Effective Model Deployment and Data Curation for Foundation Model Development

Cheng-Yu Hsieh

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington
2025

Reading Committee:
Ranjay Krishna, Co-Chair
Alexander J. Ratner, Co-Chair
Hannaneh Hajishirzi
Noah A. Smith

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Effective Model Deployment and Data Curation for
Foundation Model Development

Cheng-Yu Hsieh

Co-chairs of the Supervisory Committee:

Assistant Professor Ranjay Krishna
Computer Science and Engineering

Affiliate Assistant Professor Alexander J. Ratner
Computer Science and Engineering

While scaling—both in terms of model size and dataset volume—has driven many of the recent breakthroughs in AI, this increasingly larger-scale development trajectory faces emergent challenges not seen in traditional small-scale supervised learning setting. On model side, the exponential growth in parameter counts has rendered these highly capable but massive models prohibitively expensive to deploy or adapt for many practical applications. On the data side, although training on massive datasets improves performance on standard benchmarks, scaling alone does not guarantee the emergence of desirable model behaviors beyond traditional metrics. This thesis develops techniques to address core challenges along both the model and data axes of modern AI development. Specifically, it proposes strategies for the efficient deployment and adaptation of Transformer-based large language models, and introduces principled methods for curating reliable and effective data to evaluate and improve modern vision-language models beyond standard accuracy metrics. Collectively, these contributions aim to make large-scale AI systems more effective and accessible across diverse real-world scenarios.

Acknowledgements

The past five years throughout my PhD journey have been an incredibly enjoyable and truly once-in-a-lifetime experience. I feel deeply grateful to have shared this journey with so many inspiring and supportive people. This thesis would not have been possible without the guidance, encouragement, and companionship of my advisors, mentors, collaborators, friends, and family.

I have been very fortunate and will always be grateful to my very inspiring and genuinely caring advisors, Ranjay and Alex, who have always been by my side throughout this journey. I still remember my very first meeting with Ranjay over a video call, sharing with him some early research I had been working on. His insights into the problem were spot-on and helped shape my perspective and our research direction in a meaningful way. From then on, I have continued to learn from his clarity of thought and his dedication to doing impactful work. I have learned more from Ranjay than I could ever fully list—from his unique views on problems, the always creative solutions, the rigor in experimentation, to the clarity of presenting compelling research. Importantly, beyond research, Ranjay has been a life mentor to me. He has been there through my highs and lows, always showing deep care for my well-being. His life wisdom has guided and supported both my personal and professional growth. It is my great fortune and pleasure to work alongside him, seeing the lab grow from the very beginning to what it is today. Thank you, Ranjay—my PhD would not have been such a fulfilling journey without you. Alex is one of the biggest reasons I came to UW. He believed in me even when I had little experience, and took me in as a fresh PhD student. I have always admired Alex's sharp, practical perspective on research problems. I have learned to dare to explore bold research ideas, to execute with careful experiments, and to handle research challenges with resilience. I have also learned a great deal on how to present research through our back-and-forth, multi-version paper drafts. I am grateful for Alex's continuous support and warm guidance, both in my career and personal life. Thank you, Alex.

I would like to thank my PhD committee—Hanna, Noah, and Aylin—for their time, thoughtful feedback, and generous support throughout the process of working on this thesis. Thank you for your insightful questions, suggestions, and for pushing me to think deeply about my work. Our discussions have helped shape both this thesis and the way I view future directions to work on. I am truly grateful for your guidance and encouragement.

One of the most rewarding parts of my PhD has been the opportunity to learn from amazing mentors during my industry internships. Chen-Yu and Chun-Liang supported me wholeheartedly during my very first internship at Google, giving me both trust and freedom to explore research ideas, and working hands-on with me to tackle challenges and shaping the work. The work we did together was transformative to me and it had a lasting impact on the direction of my research. During my time at Apple, I enjoyed the deep and thoughtful discussions with Hadi, Pavan, and Oncel. They taught me to be attentive to small yet crucial details, and how to iterate fast with experiments at scale. It is my great pleasure and an exciting opportunity to be joining the team and continue to work together.

My research journey would not have been the same and fulfilling without the amazing collaborators I have been fortunate to work with and learn from. Thank you Chih-Kuan, Zifeng, Long, Abhishek, Si-An, Tomas, Ani, Ravi, Fartash, Yu-Guan, Yung-Sung, Jieyu, James, Zixian, Amita, Mahtab, Scott, Jeffrey, Abhinav, Shiwei, Peter, Ayana, Madeline, and more. My PhD life would not have been as joyful without the company with my amazing labmates and friends—Esteban, Jiafei, Ainaz, Matt, Thao, Wei-Chiu, Kuo-Hao, Xiang, Benlin, Chenhao, George, Lindsey, and more.

Last but not least, I would like to thank my family for the endless love and support they have always shown me. They have stood by me through every step of this journey, and throughout my life, cheering me on in challenging moments and giving me the strength to follow my passion and pursue my goals. Especially to my partner, Evie—thank you for always believing in me and being there with me through all the highs and lows. Knowing you are there gives me the strength and comfort to keep chasing what I believe in. I am lucky to have your constant love, your support, and the quiet strength you always give. I am grateful to have shared this journey with you, and to walk all the ones still to come.

Contents

1 Introduction **23**

1.1 Effective model deployment and adaptation with large language models 24

1.1.1 Compute-efficient model deployment 24

1.1.2 Cost-effective model adaptation 25

1.2 Effective data curation for measuring and improving compositionality in vision-language models 25

1.2.1 Reliable data curation for vision-language model evaluation 25

1.2.2 Effective data curation for improving vision-language models 26

2 Efficient Language Model Deployment: Training Smaller Models with Less Data **27**

2.1 Introduction 27

2.2 Related work 29

2.3 Distilling step-by-step 30

2.3.1 Extracting rationales from LLMs 31

2.3.2 Training smaller models with rationales 32

2.4 Experiments 34

2.4.1 Reducing training data 35

2.4.2 Reducing model size 36

2.4.3 Outperforming LLMs using minimum model size and least training data 39

2.4.4 Further ablation studies 40

2.5 Discussion 42

2.6 Limitations . . . 42
2.7 Ethics statement . . . 43

3 Effective Language Model Adaptation: Calibrating Positional Attention Bias Improves Long Context Utilization 45

3.1 Introduction . . . 45
3.2 Positional attention bias overpowers mid-sequence context . . . 47
3.2.1 U-shaped attention bias . . . 48
3.2.2 Does attention favor relevant context? . . . 50
3.3 Found-in-the-middle: modeling and isolating positional attention bias . . . 51
3.3.1 Two main factors in model attention . . . 52
3.3.2 Disentangling positional attention bias . . . 53
3.4 Improving long-context utilization with found-in-the-middle . . . 55
3.4.1 Attention calibration . . . 55
3.4.2 Calibrated v.s. uncalibrated attention . . . 56
3.4.3 Attention calibration in practice . . . 57
3.5 Related work . . . 58
3.6 Discussion . . . 60
3.7 Limitations . . . 60
3.8 Ethical Statement . . . 61

4 Data for Reliable Vision-Language Model Development: Fixing Hackable Benchmarks for Vision-Language Compositionality 63

4.1 Introduction . . . 63
4.2 Related Work . . . 65
4.3 Limit and biases of current compositionality benchmarks . . . 66
4.4 SUGARCREPE . . . 68
4.4.1 SUGARCREPE generation workflow alleviates dataset biases . . . 68
4.4.2 SUGARCREPE covers a broad range of hard negative types . . . 70

4.5 Evaluations . . . 71
4.5.1 SUGARCREPE significantly reduces dataset biases . . . 72
4.5.2 Re-evaluating recent methods for improving compositionality . . . 74
4.5.3 Comprehensive evaluations on existing pretrained vision-language models . . . 75
4.6 Discussions . . . 77
4.6.1 Limitation and future work . . . 77
4.6.2 Societal impact . . . 79

5 Data for Improved Vision-Language Model: Instruction Tuning Enables Zero-Shot Conditional Image Representations 81

5.1 Introduction . . . 81
5.2 Related work . . . 83
5.3 Conditional embeddings via instruction contrastive tuning . . . 85
5.3.1 FocalLens with MLLMs . . . 86
5.3.2 FocalLens with CLIP . . . 86
5.4 Experiments . . . 87
5.4.1 Conditional representations better characterize task-specific details . . . 88
5.4.2 FocalLens improves image representations across benchmarks . . . 89
5.4.3 Comparative analysis of FocalLens variants . . . 93
5.4.4 FocalLens representations improve downstream applications . . . 93
5.5 Discussion . . . 96

6 Conclusion 99

A Appendix: Efficient Language Model Deployment 127

A.1 Experiment detail . . . 127
A.1.1 Implementation . . . 127
A.1.2 Datasets . . . 127

B Appendix: Effective Language Model Adaptation **129**

B.1 Multi-doc QA datasets . . . 129

B.2 Implementation details . . . 130

B.3 Additional experiment results . . . 130

B.4 Compute and inference details . . . 131

C Appendix: Data for Reliable Vision-Language Model Development **133**

C.1 Implementation details . . . 133

C.1.1 Hardware information . . . 133

C.1.2 Dataset sources . . . 133

C.1.3 Software configuration . . . 134

C.2 Vision-language compositionality benchmarks . . . 134

C.2.1 Image-to-text formulation . . . 134

C.2.2 Text-to-image formulation . . . 135

C.3 SUGARCREPE . . . 136

C.3.1 Taxonomy . . . 136

C.3.2 Hard negative generation procedure and templates . . . 137

C.3.3 Adversarial refinement . . . 138

C.3.4 Dataset construction cost . . . 138

C.3.5 Dataset information . . . 139

C.4 Detailed evaluation results . . . 139

C.4.1 Full evaluation results on existing benchmarks . . . 139

C.4.2 SUGARCREPE human evaluation . . . 140

C.4.3 Additional NEGCLIP results . . . 140

D Appendix: Data for Improved Vision-Language Model **145**

D.1 Datasets . . . 145

D.2 Baselines . . . 146

D.3 Experiment details . . . 146

D.4 Instructions used for different tasks 147
D.5 Full experiment results 147
D.5.1 CelebA-Attribute full results 147
D.5.2 ImageNet-Subset full results 147

List of Figures

2.1 While large language models (LLMs) offer strong zero/few-shot performance, they are challenging to serve in practice. Traditional ways of training small task-specific models, on the other hand, requires large amount of training data. We propose Distilling step-by-step, a new paradigm that extracts rationales from LLMs as informative task knowledge into training small models, which reduces both the deployed model size as well as the data required for training. 29

2.2 Overview on Distilling step-by-step. We first utilize CoT prompting to extract rationales from an LLM (Section 2.3.1). We then use the generated rationales to train small task-specific models within a multi-task learning framework where we prepend task prefixes to the input examples and train the model to output differently based on the given task prefix (Section 2.3.2). 31

2.3 We use few-shot CoT prompting that contains both an example rationale (highlighted in green) and a label (highlighted in blue) to elicit rationales from an LLM on new input examples. 32

2.4 We compare Distilling step-by-step and Standard finetuning using 220M T5 models on varying sizes of human-labeled datasets. On all datasets, Distilling step-by-step is able to outperform Standard finetuning, trained on the full dataset, by using much less training examples (e.g., 12.5% of the full e-SNLI dataset). 35

2.5 Similar to the plots above, we compare Distilling step-by-step and Standard task distillation using 220M T5 models on varying sizes of unlabeled datasets. Distilling step-by-step is able to outperform Standard task distillation by using only a small subset of the full unlabeled dataset (e.g., 12.5% on ANLI dataset). 35

2.6 We perform Distilling step-by-step and Standard finetuning, using the full human-labeled datasets, on varying sizes of T5 models and compare their performance to LLM baselines, i.e., Few-shot CoT and PINTO Tuning. Distilling step-by-step is able to outperform LLM baselines by using much smaller models, e.g., over $700\times$ smaller model on ANLI. Standard finetuning fails to match LLM's performance using the same model size. 37

2.7 Using unlabeled datasets, we perform Distilling step-by-step and Standard task distillation on varying sizes of T5 models and compare them to Few-shot CoT. Distilling step-by-step outperforms Few-shot CoT by using $2000\times$ smaller models on e-SNLI and $45\times$ smaller models on ANLI and CQA. On SVAMP, by adding unlabeled examples from ASDiv, we close the gap to Few-shot CoT whereas Standard distillation still struggles to catch up. . . . 37

2.8 We show the minimum size of T5 models and the least amount of human-labeled examples required for Distilling step-by-step to outperform LLM's Few-shot CoT by a coarse-grained search. Distilling step-by-step is able to outperform Few-shot CoT using not only much smaller models, but it also achieves so with much less training examples compared to Standard finetuning. On ANLI, we outperform the LLM CoT with a 770M model using only 80% of the dataset, whereas Standard finetuning struggles to match even using 100% of the dataset. . 40

2.9 Similar to Figure 2.8 but using only unlabeled examples, Distilling step-by-step is able to outperform Few-shot CoT using much smaller models and with much less examples compared to Standard task distillation. On SVAMP, the x-axis corresponds to the size of ASDiv dataset used for augmenting the original SVAMP dataset, i.e., $x = 0$ is without augmentation and $x = 100$ corresponds to adding the full ASDiv dataset. 40

3.1 (a) Lost-in-the-middle refers to models' U-shape RAG performance as the relevant context's (e.g., a gold document containing the answer to a query) position varies within the input; (b) We observe models exhibit U-shape attention weights favoring leading and ending contexts, regardless of their actual contents; (c) Models do attend to relevant contexts even when placed in the middle, but are eventually distracted by leading/ending contexts; (d) We propose a calibration mechanism, found-in-the-middle, that disentangles the effect of U-shape attention bias and allows models to attend to relevant context regardless their positions. 47

3.2 **Left and Middle: Qualitatively, the model's response exhibits a strong bias towards the document at the first position (red).** This persists whether the input documents retain their original order (left: gold document at the 10th position) or are randomly shuffled (middle: gold document at the 13th position). Model responses are shown in green, with the gold answer highlighted in yellow. **Right: Our attention calibration method enables the model to find relevant context even when placed in the middle.** 49

3.3 **Quantitatively, the model's response strongly depends on the document at the first position.** This dependence persists even after randomly shuffling the document order, irrespective of its relevance to the query. We measure this dependence by computing the TF-IDF similarity score between the response and each document (gold document originally at position 10). 50

