing learning objectives, modeling approaches, and evaluation protocols.

Interactive debugging Another important aspect of reliability that has been underexplored in prior work is the role of human-model interaction. Specifically, users are more likely to trust models that they can troubleshoot and debug, as they take a proactive role in making the model reliable (Ribeiro et al., 2020; Ribeiro and Lundberg, 2022). User-centric design (as discussed previously) also includes the ability of users to intervene on the model prediction (by perturbing model inputs and outputs, for example) and diagnose undesirable model behavior. The specific nature of the interaction is a function of the model’s interaction capabilities, the specific model behavior or characteristic that is under investigation, and the expertise of the user. In this thesis, we specifically explore human-in-the-loop interactive debugging for the different aspects of reliability we study—interaction with model-generated explanations, interaction with examples containing spurious correlations for model diagnostics, and debugging of LLM-external system interactions.

In the following section, we briefly discuss how the different works described in this thesis address one or more of these challenges.

1.4 Approaches

Explainability: We develop a novel objective to extract concise explanations for text classification tasks (Paranjape et al., 2020), that is *unsupervised* and driven by insights from information theory (information bottleneck (Tishby et al., 1999)). This objective allows for fine-grained control over the conciseness of the explanation and allows stakeholders to obtain explanations of *desired length*. For data and parameter scaled models (like GPT3) that can read between the lines to solve commonsense tasks (Table 1.1), we develop a novel prompt-based technique that is centered around *what humans expect* from explanations (contrast), with *minimal supervision* to guide the prompting format (Paranjape et al., 2021b). The contrastive prompt

format is also interactive by design, allowing for analysis of the model-generated explanation via a textual perturbation.

Robustness: Distributional group robustness without explicit human supervision for the potential group memberships (i.e. the spurious correlation the model has learned while pretraining over trillion tokens and billions of parameters) is a challenge. In Paranjape et al. (2022), we develop an adversarial objective to simultaneously learn group memberships and train a robust classifier model for a range of text and image classification tasks. Discovered groups can be examined by humans more easily compared to explicitly looking for spurious correlations in thousands of examples in a bottom-up manner. Similarly, identifying the source of hallucinations and factual errors in generative models like ChatGPT is a challenge due to scale and lack of supervision. We propose a granular detailed attribution report of long model-generated text.² This facilitates humans to scrutinize model generations at fine granularity against background information, and arrive at meaningful conclusions about the usefulness of the generation for their specific application.

Interaction with external systems: LLMs are now capable of interacting with external systems like search engines, knowledge bases, APIs, and code interpreters/compilers, which further expands their capabilities. Given a complex user prompt/query, the LLM essentially behaves like an agent that can plan the accurate sequence of actions (interactions with external tools/agents/models) and the exact content of the interaction. This is not dissimilar to planning in traditional reinforcement settings. Learning to plan can be challenging since the task inherently lacks quality human supervision at scale. To make progress on this challenge, we develop a framework to automate planning in constrained scenarios via a combination of prompt design and few-shot learning. We also show how human feedback can be incorporated via interactive debugging of the plan to improve task performance.

1.5 Thesis Outline

Section 1 (Chapters 2 and 3) comprises our work on explainability. Chapter 2 focuses on extracting explanations from text (Paranjape et al., 2020) and Chapter 3 on generating textual explanations for model predictions on commonsense reasoning tasks (Paranjape et al., 2021b). Next, in Section 2 (Chapters 4 and 5), we discuss our work on robustness for large language models. Chapter 4 proposes an optimization technique for distributional robustness to identify groups of examples that potentially contain SCs (Paranjape et al.,

² <https://drive.google.com/file/d/1FcGWUf7vOKxPF7EE5zvmgdMLPEGjmsyV/view?usp=sharing>

2022) and Chapter 5 details our work on attributing long text generated by LLMs to a web-scale text corpus. In Section 3 (Chapter 7), we discuss LLM interaction with external systems and tools, exploring the native planning abilities of LLMs (Paranjape et al., 2023). Finally, in Chapter 7, we discuss important research directions for reliability in models that enable multimodal and embodied interactions, as well as personalized models tailored to individuals and organizations.

Chapter 2

Concise extractive explanations with information bottleneck

2.1 Overview

Portions of this chapter appeared in the proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Citation:(Paranjape et al., 2020).

In this chapter, we discuss our work on extracting concise *rationales* for text classification tasks that can also accurately predict the class label. There is almost always a trade-off between interpretable models that learn to extract *sparse* rationales and more *accurate* models that are able to use the full context but provide little explanation for their predictions (Lei et al., 2016; Weld and Bansal, 2019). Recent work represents rationales as binary masks over the input text (Lei et al., 2016; Bastings et al., 2019). During learning, it is common to encourage sparsity by minimizing a norm on the rationale masks (e.g. L_0 or L_1) (Lei et al., 2016; Bastings et al., 2019). It is often challenging to control the sparsity-accuracy trade-off in norm-minimization methods;

We formalize the problem of inducing controlled sparsity in the mask using the Information Bottleneck (IB) principle. Our approach seeks to extract a rationale as an optimal *compressed* intermediate representation (the bottleneck) that is both (1) minimally informative about the original input, and (2) maximally informative about the output class. We derive a novel variational bound on the IB objective for our case where we constrain the intermediate representation to be a concise subsequence of the input ensuring its interpretability.

Our objective affords us tunable fine-grained control over sparsity and allows us to bias the proportion of the input to be used as rationale.

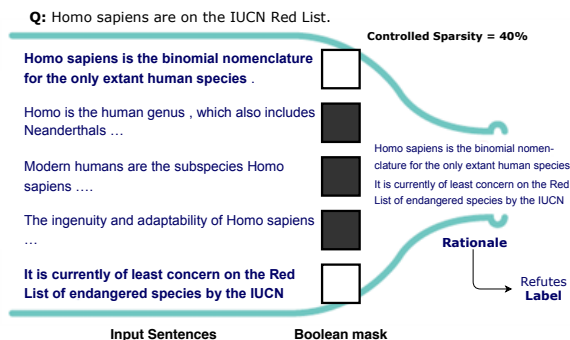


Figure 2.1: Our Information Bottleneck-based approach extracts concise rationales that are minimally informative about input, and maximally informative about the label

setting, adding a small proportion of gold rationale annotations (approximately 25% of the training examples) bridges this gap — we are able to build an interpretable model without compromising performance.

2.2 Methodology

We assume supervised text classification or regression data that contains tuples of the form (x, y) . The input document x can be decomposed into a sequence of sentences $x = (x_1, x_2, \dots, x_n)$ and y is the category, answer choice, or target value to predict. Our goal is to learn a model that not only predicts y , but also extracts a rationale or explanation z —a latent *subsequence* of sentences in x with the following properties:

1. Model prediction y should rely entirely on z and not on its complement $x \setminus z$ — *faithfulness* DeYoung et al. (2019).
2. z must be concise, i.e., it should contain as few sentences as possible without sacrificing the ability to correctly predict y .

Our interpretable model learns a boolean mask $m = (m_1, m_2, \dots, m_n)$ over the sentences in x , where $m_j \in \{0, 1\}$ is a discrete binary variable. To enforce (1), the masked input $z = m \odot x = (m_1 \cdot x_1, m_2 \cdot x_2, \dots, m_n \cdot x_n)$ is used to predict y . Conciseness is attained using an information bottleneck.

Across five tasks from the ERASER interpretability benchmark (DeYoung et al., 2019), our IB-based sparse prior objective has significant gains over previous norm-minimization techniques — up to 5% relative improvement in task performance metrics and 6% to 80% relative improvement in agreement with human rationale annotations. Our interpretable model achieves task performance within 10% of a model of comparable size that uses the entire input. We also find that in the semi-supervised

Formalizing Interpretability Using Information Bottleneck The Information Bottleneck (IB) method is used to learn an optimal compression model that transmits information from a random variable X to another random variable Y through a compressed representation Z . The IB objective is to minimize the following:

$$L_{IB} = I(X, Z) - \beta I(Z, Y), \quad (2.1)$$

where $I(\cdot, \cdot)$ is mutual information. This objective encourages Z to only retain as much information about X as is needed to predict Y . The hyperparameter β controls the trade-off between retaining information about either X or Y in Z . Alemi et al. (2016) derive the following variational bound on Equation 2.1:

$$L_{VIB} = \underbrace{\mathbb{E}_{z \sim p_\theta(z|x)} [-\log q_\phi(y|z)]}_{\text{Task Loss}} + \underbrace{\beta \text{KL}[p_\theta(z|x), r(z)]}_{\text{Information Loss}}, \quad (2.2)$$

where $q_\phi(y|z)$ is a parametric approximation to the true likelihood $p(y|z)$; $r(z)$, the prior probability of z , approximates the marginal $p(z)$; and $p_\theta(z|x)$ is the parametric posterior distribution over z .

The information loss term in Equation 2.2 reduces $I(X, Z)$ by decreasing the KL divergence¹ between the posterior distribution $p_\theta(z|x)$ that depends on x and a prior distribution $r(z)$ that is independent of x . The task loss encourages predicting the correct label y from z to increase $I(Z, Y)$.

Our Variational Bound for Interpretability The learned bottleneck representation z , found via Equation 2.2, is not human-interpretable as z is typically a compressed continuous vector representation of input x .³ To ensure interpretability of z , we define the interpretable latent representation as $z := m \odot x$, where m is a boolean mask on the input sentences in x . We assume that the mask variables m_j over individual sentences are conditionally independent given the input x , i.e. the posterior $p_\theta(m|x) = \prod_j p_\theta(m_j|x)$, where $p_\theta(m_j|x) = \text{Bernoulli}(\theta_j(x))$ and j indexes sentences in the input text.² Because $z := m \odot x$, the posterior distribution over z is a mixture of dirac-delta distributions:

$$p_\theta(z_j|x) = (1 - \theta_j(x))\delta(z_j) + \theta_j(x)\delta(z_j - x_j),$$

¹ To analytically compute the KL-divergence term, the posterior and prior distributions over z are typically K -dimensional multivariate normal distributions. Compression is achieved by setting $K \ll D$, the input dimension of X .

² We use Bernoulli distribution for m_j in this work, but any binary distribution for which KL divergence can be analytically computed can be used.

where $\delta(x - c)$ is the dirac-delta probability distribution that is zero everywhere except at c .

For the prior, we assume a fixed Bernoulli distribution over mask variables. For instance, $r(m_j) = \text{Bernoulli}(\pi)$ for some constant $\pi \in (0, 1)$. This also induces a fixed distribution on z via the definition $z := m \odot x$. Instead of using an expressive $r(z)$ to approximate $p(z)$, we use a non-parametric prior $r(z)$ to force the marginal $p(z)$ of the learned distribution over z to approximately equal π . Our characterization of the prior and the posterior achieves compression of the input via *sparsity* in the latent representation, in contrast to compression via dimensionality reduction Alemi et al. (2016).

For the intermediate representation $z := m \odot x$, we can decompose $\text{KL}(p_\theta(z_j|x), r(z_j))$ as:

$$\text{KL}(p_\theta(m_j|x), r(m_j)) + \pi H(x)$$

Since the entropy of the input, $\pi H(x)$, is a constant with respect to θ , it can be dropped. Hence, we obtain the following variational bound on IB with interpretability constraints over z :

$$L_{IVIB} = \mathbf{E}_{m \sim p_\theta(m|x)}[-\log q_\phi(y|m \odot x)] + \beta \sum_j \text{KL}[p_\theta(m_j|x) || r(m_j)] \quad (2.3)$$

The first term is the expected cross-entropy term for the task which can be computed by drawing samples $m \sim p_\theta(m|x)$. The second information-loss term encourages the mask m to be independent of x by reducing the KL divergence of its posterior $p_\theta(m|x)$ from a prior $r(m)$ that is independent of x . However, this does not necessarily remove information about x in $z = x \odot m$. For instance, a mask consisting of all ones is independent of x , but in this case $z = x$ and the rationale is no longer concise. In the following section, we present a simple way to avoid this degenerate case in practice by appropriately fixing the value of π .

The Sparse Prior Objective The key to ensuring that $z = m \odot x$ is strictly a subsequence of x lies in the fact that $r(m_j) = \pi$ is our prior belief about the probability of a sentence being important for prediction. For instance, if humans annotate 10% of the input text as a rationale, we can fix our prior belief that a sentence should be a part of the mask as $r(m_j) = \pi = 0.1 \forall j$. IB allows us to *control* the amount of sparsity in the mask that is eventually sampled from the learned distribution $p_\theta(m|x)$ in several ways. It can be explicitly tuned for the desired trade-off between end task performance and rationale length. In this work, we assume $\pi \in (0, 0.5)$ so that the sampled mask is sparse. We refer to this training objective with tunable $r(m) = \pi$ as the sparse prior (Sparse IB) method in our experiments.

2.3 Model

To optimize for objective 2.3, the posterior distribution estimator $p_\theta()$ and label likelihood estimator $q_\phi()$ are instantiated as the *explainer* and *end-task predictor* neural models respectively. Two different pre-trained transformers Devlin et al. (2019) are used to initialize both models.

Explainer $p_\theta(z|x)$: Given an input $x = x_1, x_2, \dots, x_n$ consisting of n sentences, the explainer produces a binary mask $m \in \{0, 1\}^n$ over the input sentences which is used to derive a rationale $z = m \odot x$. It maps every sentence x_j to its probability, $p_\theta(m_j|x)$ of being selected as part of z where $p(\cdot)$ is a binary distribution. The explainer contextualizes the input sequence x at the token level, and produces sentence representations $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ where \mathbf{x}_j is obtained by concatenating the contextualized representations of the first and last tokens in sentence x_j . A linear layer is used to transform these representations into logits (log probabilities) of a Bernoulli distribution. The mask $m \in \{0, 1\}^n$ is constructed by independently sampling each m_j from $p(m_j|x)$.

End-task Predictor $q_\phi(y|z)$: We define z as the rationale representation $z = m \odot \mathbf{x}$, an element-wise dot product between m_j and the corresponding sentence representation \mathbf{x}_j . The end-task predictor uses z to predict the output variable y . The same hard attention mask m is applied to all end-task transformer layers at every head to ensure prediction relies only on $m \odot x$. The predictor further consists of a log-linear classifier layer over the [CLS] token, similar to Devlin et al. (2019). When an optional query sequence is available for datasets like BoolQ, we do not mask it as it is assumed to be essential to predict y .

Training and Inference The sampling operation of the discrete binary variable $m_j \in \{0, 1\}$ in Section 2.2 is not differentiable. Lei et al. (2016) use a simple Bernoulli distribution with REINFORCE Williams (1992) to overcome non-differentiability. We found REINFORCE to be quite unstable with high variance in results. Instead, we employ reparameterization Kingma et al. (2015) to facilitate end-to-end differentiability of our approach. We use the Gumbel-Softmax reparameterization Jang et al. (2017) for categorical (here, binary) distributions to reparameterize the Bernoulli variables m_j . The reparameterized binary variable m_j^* is generated as follows:

$$m_j^* = \sigma \left(\frac{\log p(m_j|x) + g_j}{\tau} \right),$$

where σ is the Sigmoid function, τ is a hyperparameter for the temperature of the Gumbel-Softmax function, and g_j is a random sample from the Gumbel(0,1) distribution Gumbel (1948). $m_j^* \in (0, 1)$ is a continuous and differentiable approximation to m_j with low variance. During inference, we extract the top $\pi\%$ sentences with the largest $p_\theta(m_j|x)$ values, where π corresponds to the threshold hyperparameter described in Section 2.2.

Semi-Supervised Setting As we will show in Section 2.5, despite better control over the sparsity-accuracy trade-off, there is still a gap in task performance between our unsupervised approach and a model that uses full context. To bridge this gap and better manage the trade-off at minimal annotation cost, we experiment with a semi-supervised setting where we have annotated rationales for part of the training data. For input example $x = (x_1, x_2, \dots, x_n)$ and a gold mask $\hat{m} = (\hat{m}_1, \hat{m}_2, \dots, \hat{m}_n)$ over sentences, we use the following semi-supervised objective:

$$L_{semi} = \mathbf{E}_{m \sim p_\theta(m|x)} [-\log q(y|m \odot x)] + \gamma \sum_j -\hat{m}_j \log p(m_j|x) \quad (2.4)$$

While we still sample from $p(m|x)$ and train end-to-end using reparameterization, the information loss over $p(m|x)$ is replaced with the supervised rationale loss.

2.4 Experimental Setup

End Tasks We evaluate our Sparse IB approach on five text classification tasks from the ERASER benchmark DeYoung et al. (2019). All these datasets have *sentence-level* rationale annotations for validation and test sets. The ERASER tasks contain rationale annotations for the training set, which we only use for our semi-supervised experiments. We closely follow dataset processing in the ERASER benchmark setup.

Evaluation and Implementation details To evaluate quality of rationales, we report the token-level Intersection-Over-Union F1 (IOU F1), which is a relaxed measure for comparing two sets of text spans. We also report token-level F1 scores. Details about models and hyperparameters used can be found in (Paranjape et al., 2020).

Baselines We first consider two bounding scenarios where no rationales are predicted. In the **Full Context (Full)** setting, the entire context is used to make predictions; this allows us to estimate the loss in performance as a result of interpretable hard attention models that only use $\pi\%$ of the input. In the **Gold Rationale (Gold)** setting, we train a model to only use human rationale annotations during *training and inference* to estimate

an upper-bound on task and rationale performance metrics. We compare our Sparse IB approach with the following baselines. For fair comparison, all baselines are modified to use BERT-based representations.

- **Norm Minimization** (Sparse Norm) Existing approaches Lei et al. (2016); Bastings et al. (2019) learn sparse masks over the inputs by minimizing the L_0 norm of the mask m .
- **Controlled Norm Minimization** (Sparse Norm-C) For fair comparison against our approach for controlled sparsity, we modify Sparse Norm to ensure that the norm of m is not penalized when it drops below the threshold π .

$$L_{SLO-C} = \mathbf{E}_{m \sim p(m|x)}[-\log q(y|z)] + \lambda \max(0, \|m\| - \pi) \quad (2.5)$$

- **No Sparsity** No sparsity-inducing loss term, to evaluate the effect of sparsity-inducing objectives.
- **Supervised Approach (Pipeline)** Lehman et al. (2019) learn an explainer and a task predictor independently in sequence using supervision for rationales and task labels, using the output of the explainer in the predictor during inference.

2.5 Results

| Approach | FEVER | | | MultiRC | | | Movies | | | BoolQ | | |
|------------------------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| | Task | F1 | IOU | Task | F1 | IOU | Task | F1 | IOU | Task | F1 | IOU |
| 1. Full | 89.5 | 33.7 | 36.2 | 66.8 | 29.1 | 29.2 | 91.0 | 35.1 | 47.3 | 65.6 | 11.8 | 15.0 |
| 2. Gold | 91.8 | - | - | 76.6 | - | - | 97.0 | - | - | 85.9 | - | - |
| Unsupervised | | | | | | | | | | | | |
| 3. No Sparsity | 82.8 | 35.7 | 38.1 | 60.1 | 20.8 | 19.8 | 78.2 | 24.6 | 37.9 | 62.5 | 8.1 | 10.7 |
| 4. Sparse Norm | 83.1 | 40.9 | 44.0 | 59.7 | 19.9 | 20.4 | 78.6 | 23.5 | 34.7 | 62.5 | 8.5 | 12.8 |
| 5. Sparse Norm-C | 83.3 | 41.6 | 44.9 | 61.7 | 21.7 | 21.8 | 81.8 | 22.8 | 34.4 | 63.7 | 10.7 | 14.3 |
| 6. Sparse IB (Us) | 84.7 | 42.7 | 45.5 | 62.1 | 24.9 | 24.3 | 84.0 | 27.5 | 39.6 | 65.2 | 12.8 | 16.5 |
| Supervised/Semi-supervised | | | | | | | | | | | | |
| 7. Bert-To-Bert [†] | 87.7 | 81.2 | 83.5 | 62.4 | 39.9 | 40.9 | 82.4 | 14.5 | 7.5 | 54.4 | 13.4 | 5.2 |
| 8. Bert-To-Bert ^ε | 85.0 | 78.1 | 81.7 | 63.3 | 41.2 | 41.6 | 86.0 | 16.2 | 15.7 | 62.3 | 18.4 | 31.5 |
| 9. 25% data (Us) | 88.8 | 63.9 | 66.6 | 66.4 | 54.0 | 54.4 | 85.4 | 28.2 | 43.4 | 63.4 | 19.2 | 32.3 |

Table 2.1: Task, Rationale IOU F1 (threshold set to 0.1) and Token F1 for our hard-attention Sparse IB approach and baselines on test sets, averaged over 5 random seeds. Gold IOU and token F1 are 100.0. We use 25% training data in our semi-supervised setting (Section 2.3). [†] Reported Results ^ε We could not reproduce numbers for the Bert-to-Bert supervised method reported in DeYoung et al. (2019).