3.4 **Average attention weights reveal a U-shaped positional bias in the model.** Documents at the beginning and end receive greater attention, regardless of order (gold document originally at position 10). Attention is averaged across different decoder layers and attention heads. . . 51

3.5 **Attention calibration effectively improves models' context utilization ability, with its performance curves lying almost entirely above standard vanilla attention (on 22 out of 24 cases). On the most challenging settings where the gold documents are placed in the middle, attention calibration provides 6-15 points improvements.** Top/Bottom row: 10/20-doc. Numbers shown in Table B.2. 56

3.6 **Attention calibration can be applied on top of reordering-based methods to provide further performance boost. This suggests that mitigating attention bias can more fundamentally improve models' context utilization, offering a complementary way to further improve existing RAG pipeline.** Top/Bottom row: 10/20-doc. Numbers shown in Table B.2. 58

4.1 Top row: We define *Vera score gap* as the score difference between the positive and hard negative texts: $\mathrm{Vera}(T^{\mathrm{p}}) - \mathrm{Vera}(T^{\mathrm{n}})$. The entire Vera score gap distribution lies on the positive spectrum, indicating that the template-generated hard negative texts usually have low plausibility. Bottom row: Similarly, *Grammar score gap* is defined by: $\mathrm{Grammar}(T^{\mathrm{p}}) - \mathrm{Grammar}(T^{\mathrm{n}})$. On grammar score, we also find that the distribution largely rests on the positive side, suggesting that most hard negative texts in existing benchmarks exhibit grammatical errors. 67

4.2 Blind commonsense Vera model and Grammar model outperform state-of-the-art CLIP models on nearly *all* existing benchmarks by exploiting the nonsensical and non-fluent artifacts. This suggests that existing benchmarks are hackable and ineffective in measuring compositionality. 68

4.3 Example prompt (black) and actual hard negative (green) generated from ChatGPT. 69

4.4 We compare the Vera (top row) and Grammar (bottom row) score gap distributions between ARO+CREPE (leftmost column), SUGARCREPE without adversarial refinement (middle), and SUGARCREPE (rightmost). Top row: We see that Vera score gap distribution shifts from the positive spectrum to more centered around zero from ARO+CREPE to SUGARCREPE without refinement. After adversarial refinement, we ensure the score gap distribution is centered around zero on SUGARCREPE. Bottom row: Similarly, from ARO+CREPE to SUGARCREPE, we see the Grammar score gap distribution shifts from the positive spectrum to centered around zero. 73

4.5 We plot pretrained vision-language models' zero-shot top-1 accuracy on ImageNet versus their retrieval recall@1 on SUGARCREPE, where r is the Pearson correlation coefficient. This plot suggests that models' ImageNet zero-shot accuracy positively correlates with their compositionality. 77

5.1 For a given image, the CLIP embedding space is static and structured based on overall semantics. However, FocalLens dynamically rearranges the embedding space based on the specified condition, bringing instances that are more similar under that condition closer together. We show the top-2 nearest neighbors for both CLIP and FocalLens embeddings (once conditioned on "background" and once on "quantity"). 82

5.2 FocalLens is applied to two vision-language models to extract text-conditioned visual features: **(a)** modifying Llava-like VLMs, which already have text-conditioning capabilities, to produce a global visual feature, and **(b)** modifying ViT [Dosovitskiy, 2020] based CLIP-like VLMs, which already produce a global visual feature, to condition their output feature based on a text condition. 85

5.3 ColorShape examples with a query image, three conditions, and corresponding positives and distractors. 90

5.4 Linear probing results comparing CLIP and FocalLens-CLIP. 95

5.5 Comparison between CLIP and FocalLens-CLIP on conditional image retrieval. 96

C.1 Taxonomy of hard negatives considered in SUGARCREPE. 137

C.2 Example prompt templates (black) and outputs (green) from ChatGPT for REPLACE hard negatives. 142

C.3 Example prompt templates (black) and outputs (green) from ChatGPT for SWAP hard negatives. 143

C.4 Example prompt templates (black) and outputs (green) from ChatGPT for ADD hard negatives. 143

List of Tables

2.1 Distilling step-by-step works with different sizes of LLMs. When rationales are extracted from a 20B GPT-NeoX model, Distilling step-by-step is still able to provide performance lift compared to standard finetuning on 220M T5 models. 41

2.2 Our proposed multi-task training framework consistently leads to better performances than treating rationale and label predictions as a single task. Single-task training can at times lead to worse performance than standard finetuning. 42

3.1 Number of examples where the most likely used document in the model's generation falls within the first half of documents receiving higher model attention or second half receiving lower attention. We see that there is a strong correlation where documents receiving higher attention are more likely to be used in model's response. 51

3.2 High correlations between model attention with document relevance and positional bias supports our hypothesized model. 53

3.3 Calibrated attention outperforms existing methods in ranking the relevance of retrieved contexts given a user query. We report Recall@3 on NaturalQuestion when gold documents are placed in the middle of input context. 55

4.1 Existing compositionality benchmarks rely on procedurally-generated hard negatives which often do not make logical sense or are not fluent due to grammatical errors. 67

4.2 We report the number of hard negative captions of all types in SUGARCREPE. 71

4.3 We present example positive texts and their hard negatives in ARO+CREPE (generated using existing procedures) and SUGARCREPE (generated with ChatGPT). SUGARCREPE brings significant improvements in commonsense and fluency. 72

4.4 We compare the commonsense and grammar scores on hard negatives in ARO+CREPE and SUGARCREPE. We report both their respective average scores and the ratio where SUGARCREPE has higher score than ARO+CREPE in pairwise comparison. Overall, SUGARCREPE has hard negatives with better commonsense and grammar. 72

4.5 Re-evaluating hard negative augmented training shows that the method's improvements on existing benchmarks (ARO+CREPE) are hugely overestimated, particularly when the test hard negative type matches the one used in training, which can be attributed to overfitting the artifacts. Color notations: Gains compared to standard CLIP (finetuned / from scratch) $> 10\%$. 75

4.6 Our evaluation of pretrained CLIP models on SUGARCREPE shows that they demonstrate compositionality on some hard negatives but are far from human performance on others, especially on SWAP hard negatives or ones perturbing attributes and relations (also illustrated in Figure 4.5: lower overall performance on SWAP, and lower performances on attributes/relations compared to objects). We additionally evaluate recently introduced GPT-4V [OpenAI, 2023]. While it demonstrates strong results, there is still gap to human-level performance. . . 76

5.1 Image-image retrieval results on ColorShape dataset. Conditional representations from FocalLens better capture the given conditions compared to the task-agnostic representations of CLIP. 90

5.2 Results on CelebA-Attribute and GeneCIS. 91

5.3 Results on ImageNet-Subset and fine-grained classification datasets. 91

5.4 Comparison between FocalLens-MLLM and FocalLens-CLIP on fuzzy conditions with CelebA-Identity. 93

5.5 Image-Text Retrieval on SugarCrepe for vision-language compositionality evaluation. 94

5.6 Image-Text Retrieval on MMVP-VLM. 94

A.1 Dataset statistics used in our experiments. 128

B.1 Rank correlations of linear and log-linear models. 130
B.2 Our proposed attention intervention by calibrated attention stably improves models' RAG performances compared to existing re-ordering based baselines. 131

C.1 Summary on vision-language compositionality benchmarks. SUGARCREPE considers image-to-text formulation to enable larger scale evaluation set. In addition, SUGARCREPE considers a wide range of hard negative types. SHUFFLE and NEGATE are omitted as they introduce inevitable biases discussed in Sec. 4.4.2. 136
C.2 Blind models (i.e. , Vera and Grammar model) outperform all 17 existing pretrained CLIP models on nearly all existing benchmark tasks. This implies that current benchmarks fail to faithfully measure a model's vision-language compositionality. 140
C.3 Human evaluation results on the comparisons between hard negatives in ARO+CREPE and SUGARCREPE. We report the counts (out of 100 sampled examples) that the human user considers better or tie, w.r.t. both commonsense and grammatical correctness. 141
C.4 Model performances on SUGARCREPE when trained with hard negatives generated through similar procedure as how SUGARCREPE is created. 141

D.1 ImageNet-Subset datasets and number of classes per each. 145
D.2 Training hyperparameters. 146
D.3 Instructions and templates used for different datasets and conditions. 147
D.4 Full results on CelebA-Attribute. 148
D.5 Full results on ImageNet-Subset. 148

Chapter 1

Introduction

Today's rapid evolution of AI is largely driven by the scaling of two pivotal factors: *model* and *data* [Mahajan et al., 2018; Shoeybi et al., 2019; Raffel et al., 2020; Brown et al., 2020; Radford et al., 2021a; Chowdhery et al., 2022; Chung et al., 2022; OpenAI, 2022]. In particular, recent studies have pointed out that scaling both (a) the size of Transformer models [Vaswani et al., 2017] and (b) data volume, typically by crawling web contents, in tandem leads to predictable improvement in AI model performances [Kaplan et al., 2020; Hoffmann et al., 2022; Ghorbani et al., 2021]. This is evidenced by the unprecedented capabilities emerging from modern AI models across domains [Bubeck et al., 2023; Anil et al., 2023; Li et al., 2023c; Alayrac et al., 2022]. Nonetheless, the scaling development trajectory poses two predominant challenges in practice.

Model deployment challenge. As seen in language models that grow exponentially in size, consisting of hundreds of billions of parameters or more [Brown et al., 2020; Chowdhery et al., 2022], the computational requirements to serve the models become far beyond affordable for most applications. To make the matter worse, it is even more resource demanding if one were to further finetune these large pretrained models for improvements on downstream applications. This accentuates efficient ways in deploying and adapting large models for a broad range of practical scenarios with varying levels of available resources.

Data curation challenge. Using vision-language models as an example [Radford et al., 2021b], state-of-the-art models trained on billions of images have substantially improved general recognition capabilities, as measured by standard benchmarks such as ImageNet accuracy [Schuhmann et al., 2022b]. However, when deployed in real-world settings, these models often lack fine-grained or compositional visual understand-

ing [Ma et al., 2022; Yuksekgonul et al., 2023]. For instance, given an image of a horse eating grass, they may fail to distinguish the correct caption from an incorrect one such as "grass eating horse", which shares similar visual elements but differs in composition and meaning. As a result, beyond standard metrics, it becomes increasingly important to design benchmarks that can reliably characterize the desired behaviors of such models. Furthermore, it is essential to curate and scale training data in a way that effectively induces these capabilities.

Given these two major challenges, this thesis seeks to develop effective techniques for both model deployment and data curation for modern large-scale models, ultimately extending the reach of modern AI to a wide range of applications. First, on the model side, this thesis focuses on the use cases of modern large language models (LLMs) and introduce strategies to improve the efficiency and effectiveness of their deployment and adaptation (Section 1.1). Second, on the data side, this thesis focuses on vision-language models (VLMs) to examine how to curate benchmarks that more reliably evaluate their emergent capabilities beyond standard recognition accuracy, and how to further improve these models through the design of more effective training data (Section 1.2). Collectively, the core thesis is supported by four in-depth studies that address challenges spanning both the model and data aspects in today's large-scale AI development, encompassing modalities across natural language processing and computer vision, as overviewed below.

1.1 Effective model deployment and adaptation with large language models

1.1.1 Compute-efficient model deployment

Deploying large models is challenging for practical applications due to computation limitations, most notably with today's large language models (LLM) reaching over 100B parameters [Chowdhery et al., 2022; Brown et al., 2020]. Thus, practitioners instead train small, deployable models through standard approaches such as finetuning or distillation. However, standard approaches require large amounts of training data to achieve comparable performance to the large models. In Chapter 2, we tackle this trade-off between model size and number of training data by proposing Distilling step-by-step [Hsieh et al., 2023b], a mechanism that trains smaller models that outperform larger ones, and achieves so by leveraging less training data needed by existing approaches. The core idea is to extract rationales from a large teacher model as additional supervision

for training small models within a multi-task framework. Distilling step-by-step enables a 700× smaller model trained with only 80% of data on a natural language inference benchmark to outperform a few-shot prompted 540B LLM.

1.1.2 Cost-effective model adaptation

While large pretrained models display strong general capabilities, there often is a need to adapt these general-purpose models to downstream tasks for further performance improvements, especially when the tasks require specific knowledge or skills. However, the sheer size of these models makes further finetuning computationally inhibiting [Lester et al., 2021]. As a result, retrieval-augmented generation has become one popular and efficient way in equipping pretrained language models with external knowledge that makes the models adapt easily to new tasks without expensive finetuning. In Chapter 3, we devise a novel technique that significantly enhances LLMs' capability in understanding and making use of long input contexts, which is core to the success of retrieval-augmented generation [Hsieh et al., 2024a].

1.2 Effective data curation for measuring and improving compositionality in vision-language models

1.2.1 Reliable data curation for vision-language model evaluation

Compositionality, that "the meaning of the whole is a function of the meanings of its parts", is held to be a key characteristic of human intelligence, enabling people to reason, communicate, and plan. For AI models, compositional understanding allows for more effective combination of known concepts and improved generalization to novel situations, and it has long been pursued by researchers. Given its importance, there has been a recent surge of new evaluation benchmarks proposed to evaluate vision-language models' compositionality. In Chapter 4, however, we reveal that most existing compositionality benchmarks contain significant dataset biases rendering them unreliable in correctly assessing models' compositionality. To remedy this, we introduce a new benchmark through meticulously designed data curation pipeline for vision-language compositionality evaluation, and provide comprehensive evaluation and insights on this important direction [Hsieh et al., 2024b].

1.2.2 Effective data curation for improving vision-language models

Visual understanding is inherently contextual—what we focus on in an image depends on the task at hand. For instance, given an image of a person holding a bouquet of flowers, we may focus on either the person such as their clothing, or the type of flowers, depending on the context of interest. Yet, most existing image encoding paradigms represent an image as a fixed, generic feature vector, overlooking the potential needs of prioritizing varying visual information for different downstream use cases. In Chapter 5, we introduce a *conditional* visual encoding method that produces different representations for the same image based on the context of interest, expressed flexibly through natural language [Hsieh et al., 2025]. Importantly, we achieve this by strategically leveraging *vision instruction tuning data* to contrastively finetune a pretrained vision encoder to take natural language instructions as additional inputs for producing conditional image representations. Extensive experiments validate that conditional image representations from our method better pronounce the visual features of interest compared to generic features produced by standard vision encoders like CLIP. We show effectiveness of the method on a range of downstream tasks including image-image retrieval, image classification, and image-text retrieval, with significant gain on the challenging compositional understanding and fine-grained visual understanding benchmarks.

Chapter 2

Efficient Language Model Deployment: Training Smaller Models with Less Data

2.1 Introduction

Despite the impressive few-shot ability offered by large language models (LLMs) [Brown et al., 2020; Chowdhery et al., 2022; Thoppilan et al., 2022; Hoffmann et al., 2022; Smith et al., 2022b; Zhang et al., 2022], these models are challenging to deploy in real world applications due to their sheer size. Serving a single 175 billion LLM requires at least 350GB GPU memory using specialized infrastructure [Zheng et al., 2022]. To make matters worse, today's state-of-the-art LLMs are composed of over 500B parameters [Chowdhery et al., 2022], requiring significantly more memory and compute. Such computational requirements are far beyond affordable for most product teams, especially for applications that require low latency performance.

To circumvent these deployment challenges of large models, practitioners often choose to deploy smaller specialized models instead. These smaller models are trained using one of two common paradigms: *finetuning* or *distillation*. Finetuning updates a pretrained smaller model (e.g. BERT [Devlin et al., 2018] or T5 [Raffel et al., 2020]) using downstream human annotated data [Howard and Ruder, 2018]. Distillation trains the same smaller models with labels generated by a larger LLM [Tang et al., 2019; Wang et al., 2021; Smith et al., 2022a; Arora et al., 2022]. Unfortunately, these paradigms reduce model size at a cost: to achieve comparable performance to LLMs, finetuning requires expensive human labels, and distillation requires large

amounts of unlabeled data which can be hard to obtain [Tang et al., 2019; Liang et al., 2020].

In this chapter, we introduce **Distilling step-by-step**, a new simple mechanism for training smaller models with less training data. Our mechanism reduces the amount of training data required for both finetuning and distillation of LLMs into smaller model sizes. Core to our mechanism is changing our perspective from viewing LLMs as a source of noisy labels to viewing them as agents that can reason: LLMs can produce natural language rationales justifying their predicted labels [Wei et al., 2022; Kojima et al., 2022]. For example, when asked "*Jesse's room is* 11 *feet long and* 15 *feet wide. If she already has* 16 *square feet of carpet. How much more carpet does she need to cover the whole floor?*", an LLM can be prompted by chain-of-thought (CoT) technique [Wei et al., 2022] to provide intermediate rationales "*Area* $=$ *length* $\times$ *width. Jesse's room has* 11×15 *square feet.*" that better connects the input to the final answer "$(11 \times 15) - 16$". These *rationales* can contain relevant task knowledge, such as "*Area* $=$ *length* $\times$ *width*", that may originally require many data for small task-specific models to learn. We thus utilize these extracted rationales as additional, richer information to train small models through a multi-task training setup, with both label prediction and rationale prediction tasks [Raffel et al., 2020; Narang et al., 2020].

Distilling step-by-step allows us to learn task-specific smaller models that outperform LLMs using over $500\times$ less model parameters, and it does so with far fewer training examples compared to traditional finetuning or distillation (Figure 2.1). Our results show three promising empirical conclusions across 4 NLP benchmarks. First, compared to both finetuning and distillation, our resulting models achieve better performance with over 50% less training examples on average across datasets (and up to over 85% reduction). Second, our models outperform LLMs with much smaller model sizes (up to $2000\times$ smaller), drastically reducing the computation cost required for model deployment. Third, we simultaneously reduce the model size as well as the amount of data required to outperform LLMs. We surpass the performance of 540B parameter LLMs using a 770M T5 model; this smaller model only uses 80% of a labeled dataset that would otherwise be required if using an existing finetuning method. When only unlabeled data is present, our small models still perform on par or better than LLMs. We outperform 540B PaLM's performance with only a 11B T5 model. We further show that when a smaller model performs worse than an LLM, Distilling step-by-step can more efficiently leverage additional unlabeled data to match the LLM performance compared to the standard distillation approach.

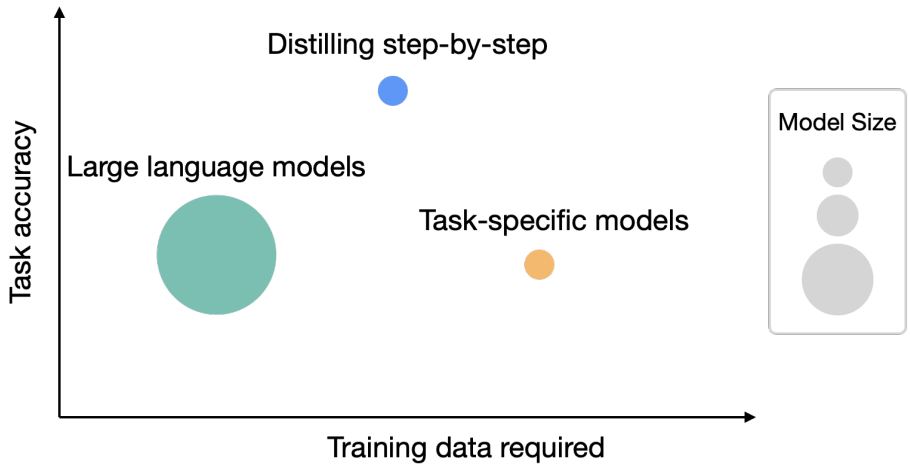


Figure 2.1: While large language models (LLMs) offer strong zero/few-shot performance, they are challenging to serve in practice. Traditional ways of training small task-specific models, on the other hand, requires large amount of training data. We propose Distilling step-by-step, a new paradigm that extracts rationales from LLMs as informative task knowledge into training small models, which reduces both the deployed model size as well as the data required for training.

2.2 Related work

Our method distills task-specific knowledge of LLMs into smaller specialist models by leveraging the emergent reasoning capabilities of today's LLMs. We draw on knowledge distillation research and methods that learn from both human-generated rationales and LLM-generated rationales.

Knowledge distillation from large models. Knowledge distillation has been successfully used to transfer knowledge from larger, more competent teacher models into smaller student models affordable for practical applications [Buciluǎ et al., 2006; Hinton et al., 2015; Beyer et al., 2022; West et al., 2021; Fu et al., 2023]. It supports learning from limited labeled data, since the larger teacher model is often used to generate a training dataset with noisy pseudo labels [Chen et al., 2020; Iliopoulos et al., 2022; Wang et al., 2021; Smith et al., 2022a; Arora et al., 2022; Agrawal et al., 2022]. The one limitation that knowledge distillation often faces is its reliance on large amounts of unlabelled data required to create a useful noisy training dataset. Although prior work has explored using data augmentation techniques to reduce this hunger for data [Tang et al., 2019; Liang et al., 2020; Srinivas and Fleuret, 2018; Milli et al., 2019], we propose an alternative approach: we reduce the need for large unlabeled data by distilling not just labels but also the teacher's rationales.

Learning with human rationales. While utilizing LLM-generated rationales is a new exciting area of investigation, using human-generated rationales has a rich history [Hase and Bansal, 2021]. For instance, human rationales can be used to regularize model behavior [Ross et al., 2017]; it can be used as additional inputs to guide a model's predictions [Rajani et al., 2019]; it can be used to improve overall model performance [Zaidan et al., 2007; Zhang et al., 2016; Camburu et al., 2018; Hancock et al., 2019; Pruthi et al., 2022]; and human rationales can be used as gold standard labels to make models more interpretable by generating similar rationales [Wiegreffe et al., 2021; Narang et al., 2020; Eisenstein et al., 2022]. Unfortunately, human rationales are expensive.

Learning with LLM generated rationales. Today's LLMs are capable of explaining their predictions by generating high-quality reasoning steps [Wei et al., 2022; Kojima et al., 2022]. These reasoning steps have been used to augment input prompts to LLMs, improving their few-shot or zero-shot performance [Wei et al., 2022; Kojima et al., 2022; Wang et al., 2022d]; reasoning steps have also been used as additional finetuning data "self-improve" LLMs [Zelikman et al., 2022; Huang et al., 2022]. Unfortunately, regardless of how LLMs are improved, their large size limits their utility in most test-time applications.

By contrast, we leverage generated rationales as informative supervision to train smaller task-specific models, i.e. models that can be deployed without incurring large computation or memory costs. Several concurrent works have also proposed a similar idea to ours – that of using extracted rationales as supervision [Wang et al., 2022b; Ho et al., 2022; Magister et al., 2022; Li et al., 2023e]. Amongst them, PINTO [Wang et al., 2022b] relies on an LLM to generate rationales at test-time, and thus does not fully solve deployment challenges. Compared with Ho et al. [2022] and Magister et al. [2022], we go beyond their experiments to provide a granular study by varying training dataset size, exploring downstream model sizes, and demonstrating the effectiveness of our method on fully unlabeled datasets.

2.3 Distilling step-by-step

We propose a new paradigm, Distilling step-by-step, that leverages the ability of LLMs to reason about their predictions to train smaller models in a data-efficient way. Our overall framework is illustrated in Figure 2.2. Our paradigm has two simple steps: First, given an LLM and an unlabeled dataset, we prompt the LLM to

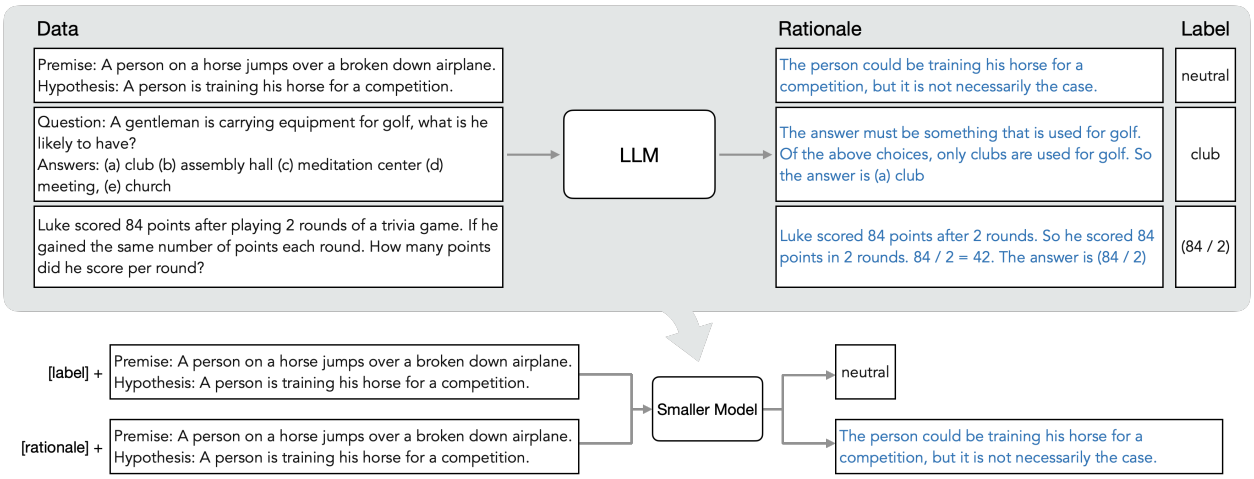


Figure 2.2: Overview on Distilling step-by-step. We first utilize CoT prompting to extract rationales from an LLM (Section 2.3.1). We then use the generated rationales to train small task-specific models within a multi-task learning framework where we prepend task prefixes to the input examples and train the model to output differently based on the given task prefix (Section 2.3.2).

generate output labels along with *rationales* to justify the labels. Rationales are natural language explanations that provide support for the model's predicted label (see Figure 2.2). Second, we leverage these rationales in addition to the task labels to train smaller downstream models. Intuitively, rationales provide richer, more detailed information about why an input is mapped to a specific output label, and often contain relevant task knowledge that may be hard to infer solely from the original inputs.

2.3.1 Extracting rationales from LLMs

Recent studies observe one intriguing emerging property of LLMs: their ability to generate rationales that support their predictions [Wei et al., 2022; Kojima et al., 2022]. While the studies have largely focused on how to elicit such reasoning capability from LLMs [Nye et al., 2021; Wei et al., 2022; Kojima et al., 2022], we use them in training smaller downstream models.

Specifically, we utilize Chain-of-Thought (CoT) prompting [Wei et al., 2022] to elicit and extract rationales from LLMs. As illustrated in Figure 2.3, given an unlabeled dataset $x_i \in D$, we first curate a prompt template p that articulates how the task should be solved. Each prompt is a triplet $(x^{\mathrm{p}}, r^{\mathrm{p}}, y^{\mathrm{p}})$, where x^{p} is an example input, y^{p} is its corresponding label and r^{p} is a user-provided rationale that explains why x^{p} can be categorized as y^{p}. We append each input x_i to p and use it as an input to prompt the LLM to generate

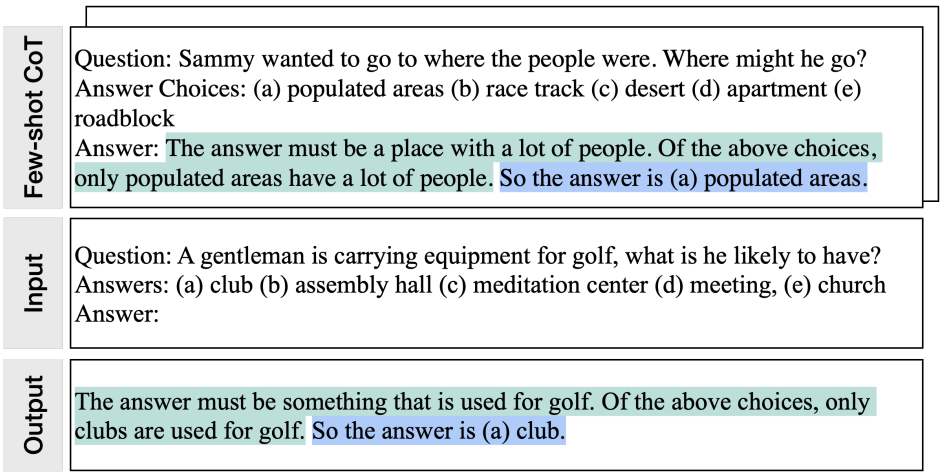


Figure 2.3: We use few-shot CoT prompting that contains both an example rationale (highlighted in green) and a label (highlighted in blue) to elicit rationales from an LLM on new input examples.

rationales and labels for each $x_i \in D$. With the demonstrations seen in p, the LLM is able to mimic the triplet demonstration to generate the rationale $\hat{r}_i$ and output $\hat{y}_i$ for x_i.