Quantitative Evaluation Table 2.1 compares our Sparse IB approach against baselines (Section 2.4). Sparse IB outperforms norm-minimization approaches (rows 4-6) in both agreement with human rationales and task performance across all tasks. We perform particularly well on rationale extraction with 5 to 80%

relative improvements over the better-performing norm-minimization variant Sparse Norm-C. Sparse IB also attains task performance within 0.5 to 10% of the full-context model (row 1), despite using $< 50\%$ of the input sentences. All unsupervised approaches still obtain a lower IOU F1 compared to the full context model for Movies and MultiRC, primarily due to their considerably lower precision on these benchmarks.

Our results also highlight the importance of explicit *controlled sparsity* inducing terms as effective inductive biases for improved task performance and rationale agreement. Specifically, sparsity-inducing methods consistently outperform the No Sparsity-baseline (row 3). One way to interpret this result is that sparsity objectives add input-dimension regularization during training, which results in better generalization during inference. Moreover, Sparse Norm-C, which adds the element of *control* to norm-minimization, performs considerably better than Sparse Norm. Finally, we see a positive correlation between task performance and agreement with human rationales. This is important since accurate models that also better emulate human rationalization likely engender more trust.

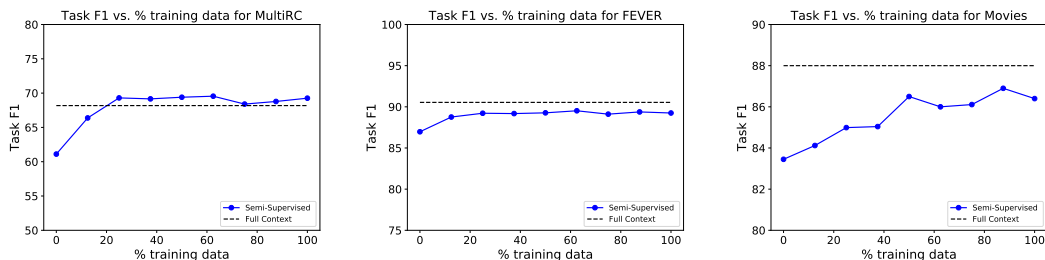


Figure 2.2: Semi-supervised experiments showing the task performance for varying proportions of rationale annotation supervision on the MultiRC, FEVER, and Movies datasets.

Semi-supervised Setting In order to close the performance gap with the full-context model, we also experiment with a setup where we minimize the task and the rationale prediction loss using rationale annotations available for a part of the training data (Section 2.3). Figure 2.2 (left, center) shows the effect of incorporating an increasing proportion of rationale annotation supervision for the FEVER and MultiRC datasets. Our semi-supervised model is even able to match the performance of the full-context models for both FEVER and MultiRC with only 25% of rationale annotation supervision. Furthermore, Figure 2.2 also shows that these gains can be achieved with relatively modest annotation costs since adding more rationale supervision to the training data seems to have diminishing returns.

Table 2.1 compares our interpretable model (row 9), which uses rationale supervision for 25% of the training data, with the full-context model and the Pipeline approach (row 8). On three (FEVER, MultiRC,

and BoolQ) out of five datasets for which rationale supervision is available, our interpretable models match the task performance of the full-context models while recording large gains in IOU (17-30 F1 absolute). Our approach outperforms the pipeline-based approach in task performance (for FEVER, MultiRC, Movies, and BoolQ) and IOU (for MultiRC and Movies). These gains may result from better exploration due to sampling and inference based on a fixed budget of $\pi\%$ sentences.

2.6 Discussion

Our approach outperforms existing norm-minimization techniques in task performance and agreement with human rationales for tasks. We are also able to close the gap with models that use the full input with $< 25\%$ rationale annotations for a majority of the tasks. As our results indicate, using our fully unsupervised information bottleneck-based objective allows for fine-grained control of the conciseness-accuracy tradeoff and enables stakeholders to obtain explanations of the desired length. However, the extractive paradigm can be unfavorable for certain tasks like commonsense question answering, where the given input provides limited context for the task (Bowman et al., 2015; Zhang et al., 2020; Rajani et al., 2019). In Chapter 3, we explore free-form open-ended textual format to generate explanations for commonsense reasoning tasks.

Chapter 3

Contrastive explanations for large language models

3.1 Overview

Portions of this chapter appeared in the proceedings of the 2021 Conference of the Association for Computational Linguistics (ACL). Citation: (Paranjape et al., 2021b).

Pretrained Language Models (PLMs) (Raffel et al., 2020; Lewis et al., 2020; Radford et al., 2019; Brown et al., 2020) have been shown to encode substantial amounts of knowledge in their parameters (Petroni et al., 2019; Talmor et al., 2020; Roberts et al., 2020) and have achieved impressive performance on commonsense reasoning (CSR) tasks without the use of external knowledge (Trinh and Le, 2018; Yang et al., 2020). However, these models provide little human-interpretable evidence of the intermediate commonsense knowledge or reasoning they used, and have been observed to overly rely on superficial dataset artifacts (Poliak et al. (2018); Geva et al. (2019)). In Chapter 2, we introduced a novel sparsity-inducing objective derived from the Information Bottleneck principle to extract rationales of desired conciseness. However, the extractive paradigm can be unfavorable for such commonsense reasoning (CSR) tasks. To overcome this limitation, recent work has shown that PLMs can explain themselves by *generating* free-form natural language explanations of their reasoning patterns (Rajani et al., 2019; Camburu et al., 2018; Narang et al., 2020). However, the space of possible free-form explanations is incredibly large, inherently ambiguous, and difficult to annotate or evaluate (Wiegrefe et al., 2020; Camburu et al., 2020; Laticinnik and Berant, 2020).

We address these challenges by proposing an unsupervised method that uses prompts that require the model to explicitly contrast different possible answers in its explanation.

Our approach is based on a key observation: Many commonsense reasoning tasks require the comparison or contrast of plausible alternatives along a distinguishing attribute. People commonly use contrastive explanations to explain their reasoning (Miller, 2018). Rather than asking “Why P ?”, they ask “Why P rather than Q ?”, where Q may be implicit from the context. Miller (2018) also argues that such a contrastive explanation also limits the space of possible explanations. Instead of exhaustively enumerating all possible reasons for an answer, we only need to consider the more limited set of reasons that might make one answer more likely than the other. Our goal is to enable PLM explanation models to similarly benefit from such constraints. We develop a small set of contrastive generation prompts that can be in-filled by a PLM such as T5 (Raffel et al., 2020) or BART (Lewis et al., 2020). These templates are designed to cover a multitude of language patterns used by humans to compare and contrast entities. Another PLM then conditions on both the original input and the generated contrastive explanation, to predict the final answer.

Our approach shows quantitative improvements in task performance over two existing methods (Shwartz et al., 2020; Laticinnik and Berant, 2020) for two commonsense reasoning tasks – Winograd Schema Challenge Levesque et al. (2012) and multiple-choice question answering about physical commonsense Sap et al. (2019). Our gains in the zero-shot setting are especially notable, outperforming the best-reported results on publicly available PLMs and improving over Shwartz et al. (2020) by up to 11%. We also show, through human evaluations, that contrastive explanations are deemed more useful for solving the original task.

3.2 Methodology

Contrastive explanation A contrastive explanation is generally defined as an answer to a counterfactual question of the form “Why P rather than Q ?” for two potential hypotheses P and Q that can follow from some event E . It explains why some *fact* P occurred instead of some *foil* Q , where Q can be implicit (Hesslow, 1988; Lipton, 1990; Miller, 2019). A good contrastive explanation points to differences between the fact and foil with regard to certain attributes, not just conveying that the fact has a certain attribute. Table 3.1 shows examples of contrastive explanations that differentiate between peanuts and raisins (on the basis of taste) or forests and fields (on the basis of vegetation densities) to explain the more probable answers to Winograd Schema instances.

Previous studies (Miller, 2019) in philosophy, psychology, and cognitive science show that humans use such contrastive explanations when explaining their decisions to each other. Importantly, Miller (2018) also argues that contrastive explanations are computationally efficient – exhaustively describing all causes for the occurrence of an event P is harder than only enlisting causes for why another event Q did not occur instead of P .

| Dataset Instance | Human-Authored Contrastive Explanation |
|---|---|
| Winograd Schema 1. The party was more interesting and uplifting than the funeral because the __ was rigid. 2. The geese prefer to nest in the fields rather than the forests because in the _ predators are more hidden. | <ul style="list-style-type: none"> ○ Parties are for celebrating while funerals are for mourning ○ People wear colorful clothes at parties and black at funerals ○ Forests are dense while fields are sparse ○ Forests have more predators than fields. |
| PIQA 1. How do you get strong hamstrings? __ (a) work out your upper body (b) work out your legs 2. How do you flood a room? __ (a) fill it with objects (b) fill it with water | <ul style="list-style-type: none"> ○ Hamstrings are located in the legs while biceps are located in the upper body ○ Hamstrings are closer to the legs than the upper body ○ Filling it with objects can clutter a room while filling it with water floods the room. |

Table 3.1: Examples of commonsense tasks that can be explained using contrastive language and some contrastive explanations authored by in-house annotators. The **Fact** and **Foil** are marked in the input.

Common sense reasoning tasks Many recently proposed commonsense reasoning tasks are framed in a multiple-choice format that facilitates contrastive explanation (see Table 3.1). The Winograd Schema Challenge (Levesque et al., 2012, WSC) is a pronoun coreference resolution task designed as a hard benchmark for evaluating everyday knowledge and commonsense reasoning (Zhang et al., 2020). For instance, in the sentence “The city councilmen refused the demonstrators a permit because they feared violence,” the pronoun *they* must be disambiguated between fact (*the city councilmen*) and foil (*the demonstrators*). Both fact and foil are explicit in such sentences. The Physical Interaction Question Answering (Bisk et al., 2020, PIQA) challenge is designed to test knowledge of physical commonsense. PIQA requires choosing between which one of two *solutions*, is a better way of achieving a *goal* posed as a question (see Table 3.1). PIQA questions relate to physical properties of entities, their affordances, and how they can be manipulated. The fact and foil are explicit in the two solutions, which typically differ from one another by a short noun phrase.

Contrastive Explanations for Commonsense Tasks To validate our intuition that contrastive reasoning is instrumental in these tasks, we performed a pilot study with 10 annotators over 100 commonsense questions

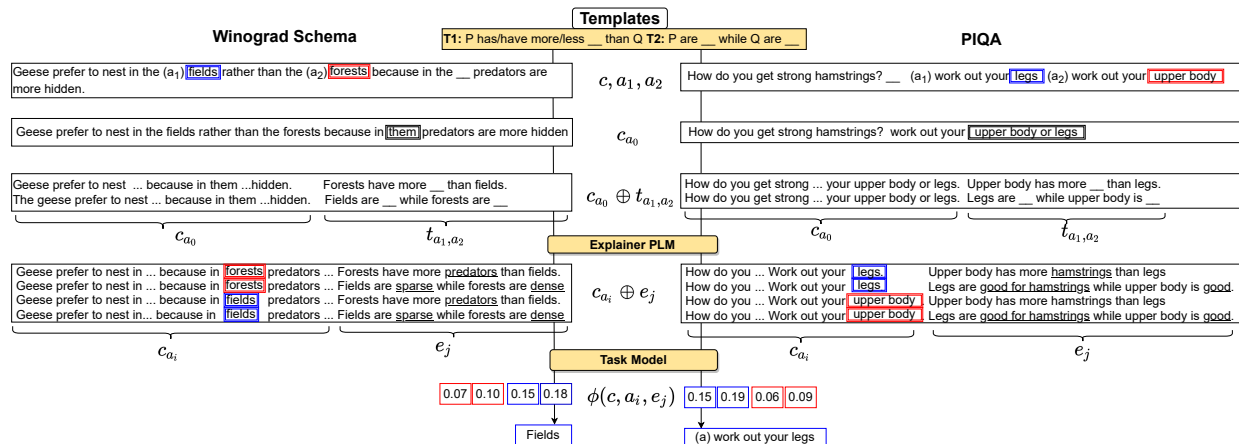


Figure 3.1: (1) A commonsense reasoning instance (c, a_1, a_2) is converted into a custom prompt $c_{a_0} \oplus t_{a_1, a_2}$ as input for the explainer PLM (2) The combination of input and explanation $(c_{a_i} \oplus e_j)$ is used by task model to score $a_i \forall i \forall j$. For a_1 and a_2 , scores are aggregated over templates.

from Winogrande and PIQA. We instructed them to answer the questions and explain their reasoning, but gave no specific instructions about what the explanations should look like. Examples are shown in Table 3.1. In 76% of Winogrande and 64% of PIQA examples, annotators explicitly contrasted the fact and foil. The frequent use of certain phrase structures, like *P are ___ while Q are ___*, strongly informed our method for generating them

3.3 Model

We assume the input to a commonsense reasoning problem consists of a textual context c which contains a placeholder $_$, and two marked answer choices a_1 and a_2 corresponding to the fact and foil (Table 3.1 (left column)). Let c_x denote substitution of x for the placeholder in c . The task is to predict whether c_{a_1} or c_{a_2} is more likely to be true, i.e., whether a_1 or a_2 best completes the context.

Our approach has two stages: First, an **Explainer PLM** generates contrastive explanations (Section 3.3.2) by infilling preset *contrastive templates* (Sec. 3.3.1) on the basis of c , a_1 , and a_2 . Then, a **Task Model** selects the correct answer conditioned on both the context and the generated explanations (Sec. 3.3.3).

3.3.1 Contrastive Templates

We develop a list of contrastive templates on the basis of an annotation study. For 250 instances from Winogrande and PIQA datasets, we asked three annotators to provide contrastive explanations by answering

this question *why is one answer more likely than the other*. We manually examined these explanations and abstracted them into templates containing at least three placeholders: two for the fact and foil being contrasted, and the rest corresponding to the properties they are being contrasted on. For instance, *peanuts are salty while raisins are sweet* becomes *Q are _ while P are _*. We retained templates that were used by annotators at least 10 times. A template may be instantiated into an explanation by replacing placeholders for the fact and foil with answers a_1 and a_2 and the remaining placeholders with the appropriate contrastive information. Examples of templates for several types of properties and example explanations using them are shown in the Appendix (Table A.1).

We evaluate the quality and coverage of our templates with another round of human evaluation. For 100 WSC and PIQA examples, we ask three annotators to either write contrastive explanations using one or more of the templates, or indicate that none of them were appropriate. Annotators used the provided prompts in over 82% of cases, indicating high coverage for the tasks we study.

3.3.2 Generating Explanations

Let t denote a contrastive template. We write t_{a_1, a_2} to denote the customization of t to an input by filling its marked placeholders for fact and foil with the answer choices. For instance, in Figure 3.1, the template *P are _ while Q are _* is customized to *Fields are _ while forests are _*¹ A full explanation may be produced by filling the remaining gaps in t_{a_1, a_2} . To do this, we leverage an infilling language model.

We first construct a neutral context c_{a_0} by filling c 's placeholder with a task-specific neutral answer that does not indicate if a_1 or a_2 is correct. For Winogrande Schema, c_{a_0} is constructed using the ambiguous pronoun in c (*them* in Figure 3.1). For PIQA, c_{a_0} is constructed as " $c \oplus a_1$ or a_2 ", where \oplus is string concatenation. (*upper body or legs* in Figure 3.1). More dataset-specific details are in Section 3.4.2. We then prepend c_{a_0} to the customized template t_{a_1, a_2} and use it as input to the infilling language model to fill in the remaining gaps in the template. We take the maximum likelihood candidate phrases from top-K decoding and use them to fill in the gaps and transform the template into a full explanation e .

We use a list of templates t_1, \dots, t_n to generate a list of candidate explanations e_1, \dots, e_n for each input, which are all fed into the task model. Task-specific heuristics used to reduce the number of prompts for each example are detailed in Paranjape et al. (2021b).

¹ In practice, we randomize the order of a_1 and a_2 when customizing the template.

3.3.3 Task Model

Given inputs c, a_1, a_2 and a list of explanations e_1, \dots, e_n , the second stage of our pipeline is a binary classifier between a_1 and a_2 which marginalizes over the explanations. We first assign a score to each answer $a \in \{a_1, a_2\}$ and explanation $e \in \{e_1, \dots, e_n\}$:

$$\phi(c, a, e) = \frac{1}{k} \log P_{\text{LM}}(c_a \oplus e),$$

Here, c_a denotes the substitution of a in c . We use ϕ as input to a binary logistic regression which marginalizes over explanations:

$$P(a | c) = \frac{\sum_i^n e^{\phi(c, a, e)}}{Z},$$

where Z is a normalizer over a_1 and a_2 . At initialization, ϕ uses a pre-trained language model, and we fine-tune it to minimize the cross-entropy loss of $P(a^* | c, a_1, a_2)$, where a^* is the correct answer. We do not fine-tune the explainer PLM since the top-K beam decoding is a discrete operation that is hard to backpropagate through. In the zero-shot setting (where the task PLM is not fine-tuned) and during inference, the answer is predicted by aggregating scores assigned to an answer by all n explanations: $\text{argmax}_i \sum_j \phi(c, a_i, e_j)$.

3.4 Experimental Setup

3.4.1 Baselines

Context-Only We experiment with a baseline that does not condition on explanations at all. Here,

$$\phi(a, c) = \frac{1}{k} \log P_{\text{LM}}(c_a),$$

and gold answer is $\text{argmax}_i \phi(a_i, c)$

Unconstrained Generation Latcinnik and Berant (2020) generate explanations from a PLM by simply beam-decoding a free-form sequence termed a *hypothesis*, that is then used by a classifier to solve the task. The model is trained end-to-end and loss terms are added to encourage the hypothesis to correspond to natural language. Explanation generation is otherwise unconstrained. For fair comparison with our approach, we do not finetune the explainer PLM.

Self-Talk Shwartz et al. (2020) is an unsupervised model that uses a PLM as the answer scorer and a (possibly different) PLM as a knowledge source, similar to our framework. They formulate the process of obtaining relevant knowledge as *self-talk* with the following steps: 1) completing natural-language clarification questions such as “what is the definition of ...” conditioned on input context, 2) generating their corresponding answers (clarifications), and 3) incorporating the clarification questions and answers as additional context to make predictions. The key difference between the two frameworks is the use of different prompting templates that seek different types of information from the PLM. While Shwartz et al. (2020) draw inspiration from inquiry-based discovery learning Bruner (1961), we target contrastive reasoning with instance-specific prompt selection.

3.4.2 Implementation details

We use BART-Large Lewis et al. (2020) and T5 Raffel et al. (2020) as the explainer PLMs. For T5 we use special symbols $\langle extra_id_0 \rangle$ and $\langle extra_id_1 \rangle$ in place of the blanks($_$) in our templates. We observe that T5 is able to replace these tokens with multi-word phrases. For BART, we substitute blanks with a sequence with four [MASK] tokens to enable the generating of multiple words. BART can choose to delete a [MASK] token during generation.