2.3.2 Training smaller models with rationales

We first describe the current framework for learning task-specific models. With this framework in place, we extend it to incorporate rationales into the training process. Formally, we denote a dataset as $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ where each x_i represents an input and y_i is the corresponding desired output label. While our framework supports inputs and outputs of any modality, our experiments limits x and y to be natural language. This text-to-text framework [Raffel et al., 2020] encompasses a variety of NLP tasks: classification, natural language inference, question answering and more.

Standard finetuning and task distillation. The most common practice to train a task-specific model is to finetune a pretrained model with supervised data [Howard and Ruder, 2018]. In the absence of human-annotated labels, task-specific distillation [Hinton et al., 2015; Tang et al., 2019] uses LLM teachers to generates pseudo noisy training labels, $\hat{y}_i$ in place of y_i [Wang et al., 2021; Smith et al., 2022a; Arora et al., 2022].

For both scenarios, the smaller model f is trained to minimize the label prediction loss:

$$\mathcal{L}_{\text{label}} = \frac{1}{N}\sum_{i=1}^{N} \ell(f(x_i), \hat{y}_i), \tag{2.1}$$

where ℓ is the cross-entropy loss between the predicted and target tokens. Note that for ease of exposition, we overload $\hat{y}_i$ in Eq. 2.1 to be either human-annotated labels y_i for the standard finetuning case, or LLM-predicted labels $\hat{y}_i$ for the model distillation case.

Multi-task learning with rationales. To create a more explicit connection between x_i's to $\hat{y}_i$'s, we use extracted rationales $\hat{r}_i$ as additional supervision. There are several ways to incorporate rationales into the downstream model's training process. One straightforward approach is feed $\hat{r}_i$ as an additional input—as proposed by other concurrent research [Rajani et al., 2019; Wang et al., 2022b]. In other words, the $f(x_i, \hat{r}_i) \rightarrow \hat{y}_i$ is trained with both text and rationale $[x, r]$ as inputs:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \ell(f(x_i, \hat{r}_i), \hat{y}_i). \tag{2.2}$$

Unfortunately, this design requires an LLM to first generate a rationale before the f can make a prediction. The LLM is still necessary during deployment, limited its deployability.

In Distilling step-by-step, instead of using rationales as additional model inputs, we frame learning with rationales as a multi-task problem. Specifically, we train the model $f(x_i) \rightarrow (\hat{y}_i, \hat{r}_i)$ to not only predict the task labels but also generate the corresponding rationales given the text inputs:

$$\mathcal{L} = \mathcal{L}_{\text{label}} + \lambda \mathcal{L}_{\text{rationale}}, \tag{2.3}$$

where $\mathcal{L}_{\text{label}}$ is the label prediction loss in Eq. 2.1 and $\mathcal{L}_{\text{rationale}}$ is the *rationale generation loss*:

$$\mathcal{L}_{\text{rationale}} = \frac{1}{N}\sum_{i=1}^{N} \ell(f(x_i), \hat{r}_i). \tag{2.4}$$

The rationale generation loss enables the model to learn to generate the intermediate reasoning steps for the prediction, and could therefore guide the model in better predicting the resultant label. This is our proposed

Distilling step-by-step. Compared with Eq. 2.2, the rationale $\hat{r}_i$ is not required in the test time, which removes the need for an LLM at test-time.

We prepend "task prefixes" (`[label]`, `[rationale]`) to the input examples and train the smaller model to output $\hat{y}_i$ when `[label]` is provided and to produce $\hat{r}_i$ with `[rationale]` [Raffel et al., 2020].

2.4 Experiments

We empirically validate the effectiveness of Distilling step-by-step. First, we show that when compared to standard finetuning and task distillation approaches, Distilling step-by-step achieves better performance with much fewer number of training examples, substantially improving the data efficiency to learn small task-specific models (Sec. 2.4.1). Second, we show that Distilling step-by-step surpasses the performance of LLMs with much smaller model size, drastically lowering the deployment cost compared to LLMs (Sec. 2.4.2). Third, we investigate the minimum resources required, w.r.t. both number of training examples and model size, for Distilling step-by-step to outperform LLMs. We show that Distilling step-by-step outperforms LLMs by using less data and smaller model, simultaneously improving both data- and deployability-efficiency (Sec. 2.4.3). Finally, we perform ablation studies to understand the influence of different components and design choices in the Distilling step-by-step framework (Sec. 2.4.4).

Setup. In the experiments, we consider the 540B PaLM model [Chowdhery et al., 2022] as the LLM. For task-specific downstream models, we use T5 models [Raffel et al., 2020] where we initialize the models with pretrained weights obtained from publicly available sources[1]. For CoT prompting, we follow Wei et al. [2022] when available, and curate our own examples for new datasets. We include more implementation details in Appendix A.

Datasets. We conduct the experiments on 4 popular benchmark datasets across 3 different NLP tasks: *e-SNLI* [Camburu et al., 2018] and *ANLI* [Nie et al., 2020] for natural language inference; *CQA* [Talmor et al., 2019; Rajani et al., 2019] for commonsense question answering; *SVAMP* [Patel et al., 2021] for arithmetic math word problems. We include more dataset details in Appendix A.1.2.

[1] `https://huggingface.co/`

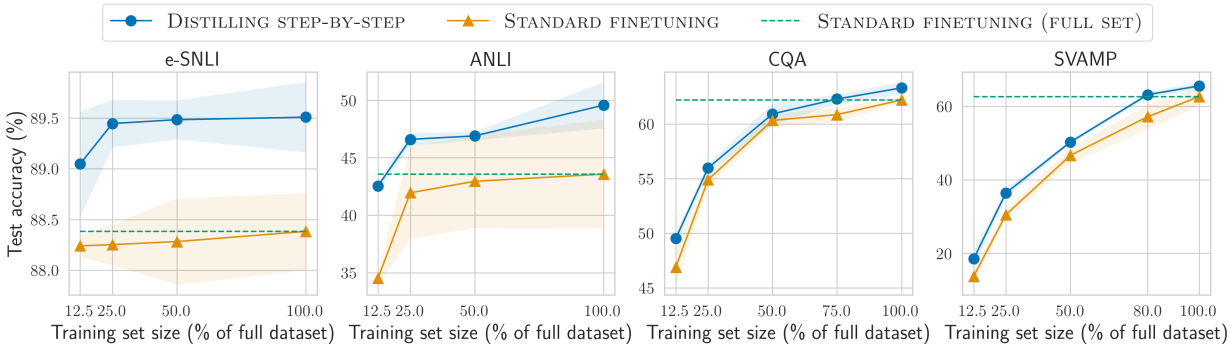


Figure 2.4: We compare Distilling step-by-step and Standard finetuning using 220M T5 models on varying sizes of human-labeled datasets. On all datasets, Distilling step-by-step is able to outperform Standard finetuning, trained on the full dataset, by using much less training examples (e.g., 12.5% of the full e-SNLI dataset).

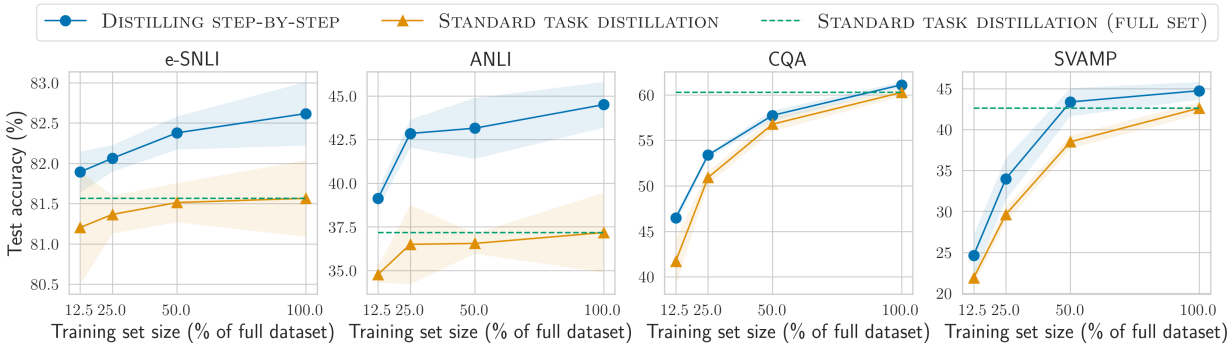


Figure 2.5: Similar to the plots above, we compare Distilling step-by-step and Standard task distillation using 220M T5 models on varying sizes of unlabeled datasets. Distilling step-by-step is able to outperform Standard task distillation by using only a small subset of the full unlabeled dataset (e.g., 12.5% on ANLI dataset).

2.4.1 Reducing training data

We compare Distilling step-by-step to two most common methods in learning task-specific models: (1) Standard finetuning when human-labeled examples are available, and (2) Standard task distillation when only unlabeled examples are available. Specifically, standard finetuning refers to the prevailing pretrain-then-finetune paradigm that finetunes a model with ground-truth labels via standard label supervision [Howard and Ruder, 2018]. On the other hand, when only unlabeled examples are available, standard task distillation learns the task-specific model by treating a teacher LLM's predicted labels as ground-truths [Hinton et al., 2015; Chen et al., 2020; Wang et al., 2021; Smith et al., 2022a; Arora et al., 2022].

In the following set of experiments, we fix the task-specific models to be 220M T5-Base models, and

compare the task performances achieved by different methods under varying number of available training examples.

Distilling step-by-step outperforms standard finetuning with much less labeled examples. When finetuned with human-labeled examples, Figure 2.4 shows that Distilling step-by-step consistently achieves better performance than standard finetuning across varying numbers of labeled examples used. Furthermore, we see that Distilling step-by-step can achieve the same performance as standard finetuning with much less labeled examples. In particular, by using only 12.5% of the full e-SNLI dataset, Distilling step-by-step can outperform standard finetuning trained with 100% of the full dataset. Similarly, we achieve 75%, 25%, and 20% reduction in training examples required to outperform standard finetuning on ANLI, CQA, and SVAMP respectively.

Distilling step-by-step outperforms standard distillation with much less unlabeled examples. When only unlabeled data is available, we compare Distilling step-by-step to standard task distillation. In Figure 2.5, we observe an overall similar trend to the finetuning setup. Specifically, we see that Distilling step-by-step outperforms standard task distillation on all 4 datasets under different numbers of unlabeled data used. We as well see that Distilling step-by-step requires much less unlabeled data to outperform standard task distillation. For instance, we need only 12.5% of the full unlabeled dataset to outperform the performance achieved by standard task distillation using 100% of the training examples on e-SNLI dataset.

2.4.2 Reducing model size

In the following set of experiments, we hold the training set size fixed (using 100% of the datasets), and compare varying sizes of small T5 models trained with Distilling step-by-step and standard approaches to LLMs. Specifically, we consider 3 different sizes of T5 models, i.e., 220M T5-Base, 770M T5-Large, and 11B T5-XXL. For LLMs, we include two baseline methods: (1) FEW-SHOT COT [Wei et al., 2022], and (2) PINTO TUNING [Wang et al., 2022b]. Few-shot CoT directly utilizes CoT demonstrations to prompt the 540B PaLM to generate intermediate steps before predicting the final labels without any further finetuning of the LLM. PINTO tuning refers to our extension of Wang et al. [2022b] to handle tasks beyond question-answering, which are not studied by Wang et al. [2022b]. Here, we finetune a 220M T5-Base model

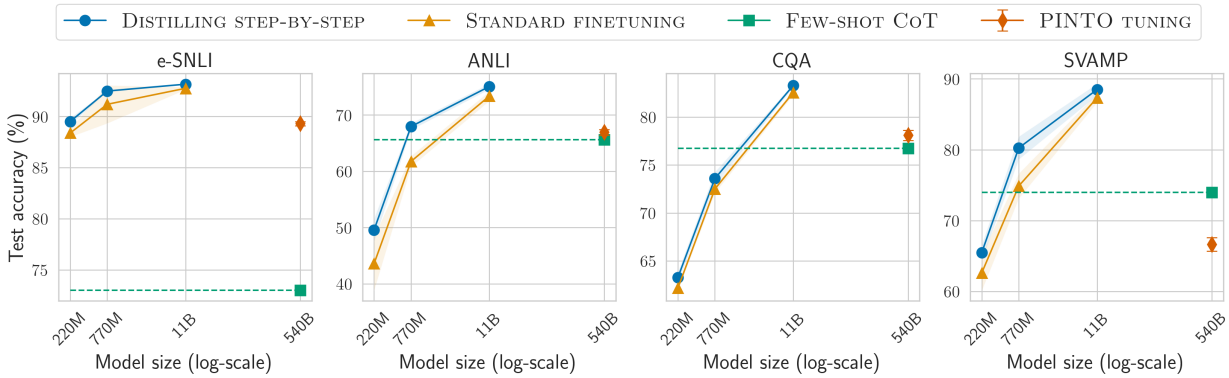


Figure 2.6: We perform Distilling step-by-step and Standard finetuning, using the full human-labeled datasets, on varying sizes of T5 models and compare their performance to LLM baselines, i.e., Few-shot CoT and PINTO Tuning. Distilling step-by-step is able to outperform LLM baselines by using much smaller models, e.g., over 700× smaller model on ANLI. Standard finetuning fails to match LLM's performance using the same model size.

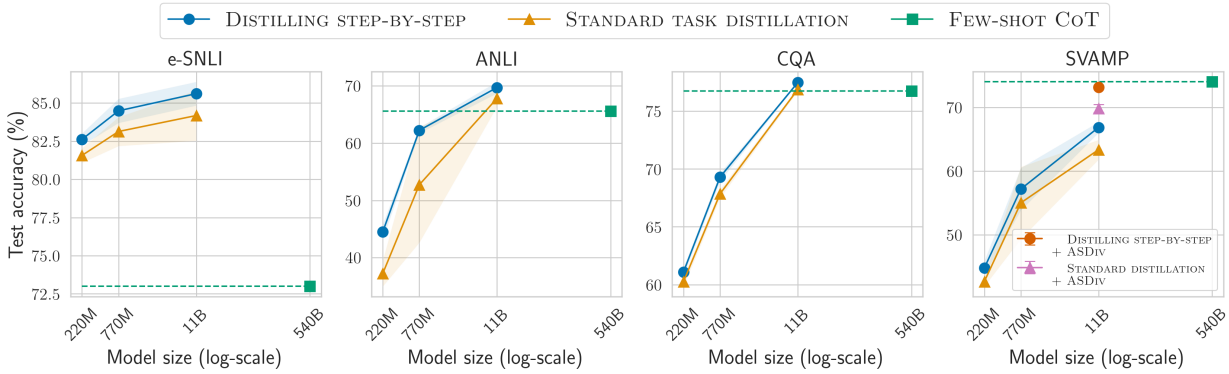


Figure 2.7: Using unlabeled datasets, we perform Distilling step-by-step and Standard task distillation on varying sizes of T5 models and compare them to Few-shot CoT. Distilling step-by-step outperforms Few-shot CoT by using 2000× smaller models on e-SNLI and 45× smaller models on ANLI and CQA. On SVAMP, by adding unlabeled examples from ASDiv, we close the gap to Few-shot CoT whereas Standard distillation still struggles to catch up.

on top of the outputs generated from the PaLM model, which can be viewed as a finetuning method for LLMs with additional parameters [Zhang et al., 2020; Lester et al., 2021].