To make fair comparisons with our baselines, we use GPT2-XL Radford et al. (2019) as the task model that estimates $\phi(a, e, c)$. GPT2-XL is the best performing PLM in Shwartz et al. (2020) for WSC and PIQA tasks. Details about hyperparameters are included in Paranjape et al. (2021b).

| | Explainer PLM (# Params) | Task model | WGRD | | PIQA | | WSC | WGND |
|------------------|-----------------------------|------------|-------------|-------------|-------------|-------------|-------------|-------------|
| | | | ZS | FT | ZS | FT | ZS | ZS |
| 1. Context-only | GPT2-XL (1.5B) | GPT2-XL | 54.8 | 77.9 | 62.6 | 80.1 | 61.5 | 52.5 |
| 2. Unconstrained | GPT2-XL | | 54.9 | 77.8 | 63.9 | 80.7 | 61.4 | 52.5 |
| 3. Self-Talk | GPT2-XL | | 55.1 | 78.4 | 69.5 | 82.3 | 62.0 | 52.8 |
| 4. Contrastive | BART-Large(680M) | | 56.8 | 78.9 | 71.8 | 82.8 | 63.2 | 53.1 |
| 5. (Ours) | T5-Large (770M) | | 59.2 | 79.1 | 72.5 | 83.5 | 63.5 | 54.1 |
| 6. | T5-11B(11B) | | 61.3 | 79.6 | 74.4 | 83.9 | 64.1 | 55.1 |

Table 3.2: Test set accuracy on Winogrande (WGRD), PIQA, WSC and Winogender (WGND). ZS is Zero-shot models while FT is fine-tuned models. WSC and Winogender don’t have training data for finetuning. Across all our models, the task model is GPT2-XL for fair comparison with Shwartz et al. (2020) and to make finetuning tractable.

3.5 Experimental Results

3.5.1 Task Performance

Setup Collecting and evaluating on gold contrastive explanations can be challenging due to their inherent ambiguity and issues with evaluation metrics like BLEU. However, we can report task accuracy as a proxy for explanation quality.

Results Table 3.2 compares the task performance of our model with a number of baselines defined in Section 3.4.1. We observe that generating and conditioning on additional information from PLMs improves performance over just using the original input (Row 1 vs. 2-6). Using templates to prompt the PLM for specific knowledge is better than unconstrained generation of text (Row 2 vs. 3-6). Contrastive explanations outperform previous work that use clarifications questions in self-talk Shwartz et al. (2020). The T5-Large explainer already surpasses the results of self-talk despite being smaller than GPT2-XL, demonstrating the impact of using contrastive explanations.

We also observe that increasing parameters in the explainer PLM (going from T5-Large to T5-11B) improves performance. At the time of publication of this work, our results on the T5-11B model were the highest reported for an open-sourced model that does not use external knowledge or fine-tuning but only relies on relevant knowledge stored in the PLM.² We hypothesize that larger models that potentially memorize more facts will only improve results using our framework.

Finally, our approach smaller improvements on finetuning the task model. This suggests that some of the task reasoning is learned by the task model, which is still implicit. The task model may also be learning to ignore explanations.

3.5.2 Human Evaluation

Setup Following the human evaluation setup in self-talk Shwartz et al. (2020), we sample up to 50 useful³ explanations produced by the T5-Large model for Winogrande and PIQA by selecting the template that has the best value for $\phi(a_{gold}, e_j, c)$ for that instance. For fair comparison, we sample the same size of examples for the explainer model in Shwartz et al. (2020) for both datasets.

² The zero-shot SOTA model Brown et al. (2020) uses the 175B parameter GPT3, which would likely also provide a stronger explainer for our approach but we did not have access to verify test this hypothesis.³ examples that were incorrectly predicted by the Context-only baseline but correctly predicted by our model.

| Metric | Self-Talk(Reported) | | Self-Talk | | Contrastive | |
|----------------|---------------------|------|-----------|------|-------------|------|
| | WGRD | PIQA | WGRD | PIQA | WGRD | PIQA |
| Relevant | 68 | 60 | 70.4 | 61.7 | 73.1 | 70.7 |
| Factual | 46 | 42 | 40.8 | 38.8 | 43.0 | 39.4 |
| Helpful | 24 | 26 | 22.5 | 27.7 | 42.8 | 32.8 |
| Grammatical | 87.2 | 87.2 | 87.5 | 87.5 | 83.5 | 83.5 |
| Flipping helps | NA | NA | NA | NA | 66.9 | 59.4 |

Table 3.3: Human Evaluation Results on Winogrande(WGRD) and PIQA. Reported human evaluation results on Self-talk are different from ours, which can be because of moderate levels of agreement (Fleiss Kappa $\kappa = 0.43$). We report average and deviation across 3 different samples of 50 examples. Grammaticality is judged together for all datasets following Shwartz et al. (2020). Only contrastive explanations can be flipped.

| Explainer PLM | WGRD | | PIQA | | WSC | WGND |
|------------------|-----------|-----------|-----------|-----------|-----------|-----------|
| | ZS | FT | ZS | FT | ZS | ZS |
| BART-Large | 52.9(5.4) | 75.9(4.0) | 66.5(7.9) | 79.1(4.6) | 59.1(6.9) | 51.8(4.5) |
| T5-Large | 56.2(5.3) | 75.3(5.0) | 68.1(6.5) | 80.2(4.2) | 60.2(5.5) | 51.2(5.6) |
| T5-11B | 58.6(4.5) | 76.1(4.7) | 70.5(5.4) | 81.0(3.6) | 62.1(3.3) | 52.9(4.2) |

Table 3.4: Flipped Evaluation Results for contrastive explanation models. Reporting test accuracy across all datasets. % Drop in performance as a result of flipping is indicated in parentheses.

Crowd workers are presented with a commonsense instance, the correct answer, and the explanation of why the answer is correct and are instructed to judge for: 1) *Grammaticality*, whether the explanation is grammatical, not entirely grammatical but understandable, or completely not understandable; and if the answer was anything but “completely not understandable”, 2) *Relevance*, whether the explanation is relevant to topic of the instance, and 3) *Factually Correctness*, whether the explanation is factually correct or likely true; and 4) *Helpfulness*, whether it adds helpful information to solve the instance. These metrics and definitions follow closely from Shwartz et al. (2020). Finally, the annotators are also shown the same explanation with fact and foil flipped and are instructed to judge if the other answer is more likely if they assume the flipped explanation to be hypothetically true.

Results Table 3.3 shows the results of human evaluation of our contrastive and self-talk explanations. The contrastive explanations are overwhelmingly preferred over self-talk explanations for relevance, factual correctness and usefulness. They may be considered less grammatical because of in-filling noise (such as incomplete phrases). They also deem the flipped explanation to be useful evidence to predict the opposite label, indicating that they are useful pieces of reasoning for people.

3.6 Discussion

Our choice of contrastive language templates also facilitates a novel way to evaluate the explanation’s usefulness in prediction. If the contrast in the explanation is reversed by flipping the position of the fact and the foil in the explanation, this may cause a model that uses the explanation to flip its prediction. The flipped explanation provides a hypothetical situation where the foil is more likely than the fact. For instance, “Peanuts are salty while raisins are sweet.” when switched to “Peanuts are sweet while raisins are salty.” may provide evidence that makes *peanuts* more likely. Any reduction in task performance can help to quantify the extent to which the model relies on the contrast provided in the explanation. We can further isolate the explanation’s usefulness for Winograd Schemas by abstracting away the fact and foil i.e. replacing them with placeholders P and Q .

We observe that flipping the output of the explainer PLM does lead to a decline in performance (see Table 3.4). This drop in performance indicates that the model may be using the contrastive knowledge to reason about the task, reversing which causes the flip in decision. However, the drop is not significant. In some cases, performance doesn’t drop below the Context-only variant, suggesting that the additional contrastive knowledge is still useful for prediction. We observe that the attribute of contrast is correctly identified, but the ordering of fact and foil is misplaced. Finetuned models also show a smaller decline. In this case, the task model may have explicitly learned to ignore the explanation in lieu of its own knowledge. We also observe that larger models have smaller decline in performance (Row 2 vs Row 3 in Table 3.4), indicating that these models may be better at ignoring sequences that they are less likely to produce. These analyses provide insights into the models’ use of contrastive explanations, facilitating interactive debugging through textual perturbations. We will show such examples of interactive debugging in Parts II (Chapter 4) and Parts III (Chapter 6) of this thesis as well.

Chapter 4

Distributional robustness to spurious correlations

In this part of the thesis (Chapters 4 and 5), we turn to a different challenge for reliability—robustness to distributional shifts: Model pre-training focuses on maximizing the likelihood of predicting the subsequent token, resulting in an amplification of biases and correlations present in the data. Users desire models that exhibit robustness in the face of such correlations present in the training data distribution, which may otherwise lead to biased predictions, hallucinations, and factuality errors in generated text. We focus on the problem of distributional robustness arising due to dataset bias and learned spurious correlations in this chapter, and discuss hallucinations in model generated text in Chapter 5.

4.1 Overview

Portions of this chapter appeared in proceedings of the 2022 International Conference on Learning Representations (ICLR). Citation: (Paranjape et al., 2022).

Neural models trained using the empirical risk minimization principle (ERM) are highly accurate on average; yet they consistently fail on rare or atypical examples that are unlike the training data. Such models may end up relying on spurious correlations (between labels and task-independent features), which may reduce empirical loss on the training data but do not hold outside the training distribution (Koh et al., 2021; Hashimoto et al., 2018). Figure 4.1 shows examples of such correlations in the MultiNLI and CelebA datasets. When the correlations are known and training data can be partitioned into dominant and rare groups, group

distributionally robust optimization (G-DRO, Sagawa et al., 2019) can efficiently minimize the worst (highest) expected loss over groups and improve performance on the rare group. A key limitation of G-DRO is the need for a pre-defined partitioning of training data based on a known spurious correlation; but such correlations may be unknown, protected, or expensive to obtain.

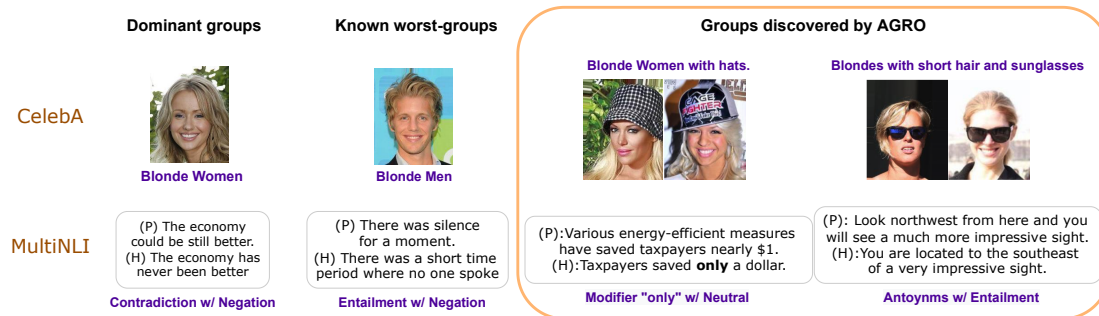


Figure 4.1: Groups discovered by AGRO on CelebA image classification dataset and MultiNLI sentence pair classification dataset.

We propose AGRO, an end-to-end approach that makes no assumptions about the nature of the task and the availability of additional supervision for partitioning of training data. We construct a new parameterized *grouper* model that produces a soft distribution over groups for every example in the training data and is jointly trained with the task model. Specifically, while G-DRO optimizes the task model to minimize the worst group-loss¹, the grouper model adversarially seeks a probabilistic group assignment such that the worst group-loss is maximized.

On four datasets in the WILDS benchmark (Koh et al., 2021) (MultiNLI, CivilComments, CelebA, and Waterbirds), AGRO simultaneously improves performance on *multiple* worst-groups² corresponding to previously characterized spurious correlations, compared to ERM and G-DRO with known group assignment. AGRO also improves worst-group performance over prior approaches that find spurious correlations and groups by 8% on average, establishing a new SOTA for such methods on two of the WILDS datasets³. We develop a novel annotation task for humans to analyze the discovered AGRO groups—distinguishing group members from random examples and perturbing them to potentially change model predictions. We find that humans can identify existing and **previously unknown** features in AGRO groups that lead to systematic model errors and are potentially spurious.

¹ Largest group-wise loss in a minibatch² A group of examples where the spurious correlation between feature and label does not hold.

³ At the time of publication of this work

4.2 Method

Problem Setup We consider the typical image/text classification problem of predicting labels $y \in \mathcal{Y}$ from input $x \in \mathcal{X}$. Training data \mathbb{D} is drawn from the joint distribution $P(\mathcal{X}, \mathcal{Y})$. In state-of-the-art classification models, inputs are typically encoded using multi-layered pretrained transformers (Vaswani et al., 2017; He et al., 2021; Bao et al., 2021) We use $g(x)$ and $h(x)$ to represent input encoding before and after fine-tuning these encoders on \mathbb{D} .

ERM principle Given a model family Θ and a loss function $l : \Theta \times \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}_+$, empirical risk minimization aims to find a model $\theta \in \Theta$ that minimizes empirical loss over data drawn i.i.d from the empirical distribution P . $\hat{\theta}_{ERM} := \operatorname{argmin}_{\theta \in \Theta} \mathbb{E}_{(x,y) \sim P}[l(x,y; \theta)]$. While ERM models achieve high accuracy on i.i.d. evaluation data on average, they often underperform when the test distribution shifts significantly from the training distribution (Madry et al., 2017). ERM also underperforms on a biased sample of the test set where a spurious correlation is absent (Koh et al., 2021).

G-DRO for Robust optimization Group-DRO (Sagawa et al., 2019) is a category of distributionally robust optimization (DRO, Duchi et al., 2016), where training distribution P is assumed to be a mixture of m groups, and each training point (x, y) comes from one group $g \in \mathcal{G}$. G-DRO minimizes the empirical worst-group risk $\hat{\mathcal{R}}(\theta)$ i.e. worst (highest) empirical loss over m groups:

$$\hat{\theta}_{DRO} := \operatorname{argmin}_{\theta \in \Theta} \{ \hat{\mathcal{R}}(\theta) := \max_{g \in \mathcal{G}} \mathbb{E}_{(x,y) \sim p(x,y|g)} [l(x,y; \theta)] \},$$

where $p(x,y|g) \forall g$ is the empirical distribution of training data over groups. With sufficient regularization over θ , G-DRO enables training models that are robust to test-time worst-group distributional shifts. In practice, prior work adopts the efficient and stable online greedy algorithm (Oren et al., 2019) to update θ . Specifically, worst-group risk $\hat{\mathcal{R}}(\theta) := \max_{q \in \mathcal{Q}} \mathbb{E}_{g \sim q, (x,y) \sim p(x,y|g)} [l(x,y; \theta)]$. The uncertainty set \mathcal{Q} are categorical group distributions that are α -covered by the empirical distribution over groups p_{train} i.e. $p_{train}(g) := |\mathbb{1}(g_i = g)| \forall i \in \mathbb{D} / |\mathbb{D}|$ and $\mathcal{Q} = \{q : q(g) \leq p_{train}(g) / \alpha \forall g\}$. Effectively, this algorithm up-weights the sample losses by $\frac{1}{\alpha}$ which belong to the α -fraction of groups that have the worst (highest) losses, where $\alpha \in (0, 1)$. G-DRO is limited to scenarios where both spurious correlations and group memberships are known, which is often not the case.

4.2.1 AGRO: Adversarial discovery of Group for Robust Optimization

AGRO combines insights from group distributionally robust optimization and error slice discovery, introducing a novel end-to-end framework to accomplish both objectives. We formalize the problem of *group discovery for robustness*, where g is an un-observed latent variable to be inferred during the training process. We replace discrete group memberships $\mathbb{1}(g_i = g); g \in \mathcal{G}$ with a soft-group assignment, i.e. a probability distribution over groups $P_{i,g} := p(g|f_i; \phi); g \in \mathcal{G}$. $P_{i,g}$ refers to the probability that the i -th example belongs to the g -th group and is realized by a neural network q with parameters ϕ . This enables modeling co-occurring spurious features and overlapping groups. The input to the *grouper model* q is a high-dimensional feature vector $f_i \in \mathcal{F}$ that can potentially encode the presence of spurious correlations in example i (described in section 4.2.2).

AGRO jointly trains the parameters of the robust task model, θ , and the grouper model, ϕ , by setting up a zero-sum game between the robust model and the grouper model. The optimization occurs over R *alternating* game rounds—the **primary round** that updates the task model θ and the **adversary round** that updates the grouper parameters ϕ . Parameters that are not optimized in a given round are frozen. In the primary round, the optimizer seeks to minimize the worst-group loss (exactly the G-DRO objective), while in the adversary round, it seeks to find a group assignment that maximizes the worst-group loss. In doing so, the adversary finds a group assignment that is maximally informative for the primary, since it produces a loss landscape over groups that is highly uneven. This forces the primary model, in the next round, to aggressively optimize the worst empirical loss to even out the landscape. With $p_{train}(g) := \sum_{i \in \mathbb{D}} P_{i,g} / |\mathbb{D}|$ and uncertainty set \mathcal{Q} is defined as before, the AGRO optimization objectives for the primary and adversary rounds respectively are:

$$\hat{\theta}, \hat{\phi}_{AGRO} := \operatorname{argmin}_{\theta \in \Theta} \left\{ \max_{q \in \mathcal{Q}(\phi)} \mathbb{E}_{g \sim q, (x,y) \sim p(x,y|q)} [l(x,y; \theta, \phi)] \right\}, \quad (4.1)$$

$$\hat{\theta}, \hat{\phi}_{AGRO} := \operatorname{argmax}_{\phi \in \Phi} \left\{ \min_{q \in \mathcal{Q}(\phi)} \mathbb{E}_{g \sim q, (x,y) \sim p(x,y|q)} [l(x,y; \theta, \phi)] \right\}, \quad (4.2)$$

Primary Round In round r , the primary classifier finds the best parameters θ that minimize the worst-group loss based on the current dynamic group assignments provided by ϕ in round $r - 1$. Updates to θ are similar to the online greedy updates used by Oren et al. (2019) i.e. up-weight the loss of α fraction of groups with the highest loss, then minimize this weighted loss.

Adversary Round In round $r + 1$, the adversarial grouper updates ϕ and learns a soft assignment of groups that *maximizes* the loss of the worst-group (highest loss group). In practice, we adopt the converse of the greedy updates made in the primary round, i.e. down-weight the loss of α fraction of groups with the highest loss, and then maximize this weighted loss.

For stable optimization, we iterate over T_1 minibatches of training data to update θ in the r^{th} round and iterate over T_2 minibatches to update ϕ in the $r + 1^{\text{th}}$ round. In the first primary round, we start with a random group assignment (i.e. random initialization for ϕ), which amounts to training an ERM model. θ is initialized with a pretrained transformer-based encoder and MLP classifier. In the first adversary round, we adopt a different initialization for ϕ , which is explained in Section 4.3. The grouper model ϕ takes as input a feature vector $f_i \in \mathcal{F}$ for each training example x_i . Next, we describe our choice of \mathcal{F} .

4.2.2 Features for Group Discovery

Most prior work uses some knowledge about the task (Wu et al., 2022; Gururangan et al., 2018), data collection process (Poliak et al., 2018) or metadata information (Koh et al., 2021) for group discovery. Instead, we develop an unsupervised end-to-end group discovery method that does not rely on any task-specific knowledge. We do so using general purpose features $f_i \in \mathcal{F}$ that can potentially indicate the presence or absence of a spurious correlations.

ERM features and errors We use features from weaker ERM models (i.e., under-trained, smaller capacity ones), which have also been used in prior work for automatic group discovery Creager et al. (2021); Sohoni et al. (2020). DOMINO (Eyuboglu et al., 2022) additionally clusters model representations *and* errors to generate coherent error slices on held-out data. However, overparameterized models often perfectly fit training data. To estimate model errors on training data, we apply K-fold cross-validation to get model predictions on held-out training instances in every fold. Specifically, for a training example x_i assigned to the k -th fold’s held-out set, we compute the following features: the model’s fine-tuned representations $h(x_i)$, the reference label y_i , and the vector of prediction probabilities over labels $\{p(\hat{y}_i|x_i; \theta_k) \forall \hat{y}_i \in \mathcal{Y}\}$, where θ_k is the fold-specific ERM classifier.

Pretrained features Recent work has shown that large transformer-based models trained on web-scale data learn general features for images which improve on out-of-distribution data (Wortsman et al., 2022a,b). Pretrained models are less likely to encode dataset-specific features in their representations. Following this observation, we also include pretrained representations $g(x_i)$ of the input x_i . In sum, the group discovery

| Dataset | Inputs | Labels | Worst Groups |
|----------------|-----------|-----------------------------|--|
| MultiNLI | Sentences | Entail, Contradict, Neutral | KWG1: Negation-Contradict, KWG2: Lexical overlap - Entail |
| Waterbirds | Image | Waterbird, Landbird | KWG1: Waterbird w/ land background, KWG2: Landbird w/ water backgrounds |
| CelebA | Image | Blonde, Non-Blonde | KWG1: Male Blondes, KWG2: Blondes wearing eyeglasses, KWG3: Blondes w/ big noses |
| Civil Comments | Comment | Toxic, Non-toxic | KWG1: Toxic comment mentioning black, KWG2: LGBTQ and KWG3: other religions |

Table 4.1: Inputs, labels and known worst-groups in WILDS datasets.

features f_i for a training example x_i are: the representations $g(x_i)$ and $h(x_i)$, the label y_i and the probabilities $\{p(\hat{y}_i|\theta_k)\forall\hat{y}_i \in \mathcal{Y}\}$.

4.3 Experimental Setup

Datasets with known spurious correlations We evaluate performance on four datasets in language and vision from the popular **WILDS benchmark** (Koh et al., 2021) to study in-the-wild distribution shifts. We select tasks that contain more than one known worst-group and report on the top three known worst-groups (referred to as KWGs) from these datasets, chosen based on the rate of correlation between feature and label. Table 4.1 briefly describes these known worst-groups by dataset.

Baselines We compare against previous automatic group discovery approaches used with G-DRO:

JTT: Liu et al. (2021) retrain on training data errors made by a weaker ERM model. Model selection is based on errors made on the validation set.

GEORGE: Sohoni et al. (2020) cluster the training and validation sets based on ERM features and loss components. The worst-group on the validation set is used for model selection.

EIIL: Creager et al. (2021) propose to train a group discovery model to adversarially maximize the invariant risk minimization objective. Upon examination of their experimental details, we find that for all large datasets (that we report on), they use labels as groups, since EIIL is hard to scale. We follow the same procedure to compare EIIL with AGRO. For small datasets Waterbirds and SST2, we use EIIL as is. The class with the worst validation performance is used for model selection.

G-DRO: Following (Sagawa et al., 2019), we optimize G-DRO based on a group assignment for a known spurious correlation, and use the same assignment on the validation set for model selection. G-DRO serves as an upper bound on the worst-group performance for this spurious correlation. Specifically, we use KWG1 to run G-DRO on all datasets.

| Group Accuracy | ERM | G-DRO | JTT | GEORGE | EIIL | AGRO |
|-----------------------------|-----------|-----------|------------------|------------------|------------------|-------------------|
| MultiNLI | | | | | | |
| Full evaluation set | 87.0(.04) | 84.8(.05) | 79.6(1.2) | 58.4(.99) | 86.4(.7) | 85.31(.5) |
| KWG1:Negation-Contr. | 75.8(.09) | 84.2(.9) | 71.9(2.0) | 54.3(0.7) | <u>76.1(1.)</u> | 82.29(.23) |
| KWG2:Lexical Overlap-Ent. | 45.5(.05) | 43.1(1.0) | 48.2(1.4) | 21.0(1.3) | 47.0(.55) | 51.8(.95) |
| Waterbirds | | | | | | |
| Full evaluation set | 97.75(1.) | 96.3(.9) | 96.4(1.3) | 97.8(.8) | 96.5(1.3) | 97.7(1.1) |
| KWG1:Waterbirds-Land | 90.2(.7) | 97.7(.8) | 83.5(2.0) | 96.2(1.0) | 78.8(.8) | 96.2(1.2) |
| KWG2:Landbirds-Water | 97.4(.8) | 93.4(1.2) | 96.1(1.5) | 95.9(1.4) | 84.6(.5) | 96.1(1.0) |
| CelebA | | | | | | |
| Full evaluation set | 95.5(.4) | 94.0(1.3) | 92.9(1.0) | 94.7(1.5) | 94.5(.91) | 95.59(1.) |
| KWG1:Males-Blonde | 39.6(.23) | 86.8(1.9) | 75.8(.8) | 57.7(3.0) | 55.5(.5) | 69.2(1.2) |
| KWG2:Eyeglasses-Blonde | 53.8(.88) | 82.7(.8) | 82.7(1.6) | 61.5(2.5) | 78.8(.7) | 82.5(.99) |
| KWG3: Big Nose-Blonde | 69.4(.78) | 92.5(.7) | 88.9(.67) | 77.8(2.1) | 82.1(.81) | 87.10(1.6) |
| Civilcomments | | | | | | |
| Full evaluation set | 92.9(.4) | 82.0(1.6) | 73.9(1.0) | 90.1(.08) | 88.0(.01) | 91.2(1.9) |
| KWG1:Black-Toxic | 58.3(.55) | 71.7(1.7) | 40.7(2.3) | 52.8(.09) | 88.7(.88) | <u>72.0(2.)</u> |
| KWG2:LGBTQ-Toxic | 53.1(.31) | 83.8(.81) | 43.6(3.1) | 64.2(1.) | 78.4(.92) | <u>66.2(1.5)</u> |
| KWG3: Other Religions-Toxic | 53.1(.67) | 82.1(.91) | 34.0(1.8) | 60.4(.41) | 84.4(.6) | <u>66.7(.8)</u> |

Table 4.2: Average and WG performance on WILDS datasets. AGRO is better at multiple worse groups and also consistently improves over ERM, unlike prior work. Best KWG performance bolded and second-best underlined. (*) indicates variance across 3 random seed runs.

ERM: ERM models trained for longer and with larger batch size already improve worst group performance and are a strong baseline.

4.4 Results

AGRO learns a soft group assignment over training data, with the goal of improving worst-group accuracy over multiple co-occurring correlations. We evaluate this hypothesis on the selected datasets from WILDS (Table 4.1). Table 4.2 reports accuracy on the full evaluation set and on multiple worst-groups.

Across datasets, AGRO improves worst-group accuracy by 25.5% on average over the ERM baseline without hurting the overall task accuracy. In particular, it improves across *multiple* known worst-groups for each dataset. On all datasets except CelebA, for KWG1, AGRO gets within 2% of the accuracy obtained by G-DRO with *ground truth* groups. This shows that AGRO’s unsupervised group discovery and robust optimization performs nearly as well as using the best manually designed groups. No other baseline consistently improves worst-group accuracy over all tasks. Approaches like GEORGE, JTT and EIIL have mixed results across datasets. Their performance depends on specific properties of the test datasets, such as a high spurious correlation ratio (in CelebA) or severe label imbalance (in Civilcomments). These results indicate that AGRO

is most suitable for new datasets whose key properties are unknown. (Paranjape et al., 2022) explores this further.

On **MultiNLI**, DeBERTa-based models achieve the best reported worst-group performance by both the ERM model and the G-DRO model (trained using KWG1 assignments). AGRO has an 8% improvement over the next highest baseline on this task (EIL). GEORGE. The worst group performance of AGRO establishes a new state of the art for group robust optimization approaches that also find groups. On **Waterbirds**, the ERM model finetuned using BEIT has competitive worst-group accuracy, surpassing the performance gap of a weaker model like ViT. We hypothesize that a stronger pre-trained model is generally less susceptible to spurious correlations (Wortsman et al., 2022a). On both worst groups, AGRO outperforms ERM and has comparable performance with the next best baseline GEORGE, once again achieving SOTA ⁴.

CelebA and **Civilcomments** have overlapping groups i.e. images with multiple facial attributes and comments with mentions of multiple demography labels, respectively. Consequently, the performance of G-DRO with ground-truth groups for KWG1 also improves performance on other worst groups. On the Male-Blonde group in **CelebA**, AGRO outperforms the ERM model by 75%. It also outperforms GEORGE and EIL on this group. Owing to a high rate of spurious correlation in the dataset (male:non-male blondes ratio is 0.02), re-weighting misclassified points (JTT) proves to be an effective grouping mechanism. **Civilcomments** has a significant label imbalance (a ratio of 0.13 for toxic to non-toxic, the lowest for all datasets considered in this work). Consequently, using the label-as-group heuristic (EIL) performs the best among other baselines, followed by AGRO.

4.4.1 Discussion

Table 4.3 has examples of slices discovered by AGRO on several datasets. The core goal of AGRO is to discover error-prone groups without prior knowledge of spurious correlations. Human expertise is needed to ascertain whether a model exploits any spurious correlations in the discovered slices (Eisenstein, 2022). Soliciting human feedback is challenging since models may exploit spurious correlations that do not correspond to human-interpretable or identifiable features. Once again, we design a method for interactive debugging that addresses this challenge. We design two annotation tasks to obtain feedback indirectly. The first annotation task evaluates **coherency**. Annotators are shown four instances from a group identified by AGRO, along with two other test instances, only one of which is from the same group.

⁴ At the time of publication of this work

| |
|--|
| WG1: Qualifiers like "all, only, solely" with Neutral class. |
| P: Chapter 5 focuses on synergistic combinations of control retrofits on a single unit. |
| H:Chapter 5 focuses solely on synergy (Neutral → Entail) |
| P:but that that's my that's probably my main hobby that sewing and reading books |
| H:Sewing and reading books are my only hobbies. (N → E) |
| P:Nuns are in their habits and Israeli women in military uniforms. |
| H:All Israeli women are in the military, so they wear military uniforms. (N → E) |
| WG2: Antonyms in P and H with Contradiction class. |
| P: Stand behind it. |
| H: Stand so that it is in front behind of you. (Contradict → Entail) |
| P: Look for the Cecil Hotel on the western end of the square. |
| H: You can see the Cecil Hotel from the eastre west of the square. (C → E) |
| P: Get there early in the morning before the crowds arrive |
| H: You should get there shortly after before they open to beat the crowds. (C → E) |

Table 4.3: Perturbation patterns on worst groups discovered in AGRO. Class flips in ()

agreement has a score of 0.7823 Fleiss Kappa. Humans were able to identify the correct group member 70.1% of the time. Some examples of the flips provided by human annotators in Task 2 are presented in Table 4.3. This *interactive debugging* experiment helps us debug model behavior in a targeted manner and accurately identify *learned* spurious correlations.

AGRO is a more effective off-the-shelf optimization technique for fully unsupervised group discovery and robustness on new classification tasks. One limitation for group discovery techniques like AGRO is the assumption made about the presence of instances with spurious correlations in the training data. In our work on counterfactual data generation, Paranjape et al. (2021a), we demonstrate how potential spurious correlations can be surfaced and used to augment training data. However, applying AGRO to the text generation scenario is significantly more challenging. Robustness for models trained on internet-scale data poses a challenge to the identification of dataset bias and potential learned spurious correlations over trillions of tokens. Such spurious correlations between text-spans result in hallucinations and factual inconsistencies in model-generated text. A robust and granular evaluation of factual errors and hallucinations in machine-generated text (for GPT-4 scale models) is an under-explored area. In Chapter 5, we explore such an evaluation framework in greater detail.

The annotators are asked to identify the test instance that belongs to the group. The second annotation task follows the first, where the annotators are asked to perturb the instance chosen to be part of the AGRO group in such a way that the task model's prediction would change. This exercise can implicitly surface the common *feature* humans think exists in the slice. We conduct the human study on 5 slices each from MNLI, SST2 and CelebA. For Task 1, we find that inter-annotator

Chapter 5

Factuality for large language models

5.1 Overview

Long-form text generations have improved dramatically over the last few years, with the advent of modern closed-source LLMs such as ChatGPT, GPT4 from OpenAI, Claude from Anthropic, etc. Open-source LLMs are also catching up fast, as evident with the LLama family of models, Mistral etc. As text generations become exceedingly human-like and plausible, the objective is to ensure their generations are also *factual*, as many of these models are reported to produce convincing *hallucinations*. Thus, the community needs to continually and critically analyze the model generations for factual correctness. However, such factual evaluations can be both costly and time-consuming to do with human annotators. Thus, a concerted focus in the research community is to devise automatic ways to evaluate factuality of generations (Yue et al., 2023; Bohnet et al., 2022; Min et al., 2023; Chern et al., 2023).

Given a model generation, one way to evaluate its factuality is using *reference-based* systems - comparison of the generated tokens with a retrieved document from a large external knowledge base, and model-based evaluation, where an external model is tasked to compare the document and model generation, generally by formulating the problem as natural language inference (NLI). (Bohnet et al., 2022; Yue et al., 2023). However, long-form text generations can have multiple facts to verify, and the generated text can have a mixture of correct and incorrect facts, making it hard for a single evaluation method to provide fine-grained analysis. Min et al. (2023) propose a novel direction of decomposing the model generation into small *atomic units*, and then performing model-based evaluation of factuality by leveraging an external knowledge base and a powerful, closed source LLM as the evaluator. This method, FactScore, allows for fine-grained evaluation of factuality

of long-form generations. However, the over-reliance of FactScore (and similar methods) on closed-source models poses two serious challenges, the first being *cost*. Min et al. (2023) state that using FactScore, one could save up to 65k\$ USD compared to human annotations. However, usage of closed-source model API's are not cheap either - in our calculations, using FactScore with GPT-4 (including claim generation and evaluation) for a single model generation can cost upwards of 0.5\$. When evaluating model generations *at scale*, the cost can quickly accumulate to a steep value (5,000\$ for 10k evaluations). Furthermore, while FactScore limits grounding to external knowledge (like Wikipedia), these sources do not completely encompass the data used to train the LLM and therefore are insufficient in explaining the source of hallucination.

Therefore, in this work, we propose an end-to-end, robust factuality evaluation framework, OpenFactEval, which can be used to evaluate model generations using purely open models, such as LLama (Touvron et al., 2023). By leveraging the sample efficiency of LLama family of models (Zhou et al., 2023), this framework is developed by instruction-tuning on high-quality, human-annotated datasets, obviating the need for large-scale prompt engineering. Given a model generation, OpenFactEval performs the following: 1) breaks down the input into multiple atomic claims and filters the claims that cannot be properly verified; 2) retrieves documents (or references) for the claim using an external, open-source retrieval framework (Lin et al., 2023a) built over a large index of the pretraining data; 3) verifies whether individual claims can be attributed in a three-way classification setup, additionally backing up its decision with long-form reasoning. Using our recipe, we were able to get competitive scores with FactScore on closed-source models (Chat GPT). This framework is cheaper to use and implement than closed-source models, with requiring only one-time fine-tuning using Q-LORA (Dettmers et al., 2023) on commodity hardware.

5.2 Methodology and Results

The OpenFactEval pipeline first decomposes generation into independent verifiable claims (5.2.1). We then retrieve document chunks using the claim as a query, against an expanded corpus of evidentiary documents compared to prior work (5.2.2). Next, individual claims are verified for factuality using a reasoning-based classification model as attributable, extrapolatory, or contradicting the retrieved evidence (5.2.3). Claim-specific verification results are aggregated into a singular score using a simple aggregation strategy.

5.2.1 Claim Generation

Problem Formulation. Since long-form LLM generation consists of several pieces of information, we adopt the definitions of an atomic fact as described in Min et al. (2023) and atomic content unit (ACU) described in Liu et al. (2022c) — as a short sentence conveying one piece of information. Atomic facts provide a fine-grained evaluation of factuality. While prior work (Min et al. (2023); Chern et al. (2023); Malaviya et al. (2023)) leverages closed-source models like ChatGPT to extract claims using few-shot prompting, we aim to leverage an open-sourced LLM (LLama2, Touvron et al. (2023)).

To give an example of the problem, consider the model generation in Table 5.1. We first break down the generation into atomic facts by few-shot prompting LLama2 70B, using the exact prompts used by Min et al. (2023). We observe that the atomic claims generated has two issues: 1) low recall: the prompting approach does not generate all claims in the model generation, and 2) right branching effect - we observe atomic claims being generated increasingly incrementally from the previous claim (Table 5.1), ending up with longer and redundant claims. This phenomenon can be attributed to the use of a specific prompt by Min et al. (2023), which can be corrected by modifying the prompt in a way to avoid such behavior. However, we found low recall issues to be difficult to completely remove by few-shot prompting. Furthermore, the original prompts by Min et al. (2023) were designed to break down a single sentence. Repurposing the prompt to break down a full passage would require significantly larger input prompt containing multiple input passages as exemplars, thereby limiting the context window length of LLama2.

Setup. To address the above-mentioned issues, we instruct-tune (or finetune with instructions) the LLama2 family of models to generate atomic claims from a given model generation. For instruction tuning, we repurpose three datasets: RoSE (Liu et al., 2022c), Complex claims dataset (Chen et al., 2022a), and the ExpertQA dataset (Malaviya et al., 2023). RoSE dataset consists of paired summary and ACUs (Liu et al., 2022c), Complex claims dataset (Chen et al., 2022a) consists of pairs of political speech excerpts and atomic claims, and the ExpertQA dataset (Malaviya et al., 2023) consists of pairs of LLM generations to expert-domain questions and human-annotated atomic claims. We preprocessed all three datasets to remove subsequences and claims with no entities. We fine-tune a Llama2 model (7B/70B) to generate a concatenated list of claims given the original text. In total, we use 6.5K (*text, atomic claims*) pairs for finetuning.

| Model Generation |
|---|
| " For You " is a song recorded by British singers Liam Payne and Rita Ora for the soundtrack to the film Fifty Shades Freed (2018). It was released on 5 January 2018 through Universal Studios and Republic Records as the lead single from the soundtrack. The music video was released on 26 January 2018. [2] The song was also included on Ora's second studio album, Phoenix (2018), and Payne's debut studio album, LP1 (2019). "For You" reached the top ten in the United Kingdom, France, Norway, Switzerland, Austria, Portugal, Poland, and Belgium, and peaked at number one in Germany. |
| LLama2 70B few-shot prompting |
| <ul style="list-style-type: none"> • " For You " is a song. • " For You " is a song from the soundtrack to the film Fifty Shades Freed. • " For You " was released in January. • " For You " was released in January 2018. • " For You " was released through Universal Studios. • " For You " was released through Universal Studios and Republic Records. |
| Fine-tuned LLama2 70B model in OpenFactEval |
| <ul style="list-style-type: none"> • " For You " is a song • " For You " is recorded by Liam Payne • " For You " is recorded by Rita Ora • Liam Payne is a British singer • Rita Ora is a British singer • " For You " is recorded for the soundtrack to the film Fifty Shades Freed • The soundtrack is through Universal Studios • The soundtrack is through Republic Records • " For You " was released on 5 January 2018 • " For You " was released as the lead single • The song "For You" was included on Liam Payne's LP1 (2019) album. • The song "For You" was included on Rita Ora's Phoenix (2018) album. • "For You" reached the top ten in the United Kingdom. • "For You" reached the top ten in France. • "For You" reached the top ten in Norway. • "For You" reached the top ten in Switzerland. • "For You" reached the top ten in Austria. • "For You" reached the top ten in Portugal. • "For You" reached the top ten in Poland. • "For You" reached the top ten in Belgium. • "For You" peaked at number one in Germany. |

Table 5.1: Claim generation with and without finetuning LLama2. Finetuning helps improve claim recall for longer generations and prevents right-branching effect.

Evaluation. We evaluate the atomic claim generation capabilities of the finetuned model on held-out test sets for each of the three datasets. Specifically, we use CNN-Daily mail test set in RoSE, and test sets for complex claims and ExpertQA. In particular, we measure precision and recall for the generated set of claims and reference set of claims. In particular, generated and reference claims are exact-matched after normalizing them for case and removing English stop-words. We further filter out claims which contain non-verifiable facts, such as subjective or fictional responses from the model. To filter the non-verifiable claims out automatically, we make use of prompts to LLama2 70B, and further evaluate its effectiveness on a held-out test set.