We present the experimental results under the two broad scenarios of having access to labeled datasets or unlabeled datasets in Figure 2.6 and Figure 2.7, respectively. We plot each method by their deployed model sizes for prediction (x-axis), and their corresponding task performances (y-axis).

Distilling step-by-step improves over standard baselines across varying model sizes used. In Figure 2.6 and Figure 2.7 respectively, we see that Distilling step-by-step consistently improves over standard finetuning and standard distillation across all sizes of T5 models. The improvements are most pronounced on ANLI, where Distilling step-by-step outperforms standard finetuning and distillation by an average of 8% and 13% on task accuracy respectively.

Distilling step-by-step outperforms LLMs by using much smaller task-specific models. In Figure 2.6 when human-labeled datasets are available, Distilling step-by-step can always outperform Few-shot CoT and PINTO tuning on all 4 datasets considered, by using much smaller T5 models. For instance, we can achieve better performances than 540B PaLM model's Few-shot CoT with 220M (over $2000\times$ smaller) T5 model on e-SNLI, 770M (over $700\times$ smaller) T5 models on ANLI and SVAMP, and 11B (over $45\times$ smaller) T5 model on CQA. These results hold true even by further finetuning the 540B PaLM model on available labeled data with PINTO tuning[2].

In Figure 2.7, by only utilizing unlabeled examples, Distilling step-by-step also outperforms the teacher LLM on 3 out of 4 datasets. Specifically, Distilling step-by-step surpasses the 540B PaLM model's Few-shot CoT performance by using 11B T5 with less than 3% of PaLM's size. On SVAMP where the distilled model underperforms, we hypothesize that the performance gap is due to the relatively small number of data points in the dataset (i.e., 800). In reaction, we propose to augment the dataset with additional unlabeled examples to close the performance gap as shown in next.

Unlabeled data augmentation further improves Distilling step-by-step. We augment the SVAMP training set with unlabeled examples from the *ASDiv* dataset [Miao et al., 2020]. ASDiv dataset contains a total of $2,305$ examples, where each example is a math word problem similar to the ones in SVAMP. In Figure 2.7 on SVAMP, we show the performances of Distilling step-by-step and standard task distillation using 11B T5 model after augmenting the training set with ASDiv. We see the data augmentation much improves the performance for both Distilling step-by-step and standard task distillation. However, even with the added unlabeled examples, standard task distillation still underperforms Few-shot CoT. On the other hand,

[2]We note that PETuning methods may outperform PINTO tuning. However, they require massive resource in both training and deployment, which is not our focus in this chapter.

Distilling step-by-step is able to much more efficiently exploit the value of the added examples to achieve the same performance level of Few-shot CoT, again, using a T5 model of size less than 3% of the 540B PaLM.

2.4.3 Outperforming LLMs using minimum model size and least training data

Here, using the LLM's performance as an anchor point, we explore the most efficient resource requirements in terms of both number of training examples and deployed model size, that Distilling step-by-step and standard finetuning/distillation need to outperform the LLM. We present the results, again under human-labeled setting and unlabeled setting, in Figure 2.8 and Figure 2.9 respectively. We visualize the results by plotting different resultant models by (1) the number of training examples used (x-axis), (2) the final task performance achieved (y-axis), and (3) the size of the model (visualized by the size of the shaded area).

Distilling step-by-step outperforms LLMs with much smaller models by using less data. On all datasets in Figure 2.8, we see that Distilling step-by-step outperforms PaLM's Few-shot CoT with much smaller T5 models using only a subset of the available training examples. Specifically, on e-SNLI, Distilling step-by-step can achieve better performance than Few-shot CoT with a model over $2000\times$ smaller (220M T5) and only 0.1% of the full dataset. In Figure 2.9 where only unlabeled datasets are available, we observe the same trend that Distilling step-by-step can, at most time, outperform Few-shot CoT with smaller model as well as less data. For instance, on ANLI, Distilling step-by-step outperforms the LLM with a $45\times$ smaller model and 50% of the full unlabeled set.

Standard finetuning and distillation require more data and larger model. Finally, in Figure 2.8 and Figure 2.9, we see that standard finetuning and distillation often need either more data or larger models to match LLM's performance. For instance, on e-SNLI in Figure 2.8, we observe that Distilling step-by-step outperform the LLM using only 0.1% of the dataset while standard finetuning requires more data to match the performance. Furthermore, on ANLI in Figure 2.8, we observe that Distilling step-by-step can outperform PaLM using 770M model with only 80% of the training set while standard finetuning struggles to match the LLM even using the full dataset and thus requires larger model to close the performance gap.

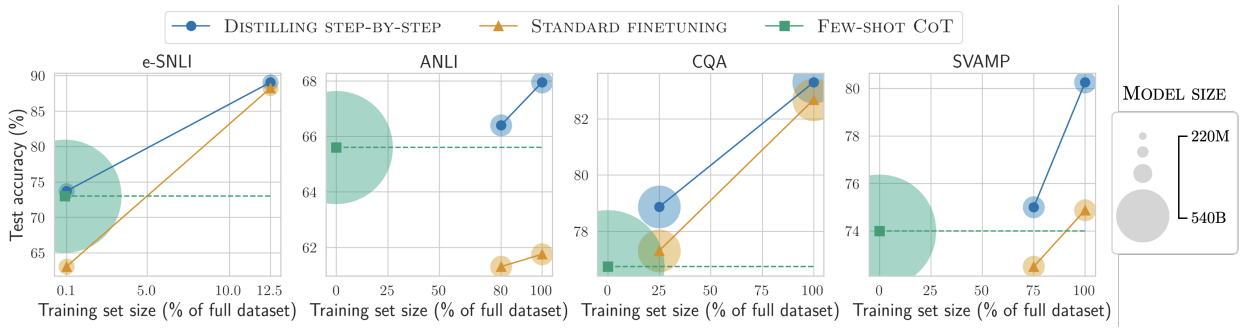


Figure 2.8: We show the minimum size of T5 models and the least amount of human-labeled examples required for Distilling step-by-step to outperform LLM's Few-shot CoT by a coarse-grained search. Distilling step-by-step is able to outperform Few-shot CoT using not only much smaller models, but it also achieves so with much less training examples compared to Standard finetuning. On ANLI, we outperform the LLM CoT with a 770M model using only 80% of the dataset, whereas Standard finetuning struggles to match even using 100% of the dataset.

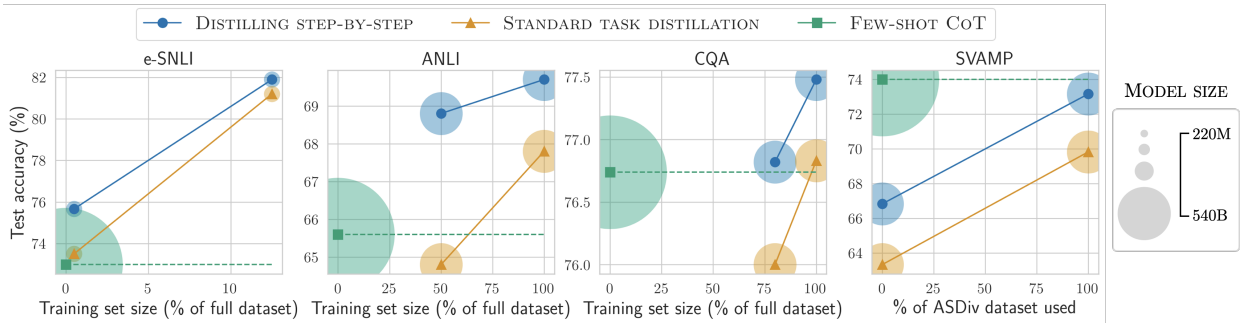


Figure 2.9: Similar to Figure 2.8 but using only unlabeled examples, Distilling step-by-step is able to outperform Few-shot CoT using much smaller models and with much less examples compared to Standard task distillation. On SVAMP, the x-axis corresponds to the size of ASDiv dataset used for augmenting the original SVAMP dataset, i.e., $x = 0$ is without augmentation and $x = 100$ corresponds to adding the full ASDiv dataset.

2.4.4 Further ablation studies

So far, we have focused on showing the effectiveness of Distilling step-by-step on reducing the training data required for finetuning or distilling smaller task-specific models. In this section, we perform further studies to understand the influence of different components in the Distilling step-by-step framework. Specifically, we study (1) how different LLMs, from which the rationales are extracted, affect the effectiveness of Distilling step-by-step, and (2) how the multi-task training approach compares to other potential design choices in training small task-specific models with LLM rationales. Here, we fix the small task-specific models to be 220M T5 models, and utilize 100% of the data on all datasets.

Table 2.1: Distilling step-by-step works with different sizes of LLMs. When rationales are extracted from a 20B GPT-NeoX model, Distilling step-by-step is still able to provide performance lift compared to standard finetuning on 220M T5 models.

		Dataset			
Method	LLM	e-SNLI	ANLI	CQA	SVAMP
STANDARD FINETUNING	N/A	88.38	43.58	62.19	62.63
DISTILLING STEP-BY-STEP	20B	89.12	48.15	63.25	63.00
DISTILLING STEP-BY-STEP	540B	89.51	49.58	63.29	65.50

Distilling step-by-step works with different sizes of decently trained LLMs. In addition to using 540B PaLM as the LLM, here we consider a relatively smaller LLM, 20B GPT-NeoX model [Black et al., 2022], from which we extract rationales for Distilling step-by-step. In Table 2.1, we see that when coupled with LLMs of different sizes, Distilling step-by-step can still provide performance improvements compared to standard finetuning. However, the performance lift is smaller when rationales are extracted from the 20B GPT-NeoX model instead of from the 540B PaLM. This can be due to the fact that the larger PaLM model provides higher-quality rationales that are more beneficial for learning the task.

Multi-task training is much more effective than single-task rationale and label joint prediction. There are different possible ways to train task-specific models with LLM-rationales as output supervisions. One straightforward approach is to concatenate the rationale $\hat{r}_i$ and label $\hat{y}_i$ into a single sequence $[\hat{r}_i, \hat{y}_i]$ and treat the entire sequence as the target output in training small models, as considered in [Magister et al., 2022; Ho et al., 2022]:

$$\mathcal{L}_{\text{single}} = \frac{1}{N}\sum_{i=1}^{N} \ell(f(x_i), [\hat{r}_i, \hat{y}_i]). \tag{2.5}$$

In Table 2.2, we compare this single-task training approach to our proposed multi-task training approach for utilizing LLM-rationales. We see that not only multi-task training consistently leads to better performance, single-task training with LLM-rationales can at times leads to worse performance than standard finetuning, e.g., on ANLI and CQA. In fact, similar results have also been observed in [Wiegreffe et al., 2021; Magister et al., 2022; Ho et al., 2022] that simply treating rationale and label predictions as a single joint task may harm the model's performance on label prediction. This validates our use of the multi-task training approach,

Table 2.2: Our proposed multi-task training framework consistently leads to better performances than treating rationale and label predictions as a single task. Single-task training can at times lead to worse performance than standard finetuning.

	Dataset			
Method	e-SNLI	ANLI	CQA	SVAMP
STANDARD FINETUNING	88.38	43.58	62.19	62.63
SINGLE-TASK TRAINING	88.88	43.50	61.37	63.00
MULTI-TASK TRAINING	**89.51**	**49.58**	**63.29**	**65.50**

and highlights the need to treat the rationales carefully so as to unleash their actual benefits.

2.5 Discussion

We propose Distilling step-by-step to extract rationales from LLMs as informative supervision in training small task-specific models. We show that Distilling step-by-step reduces the training dataset required to curate task-specific smaller models; it also reduces the model size required to achieve, and even surpass, the original LLM's performance. Distilling step-by-step proposes a resource-efficient training-to-deployment paradigm compared to existing methods. Further studies demonstrate the generalizability and the design choices made in Distilling step-by-step. Finally, we discuss the limitations, future directions and ethics statement of this chapter below.

2.6 Limitations

There are a number of limitations with our approach. First, we require users to produce a few example demonstrations ($\sim$ 10-shot for all tasks) in order to use the few-shot CoT [Wei et al., 2022] prompting mechanism. This limitation can be improved by using recent advances that suggest that rationales can be elicited without any user-annotated demonstrations [Kojima et al., 2022]. Second, training task-specific models with rationales incur slight training-time computation overhead. However, at test time, our multi-task design naturally avoids the computation overhead since it allows one to only predict labels without generating the rationales. Finally, while we observe success using LLM rationales, there is evidence that LLMs exhibit limited reasoning capability on more complex reasoning and planning tasks [Valmeekam et al., 2022]. Future

work should characterize how rationale quality affects Distilling step-by-step.

2.7 Ethics statement

It is worth noting that the behavior of the our downstream smaller models is subject to biases inherited from the larger teacher LLM. We envision that the same research progress in reducing anti-social behaviors in LLMs can also be applied to improve smaller language models.