Results. Table 5.2 reports the performance of baseline few-shot prompting approaches and our fine-tuned model on claim generation for RoSE (CNN-dailymail test), Complex claims, and ExpertQA. We observe that Llama2 7B and 70B models significantly lag behind ChatGPT in few-shot prompting for claim generation.

| | RoSE | | Complex Claims | | Expert QA | |
|--------------------|--------------|--------------|----------------|--------------|--------------|--------------|
| | Precision | Recall | Precision | Recall | Precision | Recall |
| Few-shot prompting | | | | | | |
| LLama v2 7B | 18.69 | 17.34 | 23.44 | 26.67 | 26.80 | 27.88 |
| LLama v2 70B | 19.38 | 17.77 | 25.65 | 28.44 | 29.00 | 31.20 |
| ChatGPT | 35.28 | 32.90 | 35.02 | 40.95 | 40.33 | 42.40 |
| Instruction tuning | | | | | | |
| LLama v2 7B | 37.28 | 36.07 | 35.14 | 37.01 | 42.05 | 39.80 |
| LLama v2 70B | 38.99 | 36.08 | 42.01 | 45.89 | 48.79 | 49.03 |

Table 5.2: Evaluating claim generation: Instruction tuning of open-models helps bridge the gap and even surpass performance of few-shot prompting with ChatGPT

Post-finetuning, Llama2 7B matches the performance of ChatGPT, and Llama2 70B finetuning obtains a 4-point gain over ChatGPT. In particular, LLama2 without finetuning suffers from the generation of right-branching atomic facts (Table 5.1) and generates incomplete claims for longer generations. Finetuning LLama2 helps counter both problems. These error patterns are highlighted in Table 5.1.

5.2.2 Evidence Retrieval

Problem Formulation. To evaluate the factuality of a generated claim, the best approach would be to compare it against gold-truth documents. For evaluation in-the-wild, the gold truth documents for any arbitrarily generated claims have to be retrieved from a large knowledge base corpora. Prior work on verifying claims uses search engines and LLMs to query them (Yue et al., 2023; Malaviya et al., 2023), where the supporting evidence is retrieved from a large corpus or obtained via web search. However, a search engine is again of black box nature, and rapid access to its queries is also costly and rate-limited. In our framework, we rely on retrieval over a fixed text corpus that is a subset of the pretraining data used to train the open-sourced LLM (LLama2 in this case). In particular, we retrieve evidence using every generated claim from the previous step as a query.

Setup. We consider two different retrieval approaches and associated corpora:

- *Weak Retriever*, Wikipedia + BM25: We use a snapshot of the Wikipedia dump from 2021-10-13 (which is a subset of the pretraining data for Llama) and construct a sparse BM25 index using Pyserini¹. The Wikipedia paragraph with the highest BM25 score is used for attribution.

¹ <https://pypi.org/project/pyserini/>

- *Strong Retriever*, DRAGON+: We use the DRAGON+ Lin et al. (2023a) dense retriever model used in Lin et al. (2023b) and the retrieval corpus in Lin et al. (2023b) that combines text chunks from the Wikipedia dump with additional ones from the 2017-2020 CommonCrawl dumps used in pretraining Llama2. The final retrieval data store, with the two data sources combined, contains 399M text chunks with a maximum length of 200 words. The same retriever model is also used to encode claims as queries. Once again, the highest-scoring passage is used for attribution.

Evaluation. We evaluate evidence retrieval over four datasets which are commonly used as evaluation benchmarks for factuality evaluation - GSE, (Liu et al., 2023), Attreval-Sim, Attreval-Gen (Ye et al., 2021), and ExpertQA (Malaviya et al., 2023). These datasets contain human-verified reference passages that can be used to verify LLM-generated text. To quantify the relative quality of different retrieval strategies, we report Rouge-1 between the top-1 retrieved passage and reference passages for every claim (query).

| | GSE | Simulated | GenSearch | ExpertQA |
|----------|--------------|--------------|--------------|--------------|
| Pyserini | 13.33 | 10.09 | 11.12 | 12.70 |
| DRAGON | 18.50 | 14.40 | 15.35 | 14.09 |

Table 5.3: Comparing different retrieval methods (Rouge-L on top-1 retrieval.) on claim-evidence datasets.

Results. Table 5.3 reports the performance of BM25 sparse retrieval over Wikipedia and dense retrieval (DRAGON+) on a larger index of the common crawl pre-training data used in Llama2. Unsurprisingly, a strong retriever outperforms a weak retriever significantly. Relevance of the evidentiary document increases significantly when a sparse lexical-match retrieval approach and a smaller corpus (Wikipedia, 37M passages) are replaced by a dense retrieval approach that can handle subtle semantic similarity cues and uses a 10X larger corpus (Wikipedia + CC, 399M passages).

5.2.3 Claim Verification

Problem Formulation. Given an atomic claim and a retrieved document (or henceforth, evidence), FactScore (Min et al., 2023) perform claim verification on the generated claims using an LLM evaluator, where the conditional probability of the verification labels, `true` and `false`, is measured in the LLM zero-shot output. In the case of closed-source models, Min et al. (2023) suggests measuring whether the verification labels occur in the generation of the LLM evaluator model. This adds another level of dependence

on the closed-source models, which in turn increases the cost of evaluation and reduces reproducibility. We therefore look into open-source models for claim verification.

Prior work cast claim verification as an NLI task (Bowman et al., 2015) framework, which consists of three-way classification of claim-evidence pairs (Bohnet et al., 2022; Rashkin et al., 2023; Malaviya et al., 2023; Yue et al., 2023). In these works, the evaluator model is first trained with a mix of NLI, QA, fact-checking, and summarization datasets, which enables the model to verify atomic claims given evidence. However, as noted in Yue et al. (2023), this pair-wise classification is limited in its insensitivity to fine-grained information comparisons, such as overlooking contextual cues in the reference, disregard for numerical values, and failure to perform symbolic operations.

Setup. In OpenFactEval, we again use LLama2 (Touvron et al., 2023) family of models to train the evaluator. We choose the NLI formulation from Bohnet et al. (2022), where the objective of the model is to generate the verification labels "Attributable", "Contradictory" and "Extrapolatory". This allows OpenFactEval to label claims that can be correctly verified, refuted, or deemed irrelevant with respect to the evidence document. Given the recent developments on using step-by-step reasoning to solve complex natural language understanding tasks (Wei et al., 2022b; Zhou et al., 2022; Press et al., 2022; Paranjape et al., 2023), we hypothesize that the claim verification step can also be improved by having the evaluator model reason about the verification label. Therefore, we instruction-tune LLama2 to perform claim verification while simultaneously generating the intermediary reasoning behind the verification label. To do so, we curate a dataset of step-by-step reasoning for *(claim, evidence, label)* tuple, by sampling equally from the training set of three datasets: AttrEval-GenSearch (Liu et al., 2023), AttrEval-Simulation (Yue et al., 2023), and ExpertQA (Malaviya et al., 2023)). To add the reasoning for each *(claim, evidence, label)* tuple, we first prompt LLama2 70B to generate a free-form reasoning, given the input *(claim, evidence)* pair. We de-noise the generated reasoning excerpts by manual annotation to curate a high-quality training dataset containing 500 instances of *(claim, evidence, label, reasoning)* tuples. We further augment this dataset by sampling 10K NLI-style three-way classification data from AIS scoring training data (Yue et al., 2023). Using this combined dataset, we fine-tune LLama2 (7B and 70B) to obtain OpenFactEval, which is capable of generating step-by-step reasoning and classifying atomic claims in three classes given the evidence.

| Free-form reasoning annotation for attribution |
|--|
| <p>Claim: The capital of Qasr-e Shirin County is Qasr-e Shirin</p> <p>Reference: There is also a Payame Noor University. Qasr e shirin is located on one of the most important routes in Iran. Road 48 (Iran) passes through Qasr e shirin and ends in Khosravi, which is part of Karbala Freeway. Every year hundreds of thousands of tourists go to visit Iraqi Kurdistan through Parwiz border point border terminal located north of the city. Most of the residents travel by car through the system of roads and highways. There are many private travel agencies called phone-taxi providing inner city and inter city services to travelers. The city is near a proposed railway</p> |
| <p>LLama2: Attributable. The reference states that Qasr-e Shirin is located on one of the most important routes in Iran and provides information about the roads and highways in the city, supporting the claim that it is the capital of Qasr-e Shirin County.</p> <p>Corrected: Extrapolatory. The reference does not state any information about whether Qasr e shirin is the capital city of the country.</p> |

Table 5.4: Reasoning data annotation: Annotators can correct both fact verification labels and associated reasoning by evaluating the claim against the reference chunk.

Evaluation. We evaluate the claim attribution capabilities of OpenFactEval on existing attribution evaluation benchmarks that have human-annotated labels for model full generations and atomic claims. Specifically, we compare OpenFactEval performance with three models: FactScore (Min et al., 2023) with Chat GPT as LLM evaluator, FactScore with LLama v2 as LLM evaluator, and the AttributeScore model (Yue et al., 2023). Furthermore, we perform evaluation on two axes - when we have access to gold documents, and when we use an open-source retriever model to retrieve the gold document.

We report label accuracy for GSE, Attreval-Sim, Attreval-Gen, and ExpertQA to evaluate the effectiveness of our reasoning-based attribution model. In particular, if datasets contain a single label for the entire generation, we adopt a majority-voting aggregation technique to obtain a single label from the set of claim-wise attribution labels obtained from our fine-tuned model.

| | GSE | Attreval Sim | Attreval GenSearch | ExpertQA |
|----------------------|--------------|--------------|--------------------|--------------|
| Gold Documents | | | | |
| FactScore (Chat GPT) | 44.02 | 67.50 | 70.02 | 84.62 |
| FactScore (LLaMa V2) | 24.56 | 58.77 | 55.92 | 80.20 |
| AttributeScore (AIS) | 39.60 | 61.45 | 61.07 | 88.94 |
| OpenFactEval (Ours) | 43.4 | 64.50 | 67.30 | 90.80 |
| Retrieved Documents | | | | |
| FactScore (Chat GPT) | 33.29 | 56.04 | 59.88 | 75.98 |
| FactScore (LLaMa V2) | 32.33 | 45.06 | 48.76 | 70.43 |
| AttributeScore (AIS) | 27.90 | 40.78 | 37.70 | 72.34 |
| OpenFactEval (Ours) | 34.00 | 46.22 | 51.06 | 78.33 |

Table 5.5: Evaluating attribution: Accuracy of label prediction with majority voting over claims if per-claim label is not available. Retrieved documents are from Pyserini-Wiki.

Results. Table 5.5 reports the performance of baseline and OpenFactEval. In case of FactScore, we observe LLaMA V2 evaluator underperforms ChatGPT. Finetuning the model for attribution classification (AttributeScore) helps bridge the gap between open and closed source models. Furthermore, adding additional open-ended reasoning/justification for claim verification in OpenFactEval further improves the performance. When gold documents are available, we observe FactScore with ChatGPT outperforms OpenFactEval in 3/4 datasets, with the exception of ExpertQA. However, when gold documents are not available and the alternative is to resort to retrieval of evidence, OpenFactEval becomes competitive with FactScore (with ChatGPT), with better performance in 2/4 datasets. Using only 500 high-quality annotated data, we were able to align LLaMA family of models to perform claim verification along with reasoning, which is due to its remarkable ability to fine-tune in a sample efficient setup (Zhou et al., 2023).

| OpenFactEval in action on ExpertQA |
|---|
| <p>Prompt: What does research currently say regarding the use of cryotherapy to prevent peripheral neuropathy in patients receiving paclitaxel?</p> <p>ChatGPT Generation: Research currently indicates that cryotherapy, specifically the use of frozen gloves and socks, may help prevent peripheral neuropathy in patients receiving paclitaxel chemotherapy. A randomized controlled trial found that cryotherapy could reduce sensory neuropathy symptoms, the need for dose reduction, and the incidence of severe peripheral neuropathy in patients receiving weekly paclitaxel for breast cancer treatment.</p> |
| <p>Claims:</p> <ul style="list-style-type: none"> • Research currently indicates that cryotherapy may help prevent peripheral neuropathy. • Cryotherapy specifically involves the use of frozen gloves and socks. • Cryotherapy is being studied in patients receiving paclitaxel chemotherapy. • The patients had breast cancer. • The trial was investigating the effects of cryotherapy on breast cancer treatment. |
| <ul style="list-style-type: none"> • Research currently indicates that cryotherapy may help prevent peripheral neuropathy. <i>Extrapolatory The reference mentions that cryotherapy has been studied for patients with chemotherapy-induced peripheral neuropathy (CIPN), and the results are largely inconclusive.</i> • Cryotherapy specifically involves the use of frozen gloves and socks. <p><i>Retrieved support: Chemotherapy-induced peripheral neuropathy (CIPN) treatment. Cryotherapy for patients with CIPN involves wearing frozen gloves and socks or ice packs to prevent and relieve symptoms. Various studies have been conducted to trial cryotherapy or cryocompression therapy, and their results are largely inconclusive...</i></p> <p><i>Attributable. The reference states that cryotherapy for patients with CIPN involves wearing frozen gloves and socks, supporting the claim.</i></p> <ul style="list-style-type: none"> • Cryotherapy is being studied in patients receiving paclitaxel chemotherapy. <i>Attributable The reference states that a pilot clinical trial is being conducted to study the utility of topical cryotherapy in reducing pain in patients receiving paclitaxel chemotherapy, supporting the claim.</i> • ... • The patients had breast cancer. <i>Extrapolatory The reference mentions that patients with breast cancer were treated with paclitaxel, but it does not specifically state that the patients in the study had breast cancer.</i> • The trial was investigating the effects of cryotherapy on breast cancer treatment. <i>Attributable The reference provides information about the treatment of breast cancer with cryotherapy.</i> |
| <p>OpenFactEval : 11 Claims (7 Attributable, 4 Extrapolatory)</p> |

Table 5.6: OpenFactEval for ExpertQA: Claim generation and reasoning-based attribution for individual claims. OpenFactEval labels each claim as **Attributable**, **Contradictory** and **Extrapolatory**.

5.3 Discussion

OpenFactEval is able to both break down a given model generation into atomic claims and verify their factuality. In Table 5.6, we demonstrate OpenFactEval in action on a prompt from ExpertQA and the response generated by ChatGPT. In the example, OpenFactEval provides a systematic breakdown of the generation into 11 independently verifiable claims using the fine-tuned claim generation model (Section 5.2.1). OpenFactEval also performs retrieval for every individual claim (Section 5.2.2) to get relevant evidence documents, and generates the label along with valid reasoning (Section 5.2.3). Ultimately, OpenFactEval provides a breakdown for attribution over different claims, providing users with a more informative summary of the factuality of the LLM’s generation. Unlike prior work that provides a singular score for attribution (Min et al., 2023), this granular report consists of claim-wise attribution and reasoning in natural language. Furthermore, the statistical breakdown of factuality (as shown in Table 5.6) enables users to selectively use or moderate model-generated content.

Chapter 6

Tool Augmentation and Complex Reasoning

6.1 Overview

In this part of the thesis (Chapter 6), we turn to a different challenge for reliability—sample-efficient generalization. Users expect the model to effectively generate accurate responses for a multitude of heterogeneous and previously unseen tasks, encompassing intricate multi-step reasoning and interaction with external systems and tools. Portions of this chapter appeared in Paranjape et al. (2023).

In-context learning allows large language models (LLMs) to quickly adapt to new tasks simply by using natural language instructions and a few demonstrations as a prompt to the LLM (Xie et al., 2021; Brown et al., 2020; Chowdhery et al., 2022). While this circumvents annotating large datasets or even hosting the LLM itself (since many are available through APIs), there are severe performance limitations around multi-step reasoning Liu et al. (2022b), math Patel et al. (2021), having up-to-date information Komeili et al. (2022), and others. To address these limitations, recent work proposes prompting LLMs to mimic a chain of thought (CoT) for multi-step reasoning (Wei et al.; Zhou et al., 2022; Wang et al., 2022; Press et al., 2022; Khot et al., 2022; Arora et al., 2022) or providing them with access to tools (e.g. a calculator or QA model) to enable more complex reasoning steps (Gao et al., 2022; Chen et al., 2022c; Press et al., 2022; Wei et al.; Schick et al., 2023). However, existing methods for chained reasoning with tool use are difficult to extend to new tasks and tools, requiring fine-tuning or prompt-engineering tailored for a specific task Parisi et al. (2022) or tool Schick et al. (2023).

We present **Automatic Reasoning and Tool use (ART)**, a framework that automatically generates decompositions (multi-step reasoning) for instances of new tasks. The framework also selects and uses the most

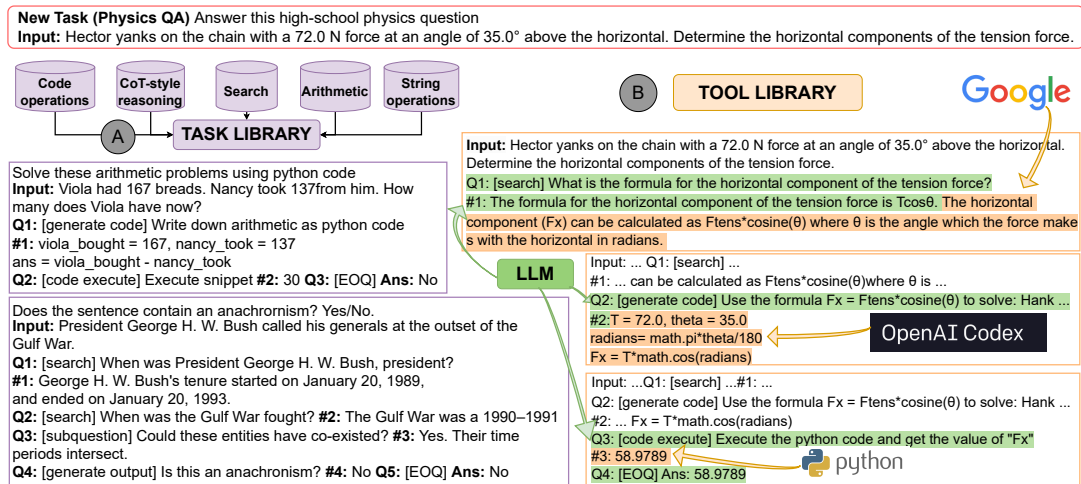


Figure 6.1: A run-through of ART on a new task, Physics QA. (A) Programs of related tasks like anachronisms and Math QA provide few-shot supervision to the LLM — related sub-steps and tools in these programs can be used by the LLM for cross-task generalization (shown in purple). (B) Tool use: Search is used to find the appropriate physics formula, and code generation and execution are used to substitute given values and compute the answer (shown in orange).

appropriate available tools (like search engines, and code execution) in individual steps. Given a new task, ART retrieves demonstrations of related tasks from a *task library* to enable few-shot decomposition and tool use. These demonstrations follow a flexible but structured query language Beurer-Kellner et al. (2022), such that it is easy to parse intermediate steps, stop generation to call external tools, and resume it after including the output of such tools (Figure 6.1). ART can generalize from demonstrations to decompose a new task and use tools in appropriate places, zero-shot. Users can fix any mistakes in the reasoning chain or add new tools by simply updating the task and tool libraries, providing new demonstrations where necessary.

We evaluate ART on 19 unseen *test tasks* from BigBench, 6 MMLU tasks, and various tasks used by related work on tool use (SQUAD, TriviaQA, SVAMP, MAWPS). ART consistently matches or outperforms automatically generated CoT reasoning chains on 32 / 34 BigBench and all MMLU tasks, by an average of over 22 percentage points. Tool-use in particular improves performance on test tasks by an average of over 12.3% compared to when no tools are allowed (Table 6.1). ART improves over direct few-shot prompting by 10.8% percentage points on average across unseen BigBench and MMLU tasks. Finally, ART enables human intervention and improvement of the reasoning process by simply updating the task and tool libraries with new demonstrations. On 12 test tasks, ART with additional human feedback surpasses the best-known results for GPT3 (at the time of publication of this work) by an average of over 20% points (Table 6.2).

6.2 Method

In Figure 6.1, ART is presented with a new task description and input instance. We also assume access to a few input-output pairs (not shown), with no decomposition or tool use supervision.

Prompt building. ART retrieves similar tasks from a *task library* (Figure 6.1(A); Section 6.2.1), and adds instances of those tasks as demonstrations in the prompt. A demonstration in the task library is written in a specific format, defined by a custom *parsing expression grammar (PeG)* (Section 6.2.1). The grammar is defined such that each task instance is decomposed into a sequence of sub-steps. Some of these sub-steps contain symbols corresponding to tools in a *tool library* (Section 6.2.2). We refer to these decompositions as programs, since the sequential reasoning steps and symbolic calls to tools are similar to a conventional program with function calls.

The resultant prompt consists of programs from related tasks and teaches the LLM how to effectively decompose instances of a new task—related sub-steps and tools in these programs can be used by the LLM for cross-task generalization. The demonstrations include calls to both search and code tools (Figure 6.1(A)).

Generation. At generation time (Figure 6.1(B)), the LLM writes its own program. ART parses the program as it is generated, and pauses generation whenever a tool call is encountered in the generated text, resuming generation after the tool is called and its output is integrated back into the program. As illustrated in the figure, a search engine is used to find the appropriate physics formula, and then the LLM uses code generation and execution to substitute the given values and compute the answer.

Human feedback (optional). Humans can add new decomposition demonstrations to the task library, or add/edit tools in the tool library in order to improve performance on a particular task of interest, or in general. In Figure 6.2(C) a user corrects a specific program by including a step that adds the unit of measurement, and adds this (modified) program to the task library. While most of our experiments do not use such feedback, we show that it is very effective at drastically improving performance when task generalization does not happen automatically. Further, it gives users flexibility to add custom tools without retraining of the LLM.

6.2.1 Task Library

We construct a library of programs for a small seed set of tasks from Big-Bench (Srivastava et al., 2022), a collaborative benchmark that measures the capabilities and limitations of language models. Big-Bench tasks span categories of traditional NLP, mathematics, commonsense reasoning, and question-answering.

Constructing the task library. We identify *five* skills that are useful across more than half of the tasks in BigBench that encompass text classification or generation of short answers in English. We group tasks in the benchmark by these skills into the following clusters:

- Arithmetic: arithmetic and algebra problems.
- Code: Generating and executing python code.
- Search and question decomposition: Single or multi-step questions that require search
- Free-form reasoning: Explaining step-by-step reasoning in natural language
- String Operations: Reformatting/editing strings, checking string entailment, etc.

We then select 2-4 tasks from each cluster and write programs (decompositions) for a few instances of each task, including calls to external tools and real outputs of those tools. Examples of programs in each cluster are in the Appendix (Tables A.2,A.3, A.4,A.5,A.6). These programs follow a specific grammar, as outlined below.

Program grammar The program format must be flexible in terms of task inputs, steps, and tool calls, such that a wide variety of NLP tasks can be covered. To do so, we define a query language Beurer-Kellner et al. (2022) that extends the decomposed prompting format of Khot et al. (2022), since it can represent decomposed reasoning steps sequentially and incorporates function calls to external tools (like other LLMs). Each program consists of a series of *nodes* — a task input node, several sub-step nodes, and an answer node. The *input node* contains the task name, a simple instruction describing the task, and the input for an instance of the task: “Answer this high-school Physics question.

Input: Hector yanks...”. The input node is followed by a sequence of sub-task nodes, represented as a (query, answer) pair “ $Q_i : \dots, \#i : \dots$ ”. The sub-task query Q_i has a sub-task name and sub-task input (“Q1: [search] What is the formula...”), while the sub-task answer $\#i$ is simply the output of the sub-task (“#1: The horizontal

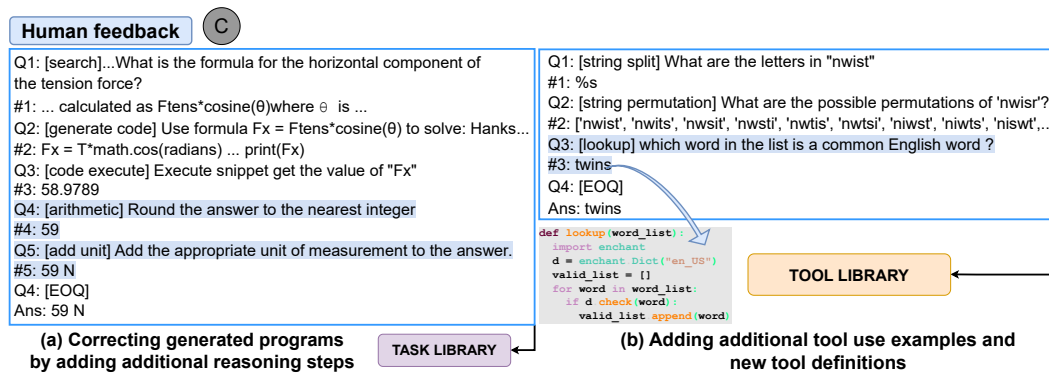


Figure 6.2: Human feedback to ART shown for (a) PQA where reasoning steps are added to the program and; (b) Word unscrambling where tool library is augmented with a new lookup tool.

component (F_x) can be calculated...”). The program ends with a dummy sub-task (“Q3: [EOQ]”), followed by a final answer node (“Ans: 59N”). All examples in Figure 6.1 follow this format.

Task Retrieval Given a new task, ART retrieves N tasks from the task library to construct a dynamic multi-task prompt. If a small number of labeled examples for the new task is available (≈ 50), we iterate over all five task clusters and select a few task programs from each cluster to compose the prompt. Ultimately, the task cluster with the highest performance on the held-out set of examples is chosen when predicting on all unlabeled examples from the task. While this strategy requires a held-out set of input-output pairs, no additional supervision is needed to generate a decomposed program.

6.2.2 Tool Library

Whenever a sub-task query name matches a tool name in the task library (e.g. “ Q_i : [search]”), generation is stopped and resumed after the tool is called and its output is incorporated into the partially completed program. We seed the tool library with the following tools (all of which have demonstrations in the task library). In particular, we describe the symbols used to represent these tools and their inputs. We also specify how the tool output is incorporated back into the program.

Search We use SerpAPI¹, which provides an API for Google search. The input to search is the sequence generated by the LLM after “ Q_i : [search]”. We extract answer box snippets when they are available or combine the top-2 search result snippets together. For PQA in Figure 6.1(B), the search query is the original

¹ <https://serpapi.com>

input followed by “What is the formula for the horizontal component of tension force?”, and the output is “... horizontal component (F_x) can be calculated as $F_{tens} \cdot \cos(\theta)$...”.

Code Generation We use the Codex (Chen et al., 2021) model for code generation. Input to code generation is the sequence generated by the LM after the sub-task query symbol “ Q_i : [generate python code]”. This argument is an instruction for code generation and is prompted to Codex as a multi-line comment in Python. For example, in Figure 6.1, Codex is prompted the instruction ““Use the formula $F_x = F_{tens} \cdot \cos(\theta)$ to solve...”” as a comment and generates “ $T = 72.0$, $\theta = 35.0$, ..., $F_x = T \cdot \text{math.cos(radians)}$ ”, which is appended to the incomplete program.

Code Execution We run Python code in a virtual Python environment with arithmetic, symbolic, and scientific computing packages pre-installed. The argument to code execute is the previous sub-task’s answer sequence “ $\#(i - 1) : \dots$ ”, i.e. the python code snippet to be executed. For $i = 1$, the task input is used as the argument since it potentially contains the code snippet to be executed. In Figure 6.1, the code snippet generated in the previous step is executed and the value of variable “ F_x ” is added to the incomplete program.

6.2.3 Human feedback

ART is specifically designed to be amenable to human feedback since it does not require additional finetuning. Consequently, users can incorporate feedback immediately into ART, by editing the task library and/or the tool library. Since ART generates multi-step reasoning programs that are interpretable, we explore feedback in the form of debugging, i.e. users *edit* existing programs rather than creating programs from scratch. These edits can be in the form of correcting sub-step outputs, adding/removing sub-steps (with appropriate inputs and answers), adding calls to new tools, etc.

For example, in Figure 6.2(a) the user edits a program by adding two sub-steps, in order to round the answer to the nearest integer and include the appropriate unit of measurement to the answer. This feedback demonstrates appropriate *decompositions* for the task, as these operations are still performed by the LLM (the tool library does not have “[arithmetic]” or “[add unit]” APIs). In contrast, in Figure 6.2(b) the user demonstrates the use of a dictionary “[lookup]” and implements it as a tool in the tool library. While most of our experiments do not rely on such feedback (and thus measure “zero shot” task transfer with no supervision for reasoning/tool-use), we show that simple operations like these can drastically improve performance on target tasks.

6.3 Experimental Setup

Evaluation Datasets In addition to 15 tasks in the task library (Section 6.2.1), we evaluate ART on 19 additional test tasks from BigBench which also belong to the five task clusters identified in Section 6.2.1. To check for cross-benchmark generalization, we further evaluate ART on a random subset of tasks from the MMLU benchmark (Hendrycks et al., 2020). Finally, we also evaluate on a subset of tasks used to evaluate Toolformer (Schick et al., 2023), in order to compare ART to a model fine-tuned for tool use. Details about hyperparameters and models used can be found in (Paranjape et al., 2023).

Baselines ART proposes an automatic framework to generate multi-step reasoning decompositions and use relevant available external tools within those decompositions. We compare with the following baselines:

- **Few-shot/Direct:** Prompting LLMs with input-output pairs (but no intermediate reasoning). We use 3 examples for BigBench and 5 examples for MMLU, as done in prior work (Suzgun et al., 2022). We evaluate this baseline for both, GPT-3 and Codex, and report the higher of the two.
- **Auto-CoT:** A baseline that automatically generates multi-step reasoning in natural language. A random subset of 5 examples is first used to elicit CoT-style reasoning (*Input + Let's think step-by-step.*). These examples and their generated output form the prompt for other unseen examples of the task. This baseline is free-form and does not include tools, and thus allows us to verify the effectiveness of our query language and task library. We evaluate this baseline for GPT-3.
- **ART-tool:** ART with tool-use turned off, i.e. the LLM generates the output of every substep, to verify the gains from tool use.
- **GPT-3 Best:** Best published GPT-3/Codex (175B) result with multi-step decomposition and/or tool use. These often include additional human supervision to decompose reasoning steps, and external tools to boost performance (with carefully constructed prompts).

6.4 Results

We evaluate ART (without human feedback) on tasks in the task library (6.4.1), and on a variety of test tasks from BigBench, MMLU, and QA benchmarks (6.4.2). Then, we show that ART can be further improved with more compute (self-consistency) and with human feedback (6.5).

| Task Name (Cluster) | Few Shot | AutoCot | ART w/o Tool Use | ART | GPT-3 Best |
|--|--------------------|---------------|---------------------|-------|--------------------|
| Test Tasks | | | | | |
| Sentence Ambiguity (Search) | 70.67 ⁵ | 51.47 | 71.00 | 73.33 | - |
| Strategy QA (Search) | 55.49 ⁵ | 27.22 | 59.37 | 66.44 | - |
| Physics (Search) | 70.09 ⁵ | 61.83 | 59.13 | 67.55 | - |
| 1-6 Δ with ART (Search) | +3.7 | +22.27 | + 5.9 | | |
| 1-6 Physics Questions (Arithmetic) | 7.02 ⁵ | 5.56 | 6.30 | 20.37 | - |
| Operators (Arithmetic) | 71.23 ⁷ | 75.52 | 71.80 | 92.00 | - |
| Unit interpretation (Arithmetic) | 58.2 ⁷ | 41.20 | 51.4 | 53.99 | - |
| Repeat copy logic (Arithmetic) | 50.01 ⁷ | 15.63 | 31.25 | 44.38 | - |
| Object Counting (Arithmetic) | 39.2 ⁷ | 26.80 | 42.2 | 87.00 | 81.20 ¹ |
| Penguins in a table (Arithmetic) | 58.23 ⁷ | 40.40 | 68.86 | 77.85 | 72.34 ¹ |
| Reasoning about objects (Arithmetic) | 71.00 ⁷ | 33.33 | 45.35 | 64.34 | 52.69 ¹ |
| Tracking shuffled objects (Arithmetic) | 22.39 ⁷ | 19.44 | 18.14 | 37.67 | 36.32 ¹ |
| 1-6 Δ with ART (Arithmetic) | +19.0 | +36.7 | + 23.1 | | +6.1 |
| 1-6 Word Unscramble (String) | 40.72 ⁷ | 32.44 | 23.03 | 42.7 | - |
| Simple Text Editing (Code) | 35.31 ⁵ | 30.21 | 20.74 | 27.65 | - |
| CS Algorithms (Code) | 73.48 ⁷ | 0.0 | 41.59 | 88.11 | - |
| Sports Understanding (CoT) | 69.74 ⁵ | 51.47 | 92.89 | - | 86.59 ¹ |
| Snarks (CoT) | 54.58 ⁵ | 57.24 | 57.13 | - | 65.2 ¹ |
| Disambiguation QA (Free-form) | 55.03 ⁵ | 48.45 | 55.89 | - | 60.62 ¹ |
| Temporal sequences (CoT) | 55.80 ⁷ | 19.70 | 49.5 | - | 81.8 ¹ |
| Ruin names (CoT) | 71.01 ⁵ | 55.28 | 60.22 | - | - |
| 1-6 Δ with ART (Misc) | 2.4 | 22.5 | 24.37 | | -9.4 |
| 1-6 Δ with ART (Overall) | +6.9 | +24.6 | +16.7 | | -1.7 |
| MMLU | | | | | |
| College Computer Science (Search) | 41.00 | 43.99 | 63.40 | 67.80 | 63.6 ⁶ |
| Astronomy (Search) | 62.10 | 41.48 | 76.71 | 79.1 | 62.5 ⁶ |
| Business Ethics (Search) | 61.60 | 48.8 | 77.17 | 81.16 | 72.7 ⁶ |
| Virology (Search) | 50.03 | 49.52 | 71.60 | 71.49 | 50.72 ⁶ |
| Geography (Search) | 77.67 | 57.07 | 70.30 | 71.71 | 81.8 ⁶ |
| Mathematics (Arithmetic) | 36.67 | 33.77 | 39.50 | 45.66 | 34.5 ⁶ |
| Δ with ART (MMLU) | +14.6 | +23.7 | +3.0 | | +8.5 |

Table 6.1: ART performance on BigBench tasks and MMLU tasks. (¹ Human-crafted CoT (Wei et al., 2022b; Suzgun et al., 2022), ⁵ InstructGPT (Ouyang et al., 2022), ⁶ Scaled instruction finetuning (Chung et al., 2022), ⁷ Code-davinci-002 (Chen et al., 2021)).

6.4.1 Results on the task library

For tasks in the task library, demonstrations in the prompt include two instances of the task itself, along with other instances from tasks in the same cluster. For brevity, results on the task library can be found in (Paranjape et al., 2023). Even with decomposition demonstrations for only two instances, ART drastically improves performance over few-shot learning (+14.9 % points on average), in line with prior work on CoT. It does not do as well on language games, code description, and auto debugging — tasks that use code generation and/or code editing models. We observe that code generation errors often lead to cascading errors in reasoning.

Similarly, ART outperforms AutoCoT on most tasks even without any tool use (by 8% points on average). We hypothesize that the program format (and PeG grammar) is better at eliciting multi-step reasoning from models than free-form CoT due to the added structure to the reasoning. When tool use is turned on, ART outperforms AutoCoT on all tasks (+17.7 % points) minus one. Tools are called in $\approx 95\%$ of test instances, and significantly improve performance (+7.91 % points). Gains from tool use are particularly significant for arithmetic tasks that benefit from representing the arithmetic problem as code that executes complex arithmetic accurately (+21.85 on average). This has also been noted in prior work (Chen et al., 2022c; Gao et al., 2022). In sum, ART is stronger than few-shot learning and AutoCoT on the library tasks (with the 2 labeled decompositions), and comparable to the best published GPT-3 results.

6.4.2 Test tasks (cross-task transfer)

We measure cross-task generalization on test tasks where ART does not use explicit supervision for decomposition and tool use. ART retrieves demonstrations from the task library according to the first strategy in Section 6.2.1, which uses a small amount of labeled input-output pairs to pick a task cluster and sample demonstration programs from that cluster.

BigBench test tasks Even though there is no decomposition or tool use supervision, the results in Table 6.1 are similar to those for tasks in the task library. ART outperforms few-shot learning (6.9%). In particular, ART has significant improvements on arithmetic tasks (+19.0) and is comparable to the few-shot performance on search tasks. Non-grammatical choices in *ruin names* and choices not in the input in *temporal sequences* are often incorrect, which the few-shot baseline may potentially learn to ignore; ART attempts to explicitly reason about them. As with library tasks, string manipulation tasks like simple text editing, word unscrambling, and repeat copy logic suffer from code generation errors.

As observed in the case of library tasks, ART is better than AutoCoT on almost all tasks (24.6 % points). Tools are once again called very frequently (89% of instances), and are responsible for a significant fraction of the gains over baselines. When compared to the best published GPT-3 results, ART performs favorably on average, especially on arithmetic tasks (+6.1 % points). As before, it does worse in tasks where good human demonstrations of how to decompose *the task itself* (provided by Suzgun et al. (2022)) have a big impact. We re-evaluate ART with more human feedback on these tasks in Section 6.5, but even without that we conclude

that ART is competitive on BigBench even when we do not have supervision for decompositions for the task at hand (i.e. there is cross-task generalization).

| Task | CoT +Human | | ART + Human | | GPT-3 Best | Human Feedback |
|-----------------------|---------------|-------------|----------------|--------------|---------------|--|
| CS Algorithms | 0.0 | 23.0 | 88.11 | 92.73 | 73.48 | C: longest common subsequence code |
| Reasoning about objs. | 33.33 | 67.75 | 64.34 | 98.90 | 71.00 | C: Define object, color, count data structure |
| Repeat Copy Logic* | 15.63 | 45.22 | 44.38 | 80.31 | 50.01 | C: string edit operation |
| Sentence Ambiguity | 51.47 | 72.33 | 73.33 | 83.67 | 70.67 | C: Constrain queries to extract relevant info. |
| Simple Text editing* | 30.21 | 35.31 | 27.65 | 36.11 | 35.31 | C: string edit operation |
| Strategy QA* | 27.22 | 29.19 | 66.44 | 69.15 | 55.49 | C: Constrain queries to extract relevant info. |
| Physics* | 61.83 | 68.21 | 67.55 | 72.55 | 70.09 | A: [search] Formula that connects mass, ... |
| Temporal Sequences | 19.70 | 30.22 | 49.5 | 88.00 | 81.8 | A: [subquestion] Is X free Yam to Zam? |
| Track Shuffled objs. | 19.44 | 36.48 | 37.67 | 99.86 | 36.32 | C: Define object pair data struct, swap logic |
| Unit Interpretation* | 41.2 | 41.2 | 53.99 | 95.0 | 58.2 | A: [add unit] Add the right unit to the answer |
| Word Unscrambling* | 32.44 | 33.40 | 42.70 | 62.11 | 40.72 | T: lookup permutations in dictionary |
| Average | 30.2 | 43.8 | 56.0 | 79.85 | 58.5 | |

Table 6.2: Improving ART and free-form CoT via human-in-the-loop feedback. (*) indicates that human-in-the-loop improvement was done over automatically generated CoT reasoning for these tasks. Feedback for ART includes correcting sub-steps in programs (“C:”), adding additional sub-steps(“A:”), and defining new tools(“T:”). Note that only five examples were edited for each task.

Other benchmarks To make sure ART does not overfit to BigBench-style tasks, we evaluate performance on randomly selected tasks from the MMLU benchmark Hendrycks et al. (2020) in Table 6.1, where ART is more effective than all baselines on 5/6 tasks (+8.5 points better than few-shot baseline on average), despite having no supervision for demonstrations or tool use. MMLU requires extensive world knowledge, and thus most of these tasks benefit from the search tool.

6.5 Discussion

We also pilot the use of task-specific feedback in Table 6.2, by having one of the authors *edit* 5 random instances of model-generated programs that resulted in errors for each task. When editing, we correct errors in sub-steps (denoted as “C:”), adds missing substeps (“A:”), or defines a new tool and demonstrates its use (“T:”). For example, this involved introducing an “add unit” sub-step for the PQA task, and implementing a dictionary lookup function as a tool for the “Word Unscrambling” task (both illustrated by Figure 6.2). We also compare human feedback applied to CoT-style reasoning. Suzgun et al. (2022) already provide reference CoT-style reasoning for some tasks.

In both cases, editing programs and adding them as demonstrations leads to significant gains in performance on the task at hand. However, the gain is much more dramatic in ART, leading it to consistently outperform the best published GPT-3 baseline (at the time of publication of this work) for the task at hand.

Further, these corrected programs and tools can be added to the task and tool libraries, and our prior results in Table 6.1 suggest that they potentially help improve ART on other tasks as well. This pilot indicates that besides being competitive on cross-task generalization, ART is very amenable to task-specific improvement with minimal human intervention.