Chapter 3

Effective Language Model Adaptation: Calibrating Positional Attention Bias Improves Long Context Utilization

3.1 Introduction

Effective prompting of large language models (LLMs) [Brown et al., 2020; Anil et al., 2023; Touvron et al., 2023] has enabled a variety of user-facing applications, including conversational interfaces (chatbots) [Thoppilan et al., 2022], search and summarization [Min et al., 2024], open-domain question answering [Izacard and Grave, 2021], tool usage [Hsieh et al., 2023a], fact checking [Asai et al., 2023], and collaborative writing [Lee et al., 2019]. Some of these applications, such as search and summarization [Ji et al., 2023; Min et al., 2023; Asai et al., 2023], require the ability to retrieve information from external knowledge sources. As a result, retrieval-augmented generation (RAG) has become a powerful solution. RAG fetches relevant documents (e.g. structured tables [Wang et al., 2024b] and API documentation [Karpukhin et al., 2020]) from external knowledge sources and makes them available in the LLMs' input prompt [Khandelwal et al., 2020; Borgeaud et al., 2022; Izacard et al., 2022b; Xu et al., 2023b]. Despite the widespread utility of RAG [Li et al., 2023a; Xiong et al., 2023; OpenAI, 2022; Gemini Team, 2023], recent experiments highlight a striking deficiency: LLMs struggle to locate relevant documents when they are placed in the middle of their input prompts [Liu

et al., 2023b; Li et al., 2023a]. They call this the *lost-in-the-middle* phenomenon.

To overcome this phenomenon, a few mechanistic strategies have been proposed [Jiang et al., 2023; Peysakhovich and Lerer, 2023]. These methods *re-rank* the relevance of different documents and *re-order* the most relevant ones to either the beginning or end of the input context. Unfortunately, re-ranking usually requires additional supervision or dedicated finetuning for performant RAG performance [Karpukhin et al., 2020; Shi et al., 2023c; Sun et al., 2023b]. Worse, re-ranking methods do not fundamentally improve LLMs' ability to utilize and capture relevant information from the provided input contexts. The underlying causes of this behavior remains unclear, even though it has been observed across multiple decoder-only LLMs [Touvron et al., 2023; Li et al., 2023a; OpenAI, 2022].

In this chapter, we make three contributions: First, we set out to understand the potential factors leading to the *lost-in-the-middle* problem. **We establish a connection between lost-in-the-middle to LLMs' intrinsic attention bias** (see Figure 3.1). Specifically, we find that models often demonstrate a *U-shaped* attention distributions, with higher attention values assigned to the beginning and end of the input prompt. This correlates well with the U-shaped RAG performance observed in prior literature [Liu et al., 2023b]. Interestingly, this focus on the beginning and end also extends to content utilization: models preferentially use information from the beginning and end of their prompts [Ravaut et al., 2023; Peysakhovich and Lerer, 2023]. This leads us to hypothesize that the positional attention bias may contribute to the phenomenon, wherein the bias could lead to over-reliance on content at the beginning/end of the input, regardless of its true relevance.

Second, we verify our hypothesis by intervening on this attention bias to determine its impact on performance. **We propose a mechanism to disentangle positional bias from model's attention.** We first esitmate this bias through measuring the change in attention as we vary the relative position of a fixed context in the LLM's prompt. By quantifying and then removing this bias from the attention scores for a given query, we can obtain the *calibrated attention* scores across the retrieved documents. This calibrated attention proves to be better correlated to the ground truth relevance of the document to a user query. In open-domain question answering tasks [Kwiatkowski et al., 2019], our proposed calibrated attention outperforms popular existing approaches for ranking the relevance of retrieved documents (up to 48 Recall@3 points). This finding challenges the recent belief that LLMs struggle to capture relevant context embedded in the middle of inputs,

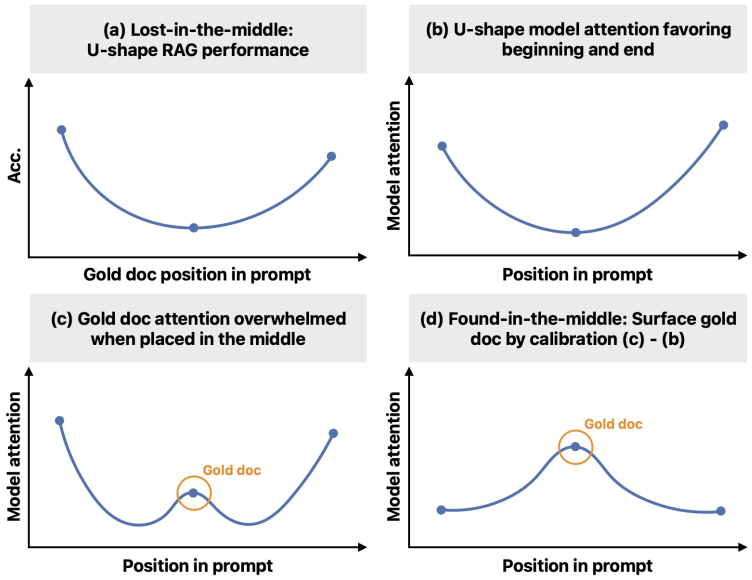


Figure 3.1: (a) Lost-in-the-middle refers to models' U-shape RAG performance as the relevant context's (e.g., a gold document containing the answer to a query) position varies within the input; (b) We observe models exhibit U-shape attention weights favoring leading and ending contexts, regardless of their actual contents; (c) Models do attend to relevant contexts even when placed in the middle, but are eventually distracted by leading/ending contexts; (d) We propose a calibration mechanism, found-in-the-middle, that disentangles the effect of U-shape attention bias and allows models to attend to relevant context regardless their positions.

suggesting they may indeed be capable of doing so, but are only hindered by the overwhelming positional bias.

Third, we operationalize our calibration mechanism as a solution for this phenomenon, naming our attention intervention *found-in-the-middle*. **We show that calibrating the attention leads to improvements across two popular LLMs with different context window lengths on two RAG tasks.** Our experiments demonstrate improvements over standard model generation by up to 15 percentage point on NaturalQuestion dataset [Kwiatkowski et al., 2019]. We hope our findings in this chapter opens up future directions in understanding LLM's attention biases and their effect on downstream tasks.

3.2 Positional attention bias overpowers mid-sequence context

Recent work has produced language models capable of handling increasingly long input contexts [Xiong et al., 2023; Li et al., 2023a]. However, many of these models struggle to locate relevant information placed in the middle of the input sequence [Liu et al., 2023b], a phenomenon known as the "lost-in-the-middle"

problem. While this problem is widely recognized, the potential factors contributing to this behavior remain poorly understood. In this chapter, we seek to deepen our understanding of the problem through a suite of exploratory qualitative and quantitative studies.

Setup. We adhere to the original experimental setup outlined in Liu et al. [2023b], utilizing an open-domain question answering task [Kwiatkowski et al., 2019] for our exploratory study. In the lost-in-the-middle setup [Liu et al., 2023b], a model is tasked to answer a user query x^{q} using a set of k related documents retrieved from an external data source $D = \{x^{\mathrm{gold}}, x_1^{\mathrm{distract}}, \ldots, x_{k-1}^{\mathrm{distract}}\}$, where only the gold document x^{gold} contains the correct answer. The question and documents are typically serialized as an input sequence $x^{\mathrm{prompt}} = [x^{\mathrm{q}}, x_1^{\mathrm{doc}}, ..., x_k^{\mathrm{doc}}, x^{\mathrm{q}}]$, prompting a language model to generate the final answer[1]. Observations indicate that model performance significantly decreases when x^{gold} is placed within the middle of the input prompt (i.e., $x_{\lfloor k/2 \rfloor}^{\mathrm{doc}}$), compared to scenarios where x^{gold} is placed at the beginning or end. Here, we reproduce lost-in-the-middle phenomenon with a `Vicuna-7b-v1.5-16k` (Vicuna) model [Li et al., 2023a] to gain deeper insights into the characteristics of the model's errors. We focus our error analysis on the setting where we have a total of 20 documents ($K = 20$). We specifically look at the examples where the model makes incorrect predictions when the gold document is placed at the middle (10-th) position.

3.2.1 U-shaped attention bias

We first examine responses generated when gold documents are placed in the **middle** of input prompts. Qualitatively, the model's response exhibits a strong bias towards the document at the first position, regardless of the gold document's location (Figure 3.2). This bias persists whether the input documents retain their original order or are randomly shuffled.

The strong correlation between the model's output and the first document could suggest that they are highly relevant, distracting the model [Shi et al., 2023a]. However, quantitatively, the model's response strongly depends on the document at the first position (Figure 3.3). This dependence persists even after randomly shuffling the document order, irrespective of its relevance to the query. We measure the dependence by computing the TF-IDF similarity between the response and each document (gold document originally at

[1]We repeat the question before and after the documents so that the model can better attend to relevant contexts [Liu et al., 2023b; Xu et al., 2023c].

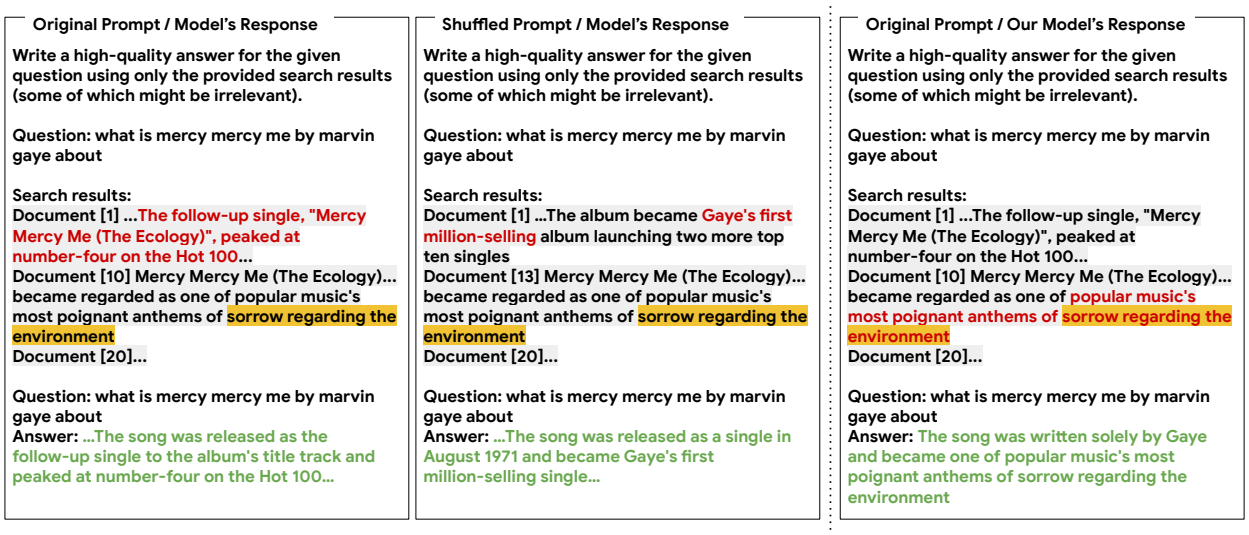


Figure 3.2: Left and Middle: Qualitatively, the model's response exhibits a strong bias towards the document at the first position (red). This persists whether the input documents retain their original order (left: gold document at the 10th position) or are randomly shuffled (middle: gold document at the 13th position). Model responses are shown in green, with the gold answer highlighted in yellow. **Right: Our attention calibration method enables the model to find relevant context even when placed in the middle.**

position 10).

To investigate the potential origins of positional bias, we visualize the model's self-attention weights, as the weights has been shown to correlate with models' generations, although not necessarily causal [Dong et al., 2021; Zhang et al., 2023]. More formally, given an input prompt consisting of K documents $x^{\text{prompt}} = [x_1^{\text{doc}}, ..., x_K^{\text{doc}}]$, where each document $x_k^{\text{doc}} = \{x_{k,i}^{\text{doc}}\}_{i=1}^{N_k}$ contains N_k tokens, let $\text{Attn} : \mathcal{X} \times \mathbb{N} \rightarrow \mathbb{R}$ denote a function that computes the average attention weights assigned to document x_k^{doc} as $\text{Attn}(x^{\text{prompt}}, k) = \sum_{i=1}^{N_k} \text{attn}(x_{k,i}^{\text{doc}})/N_k$, where $\text{attn}(x_{k,i}^{\text{doc}})$ is the attention weight value allocated to token $x_{k,i}^{\text{doc}}$ when predicting the next $|x^{\text{prompt}}| + 1$ token.

Specifically, we visualize the self-attention weights assigned to each document, averaged across all its tokens, all decoder layers, and heads. We investigate how these weights vary based on document position within the input prompt. Interestingly, Figure 3.4 (blue curve) reveals a U-shaped attention pattern. Documents near the beginning and end of the input receive higher weights, while those in the middle receive lower weights. Crucially, the U-shaped pattern persists even after randomly shuffling document order (Figure 3.4, orange curve), suggesting that this bias does not depend on the documents' actual content.

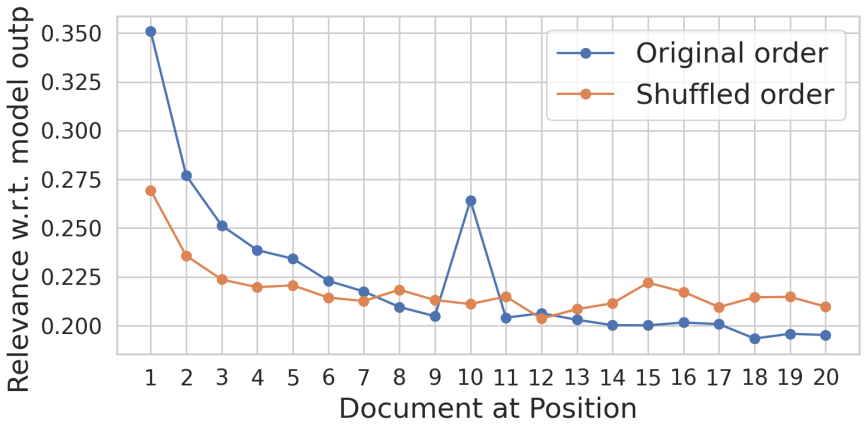


Figure 3.3: Quantitatively, the model's response strongly depends on the document at the first position. This dependence persists even after randomly shuffling the document order, irrespective of its relevance to the query. We measure this dependence by computing the TF-IDF similarity score between the response and each document (gold document originally at position 10).

3.2.2 Does attention favor relevant context?

Observation 1: Model prioritizes relevant contexts from the same position. In Figure 3.4, we observe a significant difference in attention values at x_{10}^{doc} when comparing examples with original document order (blue) and randomly shuffled order (orange). Specifically, the attention value is notably higher when when x_{10}^{doc} is controlled to be x^{gold}. This contrasts with instances where x_{10}^{doc} is uncontrolled, suggesting that apart from U-shaped positional bias, the model exhibits an ability to *prioritize* relevant context.