Our main contributions include a lightweight grammar to represent multi-step reasoning as a program (with tool calls and arguments), an extensible library of seed tasks for which programs are authored, and a tool library that consists of useful external utilities like search, code generation, and execution. The interpretable reasoning framework also allows humans to improve task decomposition and tool use to boost performance. Users can interactively debug the generated program to iteratively improve task performance. ART achieves a substantial improvement over few-shot prompting and automatic generation of CoT reasoning on unseen tasks in the BigBench and MMLU benchmarks, and substantially exceeds performance on hand-crafted CoT prompts when human feedback is incorporated. Model augmentation and interaction with tools is just the start of a larger movement towards increasingly interactive models. In Chapter 7, we research challenges to reliability for more interactive models that enable multimodal and embodied interactions, as well as personalized models tailored to individuals and organizations.

Chapter 7

Conclusion

In this chapter, we explore future avenues for enhancing the reliability of LLM-based applications. Specifically, we discuss research directions for models that are capable of multimodal and embodied interactions, as well as personalized models tailored to individuals and organizations.

Multimodal, embodied and personalized models As a consequence of the homogenization of approaches to train foundation models, there is also homogenization of models across modalities. Multimodal models (Li et al., 2023)—e.g., foundation models trained on language and vision (images/video) data (Luo et al., 2020; Kim et al., 2021; Cho et al., 2021; Ramesh et al., 2021; Radford et al., 2021), and models trained on language and audio (Chen et al., 2022b; Wang et al., 2020; Baevski et al., 2020). Language models have also become specialized in specific domains like healthcare, biochemistry, climate, and finance (Zhao et al., 2023). Nearly 40% of surveyed enterprises are already considering building enterprise-specific language models.¹ Language models are also being used for interaction and control in robotics and smart devices (Huang et al., 2022; Driess et al., 2023).

Such multi-modal and embodied models are going to interact with a heterogeneous set of external systems, tools, and data—personal documents and multimedia data, cloud data, emails and social networks, external APIs, existing software stacks, and other expert language model agents. Eventually, models may be tailor-made and adapted to specific individuals, organizations, and governing bodies (Salemi et al., 2023). They will potentially be able to support short- and long-term personalized memory, participate in long-form

¹ <https://www.forbes.com/sites/forbestechcouncil/2023/07/20/the-power-of-domain-specific-llms-in-generative-ai-for-enterprises/?sh=c6fe4761e50f>

dialogue, and control gadgets and tools that are at our disposal every day. Taken together, future models will likely be *interactive agents* that can seamlessly communicate with humans, external systems (tools), other agents (models), and the physical world, to achieve increasingly complex human tasks. This changing landscape brings the following challenges into focus, that I hope to address in my future research.

Novel technical challenges Interactive multimodal (and embodied) LLM agents require technical research progress on several fronts. In particular, the following challenges will be especially pertinent to making progress:

- **Long-trajectory planning:** Given a complex user prompt/query, LLM agents will have to plan the accurate sequence of actions (interactions with external tools/models) and the exact content of the interaction. This is not dissimilar to planning in traditional reinforcement settings/POMDP RL agents. While existing methods can plan for complex user tasks for up to 3 interactions with external systems, more complex scenarios require considerable research in a) acquiring training data for planning; b) reward models that include final task completion signal as well as intermediate rewards from external systems and sub-task completion signals; c) robust evaluation of planning in heterogeneous scenarios. In general, planning is a complex problem that may have limited transfer from language model pre-training and human alignment. However, it may benefit from agent-human interaction data collected at scale, where an agent is actively addressing millions of real user queries in an ecosystem where it has multiple external tools/models at its disposal. The reward model could especially benefit from multi-agent intermediate feedback. As a motivating example, a user request for “research paper recommendations” may require the agent to interact with tools like Google Scholar and Huggingface Hub. This agent could potentially leverage reward signals in the form of open-sourced code links in papers and the number of downloads for released models in Huggingface Hub to improve its response quality to the user.
- **Stateful models and tools:** Personalization (tailoring model behavior to specific individuals or organizations) involves adapting models to specific domains, private data, rare entities, and tokens, as well as adapting to specific tasks of the user/organization. Domain-specific LLMs are a first step towards personalized models since they involve an offline adaptation framework. However, personalized models should be capable of online adaptation *during* interactions with users, updating knowledge, and learning new skills. This online update potentially requires two layers of model memory: a non-parametric

memory that indexes and stores interaction data with users and external systems, and a permanent memory (i.e. the model parameters) that gets updated less regularly. The interplay between short- and long-term memories leads models to be *stateful*. Unlike a static model that is trained offline, a stateful model will have a dynamically updated *state* that changes over time. Stateful models will require developing robust techniques to effectively and efficiently store and retrieve from the short-term cache and to carefully update long-term parametric memory.

Novel challenges to reliability Interactive multimodal (and embodied) LLM agents also bring up new challenges to reliable deployment, again on multiple fronts. Specific problems in this space that excite me include:

- Fair personalization: Personalized models that are trained, fine-tuned, and aligned at scale are likely to exacerbate bias in the underlying training data. For example, an agent that is trained to personalize to individual dietary needs is likely to generate responses about food recipes that are largely from European cuisines. For robust personalization, adapting to different socio-cultural nuances is going to be important. Moreover, a dynamic evaluation of robust personalization will be necessary to mark progress toward fair personalization
- Incorporating user feedback for model improvement: The increasingly interactive abilities of the model can also be leveraged for interaction with humans, specifically to troubleshoot and debug model behavior. For example, factual inconsistencies/hallucinations in model responses could potentially be flagged by users while in conversation with the model agent. This natural language feedback could be incorporated to update the model for future interactions. In particular, user feedback could be stored in non-parametric short-term memory and used to change parameters of the policy and reward model of the LLM agent, or lazily update model parameters. The surgical update/removal of specific information from model parameters is still an ongoing area of research that will be required for this interactive debugging setup. Moreover, evaluating the quality and integrity of the feedback itself will be a challenge. Finally, evaluating whether user feedback has been adapted to a satisfactory degree, especially for nuanced user feedback, is also an outstanding challenge.
- Use of model-generated text in training and evaluating models: Recent work is advancing the use of model-generated data to potentially fine-tune and align models at scale. Since model-generated text is prone to hallucinations and factual inconsistencies, this is very likely to set up a negative feedback

loop that exacerbates the problem and can make the model less reliable to users. As the amount of model-generated data grows on the internet, attributing data to verifiable sources in order to filter out low-quality data will also become challenging. This is an outstanding problem in the near future, that will require us to fundamentally rethink data quality, factuality, and attribution.

In addition to the challenges highlighted above, LLM agents are also going to have a significant societal impact, which brings with it several emerging challenges—security from data poisoning and data leaks, the privacy of personal and confidential data, misuse of models to spread misinformation for nefarious purposes, the environmental impact of training and deploying these models, challenges around legality and economics of data and model development, and concerns about inequality, bias, and fairness towards protected categories. Addressing these challenges will be important to engendering trust in LLM agents.

Bibliography

Ekin Akyürek, Tolga Bolukbasi, Frederick Liu, Binbin Xiong, Ian Tenney, Jacob Andreas, and Kelvin Guu. 2022. Towards tracing knowledge in language models back to the training data. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2429–2446.

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. 2016. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*.

Simran Arora, Avanika Narayan, Mayee F Chen, Laurel J Orr, Neel Guha, Kush Bhatia, Ines Chami, Frederic Sala, and Christopher Ré. 2022. Ask me anything: A simple strategy for prompting language models. *arXiv preprint arXiv:2210.02441*.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460.

Hangbo Bao, Li Dong, and Furu Wei. 2021. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*.

Joost Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2963–2977.

Aharon Ben-Tal, Dick Den Hertog, Anja De Waegenaere, Bertrand Melenberg, and Gijs Rennen. 2013. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357.

- Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. 2022. Prompting is programming: A query language for large language models. *arXiv preprint arXiv:2212.06094*.
- Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.
- Bernd Bohnet, Vinh Q Tran, Pat Verga, Roei Aharoni, Daniel Andor, Livio Baldini Soares, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, et al. 2022. Attributed question answering: Evaluation and modeling for attributed large language models. *arXiv preprint arXiv:2212.08037*.
- Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.
- Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.
- Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.
- Jerome S Bruner. 1961. The act of discovery. *Harvard educational review*.
- Oana-Maria Camburu, Eleonora Giunchiglia, Jakob Foerster, Thomas Lukasiewicz, and Phil Blunsom. 2019a. Can i trust the explainer? verifying post-hoc explanatory methods. *arXiv preprint arXiv:1910.02065*.
- Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, pages 9539–9549.
- Oana-Maria Camburu, Brendan Shillingford, Pasquale Minervini, Thomas Lukasiewicz, and Phil Blunsom. 2019b. Make up your mind! adversarial generation of inconsistent natural language explanations. *arXiv preprint arXiv:1910.03065*.
- Oana-Maria Camburu, Brendan Shillingford, Pasquale Minervini, Thomas Lukasiewicz, and Phil Blunsom. 2020. Make up your mind! adversarial generation of inconsistent natural language explanations. In

- Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4157–4165, Online. Association for Computational Linguistics.
- Jifan Chen, Aniruddh Sriram, Eunsol Choi, and Greg Durrett. 2022a. Generating literal and implied subquestions to fact-check complex claims. *arXiv preprint arXiv:2205.06938*.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.
- Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. 2022b. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518.
- Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. 2022c. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.
- I Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, Pengfei Liu, et al. 2023. Factool: Factuality detection in generative ai—a tool augmented framework for multi-task and multi-domain scenarios. *arXiv preprint arXiv:2307.13528*.
- Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. 2021. Unifying vision-and-language tasks via text generation. In *International Conference on Machine Learning*, pages 1931–1942. PMLR.
- Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.
- Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

- Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. 2021. Environment inference for invariant learning. In *International Conference on Machine Learning*, pages 2189–2200. PMLR.
- Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. *Advances in neural information processing systems*, 28.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.
- Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. 2019. Eraser: A benchmark to evaluate rationalized nlp models. *arXiv preprint arXiv:1911.03429*.
- Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. *arXiv preprint arXiv:2005.05339*.
- Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. 2023. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*.
- Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive prompting for decomposing complex questions. *arXiv preprint arXiv:2212.04092*.
- John Duchi, Peter Glynn, and Hongseok Namkoong. 2016. Statistics of robust optimization: A generalized empirical likelihood approach. *arXiv preprint arXiv:1610.03425*.
- Jacob Eisenstein. 2022. Informativeness and invariance: Two perspectives on spurious correlations in natural language. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4326–4331.

- Sabri Eyuboglu, Maya Varma, Khaled Saab, Jean-Benoit Delbrouck, Christopher Lee-Messer, Jared Dunmon, James Zou, and Christopher Ré. 2022. Domino: Discovering systematic errors with cross-modal embeddings. *arXiv preprint arXiv:2203.14960*.
- Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2022. Pal: Program-aided language models. *arXiv preprint arXiv:2211.10435*.
- Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. *arXiv preprint arXiv:1908.07898*.
- Roberto Gozalo-Brizuela and Eduardo C Garrido-Merchan. 2023. Chatgpt is not all you need. a state of the art review of large generative ai models. *arXiv preprint arXiv:2301.04655*.
- Emil Julius Gumbel. 1948. *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office.
- Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112.
- Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. 2018. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pages 1929–1938. PMLR.
- Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.
- Germund Hesslow. 1988. The problem of causal selection. *Contemporary science and natural explanation: Commonsense conceptions of causality*, pages 11–32.
- Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

- Weihua Hu, Gang Niu, Issei Sato, and Masashi Sugiyama. 2018. Does distributionally robust supervised learning give robust classifiers? In *International Conference on Machine Learning*, pages 2029–2037. PMLR.
- Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. 2022. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*.
- Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.
- Sarthak Jain, Sarah Wiegrefe, Yuval Pinter, and Byron C Wallace. 2020. Learning to faithfully rationalize by construction. *arXiv preprint arXiv:2005.00115*.
- Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparametrization with gumble-softmax. In *International Conference on Learning Representations (ICLR 2017)*. OpenReview. net.
- Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.
- Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2020. Learning the difference that makes a difference with counterfactually augmented data. *International Conference on Learning Representations (ICLR)*.
- Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hananeh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. *arXiv preprint arXiv:2005.00700*.
- Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2022. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*.
- Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR.

- Durk P Kingma, Tim Salimans, and Max Welling. 2015. Variational dropout and the local reparameterization trick. In *Advances in neural information processing systems*, pages 2575–2583.
- Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanus Phillips, Irena Gao, et al. 2021. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR.
- Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2022. Internet-augmented dialogue generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8460–8478, Dublin, Ireland. Association for Computational Linguistics.
- Veronica Latcinnik and Jonathan Berant. 2020. Explaining question answering models through text generation. *arXiv preprint arXiv:2004.05569*.
- Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering. *arXiv preprint arXiv:2203.05115*.
- Eric Lehman, Jay DeYoung, Regina Barzilay, and Byron C Wallace. 2019. Inferring which medical treatments work from reports of clinical trials. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3705–3717.
- Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.
- Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*. Citeseer.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

- Chunyuan Li, Zhe Gan, Zhengyuan Yang, Jianwei Yang, Linjie Li, Lijuan Wang, and Jianfeng Gao. 2023. Multimodal foundation models: From specialists to general-purpose assistants. *arXiv preprint arXiv:2309.10020*, 1(2):2.
- Xiang Lisa Li and Jason Eisner. 2019. Specializing word embeddings (for parsing) by information bottleneck. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2744–2754.
- Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2019. CommonGen: A constrained text generation challenge for generative commonsense reasoning. *arXiv preprint arXiv:1911.03705*.
- Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2023a. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. *arXiv preprint arXiv:2302.07452*.
- Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, et al. 2023b. Ra-dit: Retrieval-augmented dual instruction tuning. *arXiv preprint arXiv:2310.01352*.
- Peter Lipton. 1990. Contrastive explanation. *Royal Institute of Philosophy Supplements*, 27:247–266.
- Alisa Liu, Swabha Swayamdipta, Noah A Smith, and Yejin Choi. 2022a. Wanli: Worker and ai collaboration for natural language inference dataset creation. *arXiv preprint arXiv:2201.05955*.
- Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. 2021. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR.
- Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022b. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *arXiv preprint arXiv:2205.05638*.
- Nelson F Liu, Tianyi Zhang, and Percy Liang. 2023. Evaluating verifiability in generative search engines. *arXiv preprint arXiv:2304.09848*.

- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Yixin Liu, Alexander R Fabbri, Pengfei Liu, Yilun Zhao, Linyong Nan, Ruilin Han, Simeng Han, Shafiq Joty, Chien-Sheng Wu, Caiming Xiong, et al. 2022c. Revisiting the gold standard: Grounding summarization evaluation with robust human evaluation. *arXiv preprint arXiv:2212.07981*.
- Charles Lovering, Rohan Jha, Tal Linzen, and Ellie Pavlick. 2020. Predicting inductive biases of pre-trained models. In *International Conference on learning representations*.
- Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.
- Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Jason Li, Taroon Bharti, and Ming Zhou. 2020. Univl: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*.
- Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.
- Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.
- Chaitanya Malaviya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. 2023. Expertqa: Expert-curated questions and attributed answers. *arXiv preprint arXiv:2309.07852*.
- Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Hannaneh Hajishirzi, and Daniel Khashabi. 2022. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*, 7.
- Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. 2023. Augmented language models: a survey. *arXiv preprint arXiv:2302.07842*.
- Tim Miller. 2018. Contrastive explanation: A structural-model approach. *arXiv preprint arXiv:1811.03163*.

- Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38.
- Sewon Min, Kalpesh Krishna, Xinxu Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*.
- Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2021. Cross-task generalization via natural language crowdsourcing instructions. *arXiv preprint arXiv:2104.08773*.
- Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.
- Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. 2020. Learning from failure: De-biasing classifier from biased classifier. *Advances in Neural Information Processing Systems*, 33:20673–20684.
- Junhyun Nam, Jaehyung Kim, Jaeho Lee, and Jinwoo Shin. 2022. Spread spurious attribute: Improving worst-group accuracy with spurious attribute estimation. *arXiv preprint arXiv:2204.02070*.
- Sharan Narang, Colin Raffel, Katherine Lee, Adam Roberts, Noah Fiedel, and Karishma Malkan. 2020. Wt5?! training text-to-text models to explain their predictions. *arXiv preprint arXiv:2004.14546*.
- OpenAI. 2023. Gpt-4 technical report.
- Yonatan Oren, Shiori Sagawa, Tatsunori B Hashimoto, and Percy Liang. 2019. Distributionally robust language modeling. *arXiv preprint arXiv:1909.02060*.
- Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.
- Bhargavi Paranjape, Pradeep Dasigi, Vivek Srikumar, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022. Agro: Adversarial discovery of error-prone groups for robust optimization. *arXiv preprint arXiv:2212.00921*.

- Bhargavi Paranjape, Mandar Joshi, John Thickstun, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. An information bottleneck approach for controlling conciseness in rationale extraction. *arXiv preprint arXiv:2005.00652*.
- Bhargavi Paranjape, Matthew Lamm, and Ian Tenney. 2021a. Retrieval-guided counterfactual generation for qa. *arXiv preprint arXiv:2110.07596*.
- Bhargavi Paranjape, Scott Lundberg, Sameer Singh, Hannaneh Hajishirzi, Luke Zettlemoyer, and Marco Tulio Ribeiro. 2023. Art: Automatic multi-step reasoning and tool-use for large language models. *arXiv preprint arXiv:2303.09014*.
- Bhargavi Paranjape, Julian Michael, Marjan Ghazvininejad, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021b. Prompting contrastive explanations for commonsense reasoning tasks. *arXiv preprint arXiv:2106.06823*.
- Aaron Parisi, Yao Zhao, and Noah Fiedel. 2022. Talm: Tool augmented language models. *arXiv preprint arXiv:2205.12255*.
- Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.
- Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. 2023. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.
- Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2020. How context affects language models’ factual predictions. *arXiv preprint arXiv:2005.04611*.

- Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.
- Gregory Plumb, Marco Tulio Ribeiro, and Ameet Talwalkar. 2021. Finding and fixing spurious patterns with explanations. *arXiv preprint arXiv:2106.02112*.
- Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. *arXiv preprint arXiv:1805.01042*.
- Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*.
- Danish Pruthi, Rachit Bansal, Bhuwan Dhingra, Livio Baldini Soares, Michael Collins, Zachary C. Lipton, Graham Neubig, and William W. Cohen. 2022. Evaluating explanations: How much do explanations from the teacher aid students? *Transactions of the Association for Computational Linguistics*, 10:359–375.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *Technical Report, OpenAI*.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.
- Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942.
- Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR.

- Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, Lora Aroyo, Michael Collins, Dipanjan Das, Slav Petrov, Gaurav Singh Tomar, Iulia Turc, and David Reitter. 2023. Measuring attribution in natural language generation models. *Computational Linguistics*, pages 1–64.
- Marco Tulio Ribeiro and Scott Lundberg. 2022. Adaptive testing and debugging of NLP models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3253–3267, Dublin, Ireland. Association for Computational Linguistics.
- Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.
- Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.
- Kyle Richardson and Ashish Sabharwal. 2020. What does my qa model know? devising controlled probes using expert knowledge. *Transactions of the Association for Computational Linguistics*, 8:572–588.
- Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*.
- Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2019. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*.
- Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2023. Lamp: When large language models meet personalization. *arXiv preprint arXiv:2304.11406*.
- Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.
- Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

- Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.
- Sofia Serrano and Noah A Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951.
- Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.
- Kurt Shuster, Mojtaba Komeili, Leonard Adolphs, Stephen Roller, Arthur Szlam, and Jason Weston. 2022. Language models that seek for knowledge: Modular search & generation for dialogue and prompt completion. *arXiv preprint arXiv:2203.13224*.
- Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Unsupervised commonsense question answering with self-talk. *arXiv preprint arXiv:2004.05483*.
- Nimit Sohoni, Jared Dunnmon, Geoffrey Angus, Albert Gu, and Christopher Ré. 2020. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *Advances in Neural Information Processing Systems*, 33:19339–19352.
- Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.
- Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.
- Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. olympics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.
- Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

- Naftali Tishby, Fernando C. Pereira, and William Bialek. 1999. The information bottleneck method. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
- Trieu H Trinh and Quoc V Le. 2018. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Sravya Popuri, Dmytro Okhonko, and Juan Pino. 2020. Fairseq s2t: Fast speech-to-text modeling with fairseq. *arXiv preprint arXiv:2010.05171*.
- Cunxiang Wang, Xiaoze Liu, Yuanhao Yue, Xiangru Tang, Tianhang Zhang, Cheng Jiayang, Yunzhi Yao, Wenyang Gao, Xuming Hu, Zehan Qi, et al. 2023. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *arXiv preprint arXiv:2310.07521*.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.
- Yang Wang. 2021. Recognizing reduplicated forms: Finite-state buffered machines. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 177–187, Online. Association for Computational Linguistics.
- Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, Ed Chi, and Slav Petrov. 2020. Measuring and reducing gendered correlations in pre-trained models. *arXiv preprint arXiv:2010.06032*.
- Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

- Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed H Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.
- Daniel S Weld and Gagan Bansal. 2019. The challenge of crafting intelligible intelligence. *Communications of the ACM*, 62(6):70–79.
- Peter West, Ari Holtzman, Jan Buys, and Yejin Choi. 2019. Bottlesum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3743–3752.
- Sarah Wiegreffe, Ana Marasovic, and Noah A Smith. 2020. Measuring association between labels and free-text rationales. *arXiv preprint arXiv:2010.12762*.
- Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.
- Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.
- Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. 2022a. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pages 23965–23998. PMLR.

- Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. 2022b. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7959–7971.
- Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S Weld. 2019. Errudite: Scalable, reproducible, and testable error analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 747–763.
- Yuxiang Wu, Matt Gardner, Pontus Stenetorp, and Pradeep Dasigi. 2022. Generating data to mitigate spurious correlations in natural language inference datasets. *arXiv preprint arXiv:2203.12942*.
- Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2023. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*.
- Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*.
- Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020. G-daug: Generative data augmentation for commonsense reasoning. *arXiv preprint arXiv:2004.11546*.
- Xi Ye, Rohan Nair, and Greg Durrett. 2021. Evaluating explanations for reading comprehension with realistic counterfactuals. *arXiv preprint arXiv:2104.04515*.
- Xiang Yue, Boshi Wang, Kai Zhang, Ziru Chen, Yu Su, and Huan Sun. 2023. Automatic evaluation of attribution by large language models. *arXiv preprint arXiv:2305.06311*.
- Hongming Zhang, Xinran Zhao, and Yangqiu Song. 2020. Winowhy: A deep diagnosis of essential commonsense knowledge for answering winograd schema challenge. *arXiv preprint arXiv:2005.05763*.
- Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A Smith. 2023. How language model hallucinations can snowball. *arXiv preprint arXiv:2305.13534*.

Xujiang Zhao, Jiaying Lu, Chengyuan Deng, Can Zheng, Junxiang Wang, Tanmoy Chowdhury, Li Yun, Hejie Cui, Zhang Xuchao, Tianjiao Zhao, et al. 2023. Domain specialization as the key to make large language models disruptive: A comprehensive survey. *arXiv preprint arXiv:2305.18703*.

Andrey Zhmoginov, Ian Fischer, and Mark Sandler. 2019. Information-bottleneck approach to salient region discovery. *arXiv preprint arXiv:1907.09578*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinu Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2023. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Chapter A

Appendix

| Prompt Pattern | Commonsense Example & Model Generated Explanation |
|---|---|
| Personal Characteristics ⇒ <i>P</i> likes/likes to _ while <i>Q</i> likes/likes to _ <i>P</i> likes/likes to _ while <i>Q</i> does not like/like to _ <i>P</i> prefers/prefers to _ while <i>Q</i> prefers _ <i>Q</i> prefers _ while <i>P</i> does not prefer/prefer to _ <i>Q</i> thinks _ while <i>P</i> thinks/does not think _ | Megan said it would be liberating to go out without makeup like Elena does since __ never wore makeup Explanation: Elena likes to <u>be natural</u> while Megan likes to <u>wear lipstick</u> |
| Object Characteristics <i>P</i> is taller/shorter/smaller/larger/slower/faster than <i>Q</i> ⇒ <i>P</i> is/are _ while/but/however <i>Q</i> is/are _ <i>Q</i> has/have _ while/but/however <i>P</i> has/have _ <i>P</i> has/have more/less _ than <i>Q</i> <i>P</i> is/are _ than <i>Q</i> | How to tie pieces of paper together? __ (a) Thread ruler through the holes (b) Thread ribbon through the holes Explanation: Ruler is <u>hard</u> while a ribbon is <u>flexible</u> |
| Spatial/Temporal Contrast ⇒ <i>P</i> is inside/outside/above/below <i>Q</i> _ is closer to <i>P</i> and farther away from <i>Q</i> <i>P</i> is to the right/left of <i>Q</i> <i>Q</i> takes longer to _ than <i>P</i> <i>Q</i> is closer to _ while <i>P</i> is closer to _ <i>Q</i> happened before/after <i>P</i> | Emily looked up and saw Patricia racing by overhead. __ was on the ramp. Explanation: Emily is below Patricia |
| Use cases and causes <i>P</i> is used for _ <i>Q</i> <i>P</i> is used to do <i>Q</i> _ ⇒ <i>P</i> is used for/to/in _ while <i>Q</i> is used for/to/in _ <i>Q</i> is used _ while <i>P</i> is used _ <i>Q</i> because _ while <i>P</i> because _ <i>Q</i> can cause _ while <i>P</i> results in _ | To prepare the puff pastry for your pie, line a baking sheet with parchment. Then __ (a) Unroll the pastry, lay it over baking twine . (b) Unroll the pastry, lay it over fishing line . Explanation: Baking twine is used in <u>baking</u> while fishing line is used in <u>fishing</u> |

Table A.1: Contrastive Patterns and Examples of outputs generated by the T5-large model. The pattern the PLM completes are marked ⇒ .

String Operations

In these examples, you are given a task description and an input. Break the input down into subtasks in order to solve the task. You can use string operations like splitting, reformatting, editing or merging. You can also use other operations like arithmetic and logic.

Description: (Date Understanding) Find the required date in MM/DD/YYYY using information about related events and dates in the input. Clue: First find what day is today.

Input: The deadline is Jun 1, 2021, which is 2 days away from now. What is the date 24 hours later in MM/DD/YYYY?

Q1: [string reformat] Jun 1, 2021 in MM/DD/YYYY

#1: 06/01/2021

Q2: [arithmetic] 06/01/2021 is 2 days away from now. What date is today?

#2: Today is 04/01/2021

Q3: [arithmetic] What date is 24 hours later than today?

#3: 05/01/2021

Q4: [EOQ]

Ans: 05/31/2021

—

Description: (Language games) Translate English into Pig Latin.

Input: (English) Sami made his way across the bar and hugged Layla.

Q1: [string split] What are the words in "Sami made his way across the bar and hugged Layla."?

#1: ["Sami", "made", "his", "way", "across", "the", "bar", "and", "hugged", "Layla", "."]

Q2: [string edit] Transfer the initial consonant of each word to the end of the word and adding "ay" after it.

#2: ["Amisay", "ademay", "ishay", "ayway", "acrossyay", "ethay", "arbay", "andyay", "uggedhay", "Aylalay", "."]

Q3: [string merge] Concatenate #2 into a full sentence.

#3: Amisay ademay ishay ayway acrossyay ethay arbay andyay ugedhay Aylalay.

Q4: [EOQ]

Ans: Amisay ademay ishay ayway acrossyay ethay arbay andyay ugedhay Aylalay.

—

Description: (Kth letter concatenation) Take the letters at position 3 of the words in a list of words and concatenate them using a space.

Input: Take the letters at position 3 of the words in "Savita Saeed Ramos Sato Yadav" and concatenate them using a space.

Q1: [string split] What are the words in "Savita Saeed Ramos Sato Yadav"?

#1: ["Savita", "Saeed", "Ramos", "Sato", "Yadav"]

Q2: [string index] What is the third letter of words in the list in #1?

#2: ["v", "e", "m", "t", "d"]

Q3: [string merge] Concatenate #2 with spaces

#3: "v e m t d"

Q4: [EOQ]

Ans: v e m t d

—

Descripton: %s

Input: %s

Q1:

Table A.2: Programs in the task library for tasks requiring string manipulation.

 Arithmetic

In these examples, you are given a task description and an input. Break the input down into subtasks in order to solve the task. You can generate python code to solve arithmetic and algebra equations in using functions from sympy.

```
from sympy import Symbol
from sympy import simplify
import math
from sympy import solve,t
# solve,t(equations, variable) : solvingtheequationsandreturnthevariablevalue.
```

Description: (Aqua-rat) Solve the following arithmetic problems on ratios and fractions, writing out intermediate arithmetic calculations as python code. Store your result as a variable named 'ans'.

Input: In a flight of 600 km, an aircraft was slowed down due to bad weather. Its average speed for the trip was reduced by 200 km/hr and the time of flight increased by 30 minutes. The duration of the flight is: A)1 hour B)2 hours C)3 hours D)4 hours E)5 hours

Q1: [generate python code] write python code to solve the problem, using math and sympy.

```
#1:
duration = Symbol('duration', positive=True)
delay = 30 / 60
total_distance = 600
original_speed = total_distance / duration
reduced_speed = total_distance / (duration + delay)
solution = solve,t(original_speed - reduced_speed - 200, duration)
ans = solution[duration]
print(ans)
```

Q2: [code execute] Execute the python code in #1 and get the value of "ans"

#2:

1.0

Q3: [compare] Which of the options among A)1 hour B)2 hours C)3 hours D)4 hours E)5 hours is most similar to the answer?

#3: A

Q4: [EOQ]

Ans: A

Description: (Elementary Math) Solve the following middle-school arithmetic problems, writing out intermediate arithmetic calculations as python code. Store your result as a variable named 'ans'.

Input: Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for \$2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Q1: [generate python code] write down the arithmetic or algebra equations as python code, storing the answer as 'ans'

```
#1:
total_eggs = 16
eaten_eggs = 3
baked_eggs = 4
sold_eggs = total_eggs - eaten_eggs - baked_eggs
dollars_per_egg = 2
ans = sold_eggs * dollars_per_egg
print(ans)
```

Q2: [code execute] Execute the python code in #1 and get the value of "ans"

#2: 18

Q3: [EOQ]

Ans: 18

Description: (Grage school Math) Solve the following middle-school arithmetic problems, writing out intermediate arithmetic calculations as python code. Store your result as a variable named 'ans'.

Input: Joseph and Getty went to buy ice creams, they together bought 36 ice creams. On the way back, Joseph ate 12 of the ice cream, and he has 2 ice creams left now.

Q1: [generate python code] write down the arithmetic or algebra equations as python code, storing the answer as 'ans'

```
#1:
num_ice_creams_bought_by_joseph = 2 + 12
total_ice_creams = 36
ans = total_ice_creams - num_ice_creams_bought_by_joseph
print(ans)
```

Q2: [code execute] Execute the python code in #1 and get the value of "ans"

#2: 22

Q3: [EOQ]

Ans: 22

Descripton: %s

Input: %s

Q1:

Table A.3: Programs in the task library for tasks requiring arithmetic operations.

 Search

In these examples, you are given a task description and an input. Break the input down into subtasks in order to solve the task. You can use search functions like Google search in one or more of your substeps, if there is insufficient information. Other functions like arithmetic and logical operations can also be used.

Description: (Known or Unknown) Choose the option that best answers the question. If the question does not have a known answer, choose "Unknown".

Input: How many hairs were on Neil Armstrong's head when he landed on the moon?

choice: Unknown

choice: Five million

Q1: [search] How many hairs were on Neil Armstrong's head when he landed on the moon?

#1:

Apollo 11 (July 16–24, 1969) was the American spaceflight that first landed humans on the Moon. Commander Neil Armstrong and lunar module pilot Buzz Aldrin.

Neil Alden Armstrong (August 5, 1930 – August 25, 2012) was an American astronaut and aeronautical engineer who became the first person to walk on the Moon.

Q2: [subquestion] Does the information help answer the question? There could be no definitive answer because the question is too specific, about personal details not in public record, because the answer is not yet known, or the question is opinion-based.

#2: No. The question is too specific

Q3: [compare] What is the final answer?

#3: Unknown

Q4: [EOQ]

Ans: Unknown

Description: (Anachronisms) An anachronism is a mistake in chronology, or a person, thing, or event that is out of its proper time. Does the sentence contain an anachronism? Answer Yes/No.

Input: President George H. W. Bush called his generals to the Oval Office at the outset of the Gulf War.

Q1: [tag] What are the entities in this sentence?

#1:

President George H. W. Bush

Gulf War

Q2: [search] When was President George H. W. Bush president?

#2: George H. W. Bush's tenure as the 41st president of the United States began with his inauguration on January 20, 1989, and ended on January 20, 1993.

Q3: [search] When was the Gulf War fought?

#3: The Gulf War[b] was a 1990–1991 armed campaign waged by a 35-country military coalition in response to the Iraqi invasion of Kuwait.

#4: [subquestion] Could these entities have co-existed based on their time periods alone?

Yes. Their time periods intersect.

Q5: [generate output] Is this an anachronism?

#5: No

Q6: [EOQ]

Ans: No

Description: (Hindu Knowledge) Answer questions about Hindu mythology by choosing the option that best answers the question.

Input: In the Mahabharata, Karna is cursed to forget the incantations needed to use which weapon?

choice: Anjalikastra

choice: Narayanastra

choice: Agneyastra

choice: Brahmastra

Q1: [search] In the Mahabharata, Karna is cursed to forget the incantations needed to use which weapon?

#1: As a result, he cursed Karna, saying that HIS MARTIAL SKILLS, including the use of BRAHMASTRA, would abandon him when he needed them most. Indra, the King of Gods, stung Karna in the form of a bee to get him cursed by Parshuram. Karna walked through the woods in despair, feeling dejected by the curse. A skilled & devoted warrior...

Q2: [compare] Which option is the answer in #3 most similar to?

#2: Brahmastra

Q3: [EOQ]

Ans: Brahmastra

Descripton: %s

Input: %s

Q1:

Table A.4: Programs in the task library for tasks requiring search for world knowledge.

Code Generation and Editing

In these examples, you are given a task description and an input. Break the input down into subtasks in order to solve the task. You can use a python code generation and execution function in one or more of your substeps, if required. Other functions like arithmetic and logical operations can also be used.

Description: (Auto Debugging) Debug the following code snippets by finding the answer or the error message.

Input:

```
...
if x < 5:
    pass
...
```

What error does this program surface?

Q1: [execute] Execute the following python code snippet.

```
if x < 5:
    pass
#1:
```

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

NameError: name 'x' is not defined

Q2: [generate answer] What is the final error message?

#2: NameError: name 'x' is not defined

Q3: [EOQ]

Ans: NameError: name 'x' is not defined

Description: (Code description) Given a python code snippet, choose the option that is the best description of the code snippet.

Input:

Python code:

try:

```
    n = int(input())
    m = int(input())
    integer_sum = int(n) + int(m)
    print(integer_sum)
```

except:

```
    print('error')
```

choice: prints number between 5 and 6

choice: try input and except error

choice: inputs the string 'try'

choice: prints sum of two input numbers only if they are integers otherwise raises error

Q1: [code generate] prints number between 5 and 6

#1:

```
import random
print(random.uniform(5,6))
```

Q2: [code generate] try input and except error

#2:

try:

```
    file = open(file_name, "r")
    data = file.read()
    file.close()
```

except:

```
    print('error')
```

Q3: [code generate] inputs the string 'try'

#3: print('try')

Q4: [code generate] prints sum of two input numbers only if they are integers otherwise raises error

#4:

```
#!/usr/bin/python
```

```
a=raw_input("enter first number : ")
```

```
b=raw_input("enter second number : ")
```

try:

```
    sum=int(a)+int(b)
    print "sum is: ",sum
```

except:

```
    print "enter integer values only"
```

Q5: [compare] Which of the generated code snippets are most like the original one?

#5: prints sum of two input numbers only if they are integers otherwise raises error

Q6: [EOQ]

Ans: prints sum of two input numbers only if they are integers otherwise raises error

Description: %s

Input: %s

Q1:

Table A.5: Programs in the task library for tasks requiring code generation or editing.

 CoT Reasoning

In these examples, you are given a task description and an input. Break the input down into subtasks in order to solve the task. Thinking though the problem explicitly can be one of the substeps you use.

Description: (Sports Understanding) Determine whether an artificially constructed sentence relating to sports is plausible. The final answer should be "yes" or "no".

Input: Is the following sentence plausible? "Santi Cazorla scored a touchdown."

Q1: [think step-by-step]

#1: Let's think step-by-step. Santi Cazorla is a soccer player. Touchdown is part of American football and rugby. So the answer is no.

Q2: [EOQ]

Ans: no

Description: (Hyperbation) Identify correct adjective ordering from the two choices. This involves selecting what would be considered the more inexplicably "intuitive" sentence by a native English speaker.

Input: Which sentence has the correct adjective order:

Options:

(A) repulsive small Brazilian exercise ship

(B) Brazilian repulsive exercise small ship

Q1: [think step-by-step]

#1: Let's think step-by-step. When there is more than one adjective before a noun, the adjectives need to respect the following order before a noun: "[1. opinion] [2. size] [3. age] [4. shape] [5. color] [6. origin] [7. material] [8. purpose] noun".

Option (A): "repulsive small Brazilian exercise ship". (1) "repulsive" falls into the opinion category. (2) "small" falls into the size category. (3) "Brazilian" falls into the origin category. (4) "exercise" falls into the purpose category. Option (A) has the following adjective order: [1. opinion] [2. size] [6. origin] [8. purpose] (or, in numeric terms, 1 2 6 8). Because $1 < 2 < 6 < 8$ is correct, (A) has the correct ordering.

Option (B): "Brazilian repulsive exercise small ship". Option (B) has the following adjective order: [6. origin] [1. opinion] [8. purpose] [2. size] (or, in numeric terms, 6 1 8 2). Because $6 < 1 < 8 < 2$ is not correct, (B) does not have the correct ordering. So the answer is (A).

Q2: [EOQ]

Ans: (A)

Description: (Formal Fallacies) Distinguish deductively valid syllogistic arguments from formal fallacies, paying specific attention to negations.

Input: "It is not always easy to see who is related to whom – and in which ways. The following argument pertains to this question: To begin with, Lesley is a close friend of Fernando. Moreover, being a close friend of Fernando or a schoolmate of Lowell is sufficient for being a great-grandfather of Leroy. It follows that Lesley is a great-grandfather of Leroy."

Is the argument, given the explicitly stated premises, deductively valid or invalid?

Options:

- valid

- invalid

Q1: [think step-by-step]

#1:

Let's think step-by-step.

(1) Lesley is a close friend of Fernando: Lesley = friend(Fernando).

(2) Being a close friend of Fernando or a schoolmate of Lowell is sufficient for being a great-grandfather of Leroy: If $X = \text{friend}(\text{Fernando})$ OR $\text{SCHOOLMATE}(\text{Lowell})$, then $X = \text{great-grandfather}(\text{Leroy})$.

Hypothesis: Does it follow that Lesley is a great-grandfather of Leroy: Lesley = great-grandfather(Leroy)?

Let's see whether the Hypothesis can be deduced from the arguments (1) and (2) by logical reasoning?

By (1), we have Lesley = friend(Fernando). By (2), we have if Lesley = friend(Fernando), then Lesley = great-grandfather(Leroy).

So, it is true that Lesley is a great-grandfather of Leroy. So the answer is valid.

Q2: [EOQ]

Ans: valid

Description: (Reasoning about colored objects) Given a collection of colored objects in the text input, answer the question at the end of the input.

Input: On the nightstand, there is a red pencil, a purple mug, a burgundy keychain, a fuchsia teddy bear, a black plate, and a blue stress ball. What color is the stress ball?

Q1: [think step-by-step]

#1: Let's think step-by-step. According to this question, the color of the stress ball is blue. So the answer is blue.

Q2: [EOQ]

Ans: blue

Descripton: %s

Input: %s

Q1: ""

Table A.6: Programs in the task library for tasks requiring free-form chain-of-thought style reasoning about logic and linguistics.