Observation 2: Model prioritizes highly-weighted documents for generation. Based on these observations, we hypothesize that positional attention bias significantly influence the model's tendency to rely heavily on the first documents during output generation. Specifically, the models are more likely to incorporate the document receiving the highest attention (often the first) into its output. To validate this, for each of the examples of interest (where the model makes incorrect predictions), we divide their documents into first half receiving higher model attention and second half receiving lower attention. We then count the number of examples in which the first or second half contains the document that is most likely used in the model's generation (i.e., having the highest TF-IDF score with model's response). In Table 3.1, we show that documents receiving higher attention positively correlates with them being used in the model's generation.

From the above studies, we see that not only the model exhibits a U-shape positional attention bias, but this

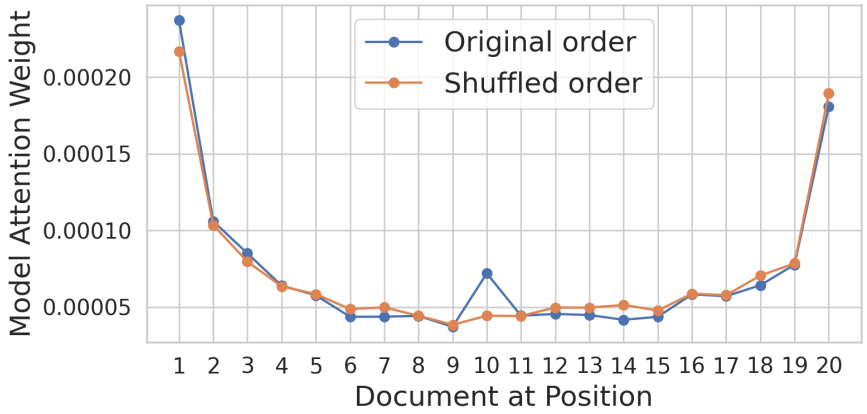


Figure 3.4: Average attention weights reveal a U-shaped positional bias in the model. Documents at the beginning and end receive greater attention, regardless of order (gold document originally at position 10). Attention is averaged across different decoder layers and attention heads.

Table 3.1: Number of examples where the most likely used document in the model's generation falls within the first half of documents receiving higher model attention or second half receiving lower attention. We see that there is a strong correlation where documents receiving higher attention are more likely to be used in model's response.

	Most Likely Used	
	# of examples	%
Highest Half Attention	526	74%
Lowest Half Attention	186	26%

bias also correlates strongly with the model's biased tendency in using documents placed at certain positions in forming its response. We thus conjecture that lost-in-the-middle happens because of the dominating force of positional bias.

3.3 Found-in-the-middle: modeling and isolating positional attention bias

Ideally, a model should leverage contexts in the input prompts—faithfully according to their relevance—for generating the response, instead of biasing towards contexts placed at certain positions within the input. Towards this goal, we are interested in modeling the positional attention bias and mitigating it such that model attention can reflect the true relevance of the input context and ultimately improve models' effective utilization of the full context window.

3.3.1 Two main factors in model attention

In Sec. 3.2, we find that there are two main forces driving the model attention assigned to different documents of an input prompt: (a) where the document locates within the entire input, and (b) the relevance of the document.

Our hypothesis. We thus consider modeling the observable attention weights allocated to the k-th document of an input x^{prompt} as:

$$\text{Attn}(x^{\text{prompt}}, k) = f(\text{rel}(x_k^{\text{doc}}), \text{bias}(k)), \tag{3.1}$$

where $\text{rel}(\cdot)$ measures the relevance of an input document, $\text{bias}(\cdot)$ characterizes the positional attention bias, and $f(\cdot)$ is some unknown monotonically increasing function w.r.t. to both $\text{rel}(x_k^{\text{doc}})$ and $\text{bias}(k)$. For ease of exposition, in the remainder of the paper, we overload $\text{Attn}(x^{\text{doc}}, k)$ to denote the attention value assigned to document x_k^{doc} placed at the k-th position within an input prompt containing K documents.

Corroborating our assumed model. Here, we conduct a suite of controlled experiments using NaturalQuestion with $K = 20$ and a `Vicuna-7b-v1.5-16k` model to corroborate our assumed model. Specifically, for Eq. 3.1 to hold, it implies that:

Condition 1: When the relevance term is fixed, model attention increases as positional bias increases. That is, given two documents x^{doc1} and x^{doc2}: *if* $\text{Attn}(x^{\text{doc1}}, k) > \text{Attn}(x^{\text{doc1}}, l)$*, then* $\text{Attn}(x^{\text{doc2}}, k) > \text{Attn}(x^{\text{doc2}}, l)$.

Condition 2: Similarly, when the document position k is fixed, model attention increases as the relevance of the document increase: *if* $\text{Attn}(x^{\text{doc1}}, k) > \text{Attn}(x^{\text{doc2}}, k)$*, then* $\text{Attn}(x^{\text{doc1}}, l) > \text{Attn}(x^{\text{doc2}}, l)$.

We validate Condition 1 and 2 on 100 randomly sampled examples from NaturalQuestion dataset, each with $K = 20$ documents. For validating Condition 1, given a pair of documents $(x^{\text{doc1}}, x^{\text{doc2}})$ and positions (k, l), we can compute whether the relationship holds across all possible pairs. We can similarly test for Condition 2. In Table 3.2, we see that the percentage of valid example pairs are decently high, 83% and 72% respectively, for both conditions, providing supports to our hypothesis.

Recall that our goal is to disentangle positional attention bias from model attention such that the model can faithfully attend to relevant contexts, independent from their positions. So far, while we have established

Table 3.2: High correlations between model attention with document relevance and positional bias supports our hypothesized model.

Hypothesis test	$\mathrm{rel}(x^{\mathrm{doc}})$	$\mathrm{bias}(k)$	% of valid pairs
Condition 1	Fixed	Varying	83%
Condition 2	Varying	Fixed	72%

the monotonic increasing nature of f in Eq. 3.1, we have yet characterize the actual form of f to remove the positional bias term from model attention.

To approximate f, we consider simple linear models by following machine learning principles (a.k.a. Occam's razor), for robust estimation:

$$\mathrm{Attn}(x^{\mathrm{doc}}, k) = \mathrm{rel}(x^{\mathrm{doc}}) + \mathrm{bias}(k) + \epsilon, \tag{3.2}$$

where ϵ is a noise.

To test how the model captures the underlying relationship, we compute Spearman's rank correlation between $\mathrm{Attn}(x^{\mathrm{doc1}}, k) - \mathrm{Attn}(x^{\mathrm{doc2}}, k)$ and $\mathrm{Attn}(x^{\mathrm{doc1}}, l) - \mathrm{Attn}(x^{\mathrm{doc2}}, l))$ over quadruplets of $(x^{\mathrm{doc1}}, x^{\mathrm{doc2}}, k, l)$ collected from NaturalQuestion. A high correlation indicates small discrepancy between $\mathrm{Attn}(x^{\mathrm{doc1}}, k) - \mathrm{Attn}(x^{\mathrm{doc2}}, k)$ and $\mathrm{Attn}(x^{\mathrm{doc1}}, l) - \mathrm{Attn}(x^{\mathrm{doc2}}, l))$. From our study, the linear model results in decently high correlation, 0.76, suggesting its effectiveness despite the simplicity. We therefore adopt Eq. 3.2 as our model and leave other alternatives with more degree of freedoms as future work [2].

3.3.2 Disentangling positional attention bias

Most notably, having a simple form of f allows us to isolate the effect of positional bias from model attention. Specifically, following from Eq. 3.2, we can first obtain a reference model attention value with a dummy document x^{dum} by:

$$\mathrm{Attn}(x^{\mathrm{dum}}, k) = \mathrm{rel}(x^{\mathrm{dum}}) + \mathrm{bias}(k) + \epsilon. \tag{3.3}$$

[2] In Appendix B.3, we also explore log-linear models, which results in competitive 0.75 rank correlation.

By subtracting Eq. 3.2 and Eq. 3.3, we can offset the bias term and obtain:

$$\begin{aligned} &\mathrm{rel}(x^{\mathrm{doc}}) \\ &= \mathrm{Attn}(x^{\mathrm{doc}}, k) - \mathrm{Attn}(x^{\mathrm{dum}}, k) + \mathrm{rel}(x^{\mathrm{dum}}) \end{aligned} \tag{3.4}$$

Consider using a consistent dummy document x^{dum} which has a constant $\mathrm{rel}(x^{\mathrm{dum}})$, we are then able to obtain the true relevance of different documents x^{doc}, free from the positional bias. We refer to $\mathrm{Attn}(x^{\mathrm{doc}}, k) - \mathrm{Attn}(x^{\mathrm{dum}}, k)$ as *calibrated attention* as it removes the baseline attention, and call the overall calibration mechanism *found-in-the-middle*.

Calibrated attention finds relevant contexts in the middle. Eq. 3.4 allows us to leverage calibrated attention to estimate and rank the relevance of different documents within an input prompt. To validate the effectiveness of our model, we evaluate using calibrated attention to re-rank documents in an input prompt w.r.t. a given query. We evaluate on NaturalQuestion with the Vicuna model where we focus on the most challenging setting when the gold document in placed in the middle of the input prompt. We compare our model to:

- Vanilla attention: Using uncalibrated attention $\mathrm{Attn}(x^{\mathrm{prompt}}, k)$ to rank the documents.
- Query generation [Sun et al., 2023b]: Using likelihood of the model in generating the query based on the document.
- Relevance generation [Sun et al., 2023b]: Prompting the model to answer whether a document is relevant to a query.

In Table 3.3, we compare Recall@3 of different methods where we vary the total number of documents retrieved. We see that the proposed calibrated attention consistently outperforms vanilla attention by a large margin, and also shows superior performances when compared to the other two re-ranking metrics. The results validate that our proposed modeling approach is effective, and that if calibrated appropriately, language models can locate relevant information even when they are hidden in the middle of the input.

Table 3.3: Calibrated attention outperforms existing methods in ranking the relevance of retrieved contexts given a user query. We report Recall@3 on NaturalQuestion when gold documents are placed in the middle of input context.

	Number of total documents	
Method	$K = 10$	$K = 20$
Vanilla attention	0.3638	0.2052
Query generation	0.6851	0.5815
Relevance generation	0.5521	0.4012
Calibrated attention	**0.7427**	**0.6832**

3.4 Improving long-context utilization with found-in-the-middle

Having validated that calibrated attention through found-in-the-middle is effective in locating relevant information within a long input context, we are ultimately interested in leveraging it to tackle lost-in-the-middle problem and practically improve a model's RAG performance.

3.4.1 Attention calibration

To allow the model to attend to contexts without being dictated by positional bias, we propose to intervene the model's attention based on the proposed calibrated attention. Specifically, given an input x^{prompt}, instead of allocating $\text{rel}(x_k^{\text{doc}}) + \text{bias}(k)$ attention to the k-th document, our ideal model attention $\text{Attn}_{\text{calibrated}}(x_k^{\text{doc}})$ would reflect only the relevance of the context $\text{rel}(x_k^{\text{doc}})$.

To achieve this, we propose to redistribute the attention values assigned to $\{x_k^{\text{doc}}\}_{k=1}^K$ according to $\text{rel}(x_k^{\text{doc}})$. Specifically, for each document x_k^{doc}, we propose to rescale the attention values on the tokens within the document, $\{x_{k,i}^{\text{doc}}\}_{i=1}^{N_k}$, by:

$$\text{attn}_{\text{calibrated}}(x_{k,i}^{\text{doc}}) = \frac{\alpha_k}{\text{Attn}_{\text{original}}(x_k^{\text{doc}})} \cdot \text{attn}_{\text{original}}(x_{k,i}^{\text{doc}}) \cdot C, \tag{3.5}$$

where $\alpha_k = \text{Softmax}(\text{rel}(x_k^{\text{doc}}), t)$, t is the temperature hyperparamter, and C is a normalization constant to ensure the total attention $\sum_{k,i} x_{k,i}^{\text{doc}}$ remains unchanged. With the rescaling, we effectively make the final attention on x_k^{doc}:

$$\text{Attn}_{\text{calibrated}}(x_k^{\text{doc}}) \propto \text{Softmax}(\text{rel}(x_k^{\text{doc}}), t), \tag{3.6}$$

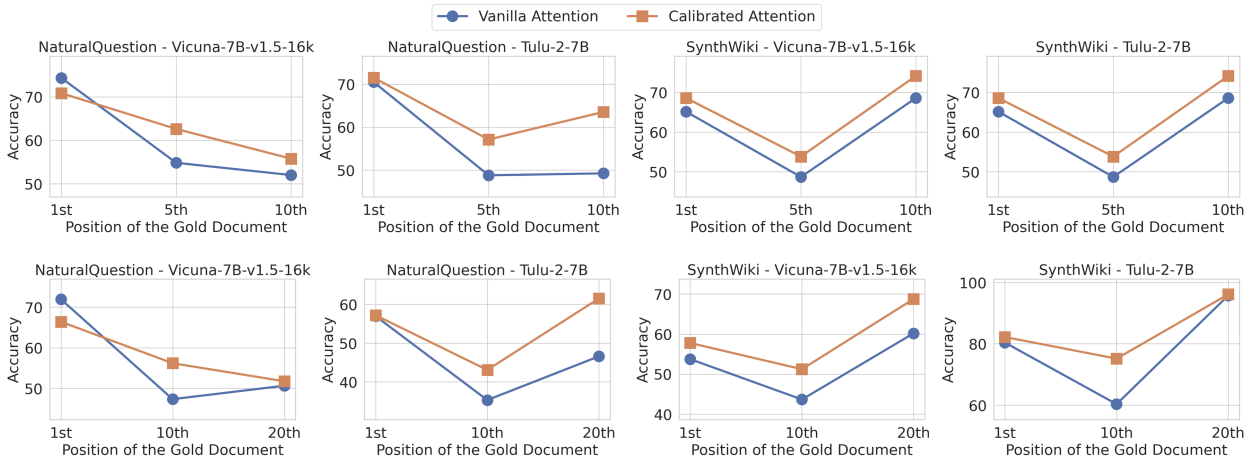


Figure 3.5: Attention calibration effectively improves models' context utilization ability, with its performance curves lying almost entirely above standard vanilla attention (on 22 out of 24 cases). On the most challenging settings where the gold documents are placed in the middle, attention calibration provides 6-15 points improvements. Top/Bottom row: 10/20-doc. Numbers shown in Table B.2.

where higher attention is allocated to more relevant context, and t controls the disparity level.

3.4.2 Calibrated v.s. uncalibrated attention

We evaluate the performance of the proposed attention calibration method. We conduct experiments on two multi-document question answering tasks (more details in Appendix B.1), NaturalQuestion [Kwiatkowski et al., 2019] and SynthWiki [Peysakhovich and Lerer, 2023], with two models supporting different context window length: `Vicuna-7b-v1.5-16k` (Vicuna) [Li et al., 2023a] and `tulu-2-7b` (Tulu) [Wang et al., 2023] with 16k and 8k context window respectively. For each dataset, we consider two settings with different number of retrieved documents, $K = \{10, 20\}$. We leave further implementation details in Appendix B.2.

Found-in-the-middle improves long-context utilization across various datasets and models. In Figure 3.5, we see that found-in-the-middle attention calibration consistently outperforms the uncalibrated baseline by a large margin (up to 15 percentage point (pp) improvement) across different tasks and models. On the most challenging scenario when the gold document is placed mid-sequence, attention calibration consistently offers improvements from 6-15 pp. Notably, we see that attention calibration's performance curve lies almost entirely above the vanilla baseline curve (except 2 out of 24 cases), validating the effectiveness of our method in improving models' long context utilization.

3.4.3 Attention calibration in practice

In practice, to avoid the lost-in-the-middle effect, one commonly adopted workaround is to reorder the document positions, where documents considered more relevant are placed towards the beginning (or end) of the input. While these methods have led to performance improvements over the baseline without reordering, without handling the model's intrinsic bias, reordering-based methods' performance relies heavily on the correct ranking of the documents. We are thus interested in validating whether attention calibration can be applied on top of re-ordering methods to provide another layer of improvements.

Attention calibration improves existing RAG pipelines. We continue using NaturalQuestion and SynthWiki for evaluation. We compare to existing reordering methods including:

- Prompt reordering [Sun et al., 2023b; Liang et al., 2023]: Reorder documents based on relevance score generated through prompting.

- LongLLMLingua-r_k [Jiang et al., 2023]: Reorder documents using query generation as the reranking metric.

- Attention sorting [Peysakhovich and Lerer, 2023]: Reorder documents using vanilla model attention assigned to the documents.

In Figure 3.6, we note that LongLLMLingua-r_k and prompt reodering are invariant to the gold document's position since they compute the relevance of each document independently. First, we see that reordering methods do alleviate lost-in-the-middle problem where models' performances increase when gold documents is placed mid-sequence. More importantly, we see that by applying attention calibration on top of a reordering mechanism (LongLLMLingua-$r_k$in this case), LongLLMLingua-r_k with calibration consistently achieve the highest performance across datasets and models. These results suggest that attention calibration can more fundamentally improve models' context utilization, providing a complementary way to re-ordering methods to further improve current RAG pipeline.

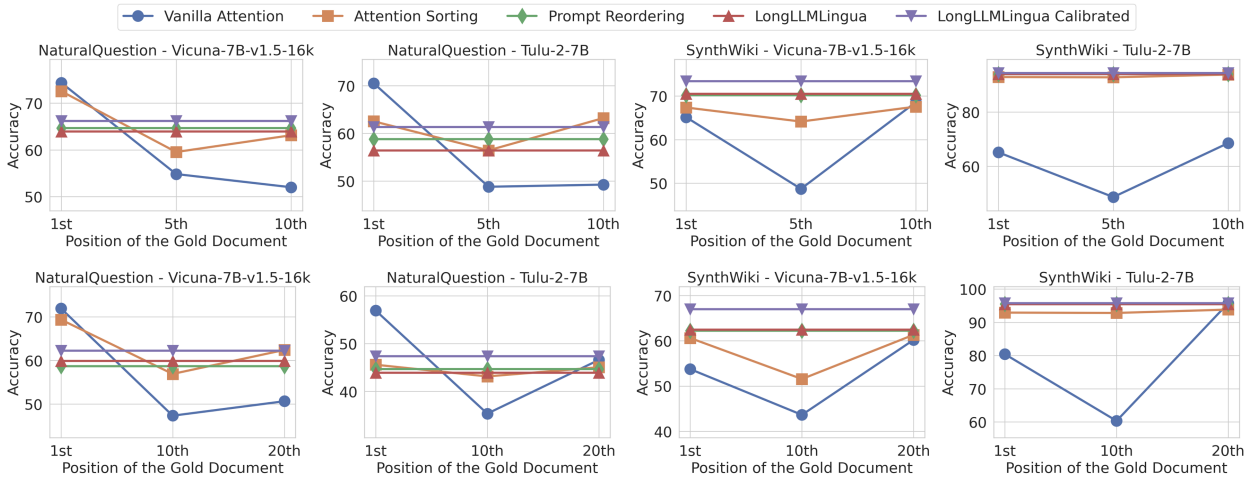


Figure 3.6: Attention calibration can be applied on top of reordering-based methods to provide further performance boost. This suggests that mitigating attention bias can more fundamentally improve models' context utilization, offering a complementary way to further improve existing RAG pipeline. Top/Bottom row: 10/20-doc. Numbers shown in Table B.2.

3.5 Related work

Retrieval augmented generation. While LLMs exhibit strong capabilities [Gemini Team, 2023; OpenAI, 2022; Touvron et al., 2023], their knowledge is inherently limited in its pretraining data, and they are observed to struggle in handling knowledge intensive tasks [Petroni et al., 2020]. To tackle this, retrieval augmented generation (RAG) is an effective framework that retrieves relevant information from external knowledge sources to aid and ground language models' generation [Lewis et al., 2020; Khandelwal et al., 2020; Borgeaud et al., 2021; Izacard and Grave, 2021; Izacard et al., 2022b].

Although RAG has powered many recent language model applications from question-answering [Izacard and Grave, 2021] to automatic task completion [Shen et al., 2023], recent work show that LLMs tend to *lost-in-the-middle*, significantly hindering the full potential of RAG [Liu et al., 2023b]. In this chapter, we take a step further to understand the lost-in-the-middle problem from the viewpoint of attention bias. Moreover, we propose a remedy through attention calibration, which improves upon existing RAG frameworks.

Long-context utilization in language models. There is a rich literature on enabling LLMs to handle longer input contexts, including designing efficient training and finetuning schemes [Dao et al., 2022; Li et al., 2023b,a; Shi et al., 2023b] and inference-time methods that extend an LLM's context length [Press et al.,

2021; Ratner et al., 2023; Xiao et al., 2023; Bertsch et al., 2023]. Nonetheless, even models specifically trained for long-context suffer lost-in-the-middle problem [Liu et al., 2023b; Li et al., 2023a].

To improve LLMs' performance on handling long contexts, recent methods design better prompting techniques and pipelines that mechanically work around the lost-in-the-middle problem [Chen et al., 2023a; Jiang et al., 2023; Peysakhovich and Lerer, 2023; Junqing et al., 2023]. For instance, to avoid having the models process long input contexts, [Chen et al., 2023a; Junqing et al., 2023] proposes to split long inputs into shorter contexts for models to better understand. To avoid relevant context being missed by the model, [Jiang et al., 2023; Peysakhovich and Lerer, 2023] proposes to rank the relevance of different parts of the input and re-order the most important parts to either the beginning or end of the entire input, where the models tend to focus more.

While these existing solutions lead to improved model performances by manipulating the input contexts, they do not fundamentally improve LLMs' underlying long-context utilization capability. In contrast, we set out to directly improve LLMs' long-context utilization capability to mitigate lost-in-the-middle problem.

Self-attention and attention bias. The attention mechanism is initially introduced in RNN-based encoder-decoder architectures [Bahdanau et al., 2015; Luong et al., 2015]. Building upon the self-attention mechanism, transformers [Vaswani et al., 2017] have achieved state-of-the-art performance in various domains [Devlin et al., 2018; Dosovitskiy, 2020]. Self-attention has also been widely used as a proxy to understand and explain model behaviors [Clark et al., 2019; Hao et al., 2021; Vashishth et al., 2019].

However, the relationship between the lost-in-the-middle problem and LLM's self-attention has been under-explored. As an initial trial, "attention sorting" [Peysakhovich and Lerer, 2023] sorts documents multiple times by the attention they receive to counter lost-in-the-middle. Recently, He et al. [2023] construct a dataset for training LLMs to focus on the most relevant documents among long contexts. Unlike the method, which necessitate significant investment in data collection and LLM tuning, our method offers an efficient solution by mitigating lost-in-the-middle problem with off-the-shelf LLMs.

3.6 Discussion

In this chapter, we understand and address the lost-in-the-middle phenomenon, by establishing a connection between the phenomenon and models' positional attention bias. We mitigate the bias by attention calibration which directly modifies the model's attention mechanism, enabling LLMs to more faithfully attend to contexts based on their relevance, rather than their position. Experiments show that attention calibration improves the performance compared to its uncalibrated counterpart especially when relevant context occurs in the middle of the input. We additionally show attention calibration can be applied on top of existing reordering pipelines to further improve models' performance.

3.7 Limitations

While our study presents significant advances in addressing the "lost-in-the-middle" problem and improving RAG performance in LLMs, several limitations are noteworthy:

Simplification of the mechanism behind positional attention bias. We proposed a simple hypothesis to model the positional attention bias, as shown in Eq. 3.1. However, the intrinsic mechanisms that drive this bias could be more intricate and dynamic than our current model accounts for. It is possible that some aspects of attention bias are learnable or adaptive, responding to subtle aspects of the data or training process that our current approach does not consider.

Computational overhead. Our method of calibrating positional attention bias, while effective, introduces additional computational overhead. Specifically, we require extra $O(K)$ model forward passes to calibrate attention at each position, compared to vanilla model generation. However, in this study we aim to discover and calibrate the positional attention bias from a scientific perspective. We expect that our discovery can enable future research into developing more calibration methods with lower computational overhead.

Positional attention bias may be beneficial. Our method aims to completely remove positional attention bias. However, it is important to note that this positional bias might actually be beneficial in certain contexts. In some specific tasks or scenarios, the natural tendency of models to focus more on the beginning and end of

inputs could align well with the structure of the task or the nature of the data. Therefore, understanding the tasks and the applications is required before adopting our proposed calibration method.

The root cause of attention bias is unclear. In this chapter, we aim to discover and understand the connection between the lost-in-the-middle problem and LLMs' intrinsic attention bias. However, we have yet definitively pinpoint the root cause of attention bias in LLMs. The cause of such a bias could be attributed to the distribution of pretraining corpora, the transformer model architecture, and the optimization process. Future research needs to delve deeper into the origins of this phenomenon.

3.8 Ethical Statement

In our research, we focus on enhancing the performance of large language models using existing public datasets, ensuring that no personal or sensitive data was collected or utilized. Our attention calibration method is aimed at improving the efficiency and accuracy of retrieval-augmented generation, with potential benefits across various domains including search engines, question-answering systems, and other text-based applications. It is important to acknowledge that as our technique builds upon pre-trained language models, it may inadvertently inherit and propagate existing biases inherent in these models. Apart from this significant concern, we do not identify any other immediate risks arising from the methodologies or findings presented in this chapter.

Chapter 4

Data for Reliable Vision-Language Model Development: Fixing Hackable Benchmarks for Vision-Language Compositionality

4.1 Introduction

Scholars today herald *compositionality* as a fundamental presupposition characterizing both human perception and linguistic processing [Cresswell, 1973]. Through compositional reasoning, humans can comprehend new scenes and describe those scenes by composing known atoms [Janssen and Partee, 1997; Hupkes et al., 2020; Bottou, 2014; Chomsky and Halle, 1965]. For instance, compositionality allows people to differentiate between a photo of "a girl in white facing a man in black" and "a girl in black facing a man in white". For a while now, vision-language research has sought to develop models that can similarly comprehend scenes and express them through compositional language [Krishna et al., 2017; Ji et al., 2020; Lu et al., 2016; Grunde-McLaughlin et al., 2021].

Given its importance, a surge of new benchmarks have been proposed to evaluate whether vision-language models exhibit compositionality. Recently, Winoground [Thrush et al., 2022], VL-CheckList [Zhao et al., 2022], ARO [Yuksekgonul et al., 2023], CREPE [Ma et al., 2022], and Cola [Ray et al., 2023] have entered the machine learning zeitgeist. Evaluation is mostly done through an image-to-text retrieval task

formulation [Zhao et al., 2022; Yuksekgonul et al., 2023; Ma et al., 2022]: by measuring how often models pick the description, "a girl in white facing a man in black" when presented with an image of it, and avoid choosing the incorrect *hard negative* description, "a girl in black facing a man in white".

In this chapter, we uncover a crucial vulnerability in not just one but all these image-to-text compositionality benchmarks: We find that a *blind* model that never looks at the image, can identify the correct caption and avoid choosing the supposed "hard negatives". This blind model outperforms a wide array of pretrained vision-language models across the suite of benchmarks [Radford et al., 2021a; Ilharco et al., 2021; Gadre et al., 2023]. We explain this undesired hackability in existing benchmarks by showcasing that there exists a significant distributional gap between the positive and hard negative captions. For instance, in the ARO benchmark [Yuksekgonul et al., 2023], human-generated positive captions differ drastically from the hard negative texts generated by randomly shuffling words in the positive captions. As new research has begun to propose methods that claim to improve compositionality on these benchmarks [Yuksekgonul et al., 2023; Ray et al., 2023], we find it critical to highlight our findings and propose a solution.

We propose a solution to existing hackable benchmarks by introducing SUGARCREPE, a new benchmark to faithfully evaluate compositionality. In curating SUGARCREPE, we identify two main *biases* [1] that result in the distributional gap between positive and hard negatives; and employ mechanisms to fix the shifts. In particular, we find the current procedure in generating hard negatives introduces descriptions that are (1) not plausible and (2) non-fluent. For example, while the caption "olives and grapes on a plate" is a sensical fluent caption, benchmarks often have non-plausible hard negatives like "olives and grapes inside a plate" or simply incomprehensible ones like "right has word another word. There is a words" (see Table 4.1 for more examples). We mitigate such biases by first leveraging a modern large language model, ChatGPT [OpenAI, 2022], to generate plausible and natural hard negative texts instead of relying on simple rule-based templates employed by existing benchmarks [Ma et al., 2022; Yuksekgonul et al., 2023]. Then, we subsample the dataset through an adversarial refinement process to ensure the identified biases are maximally removed by drawing on recent dataset de-biasing work [Zellers et al., 2018; Sakaguchi et al., 2021; Le Bras et al., 2020]. Taken together, this workflow is where SUGARCREPE derived its name: **S**ynthetic yet **U**nbiased **G**eneration with **A**dversarially **R**efined **C**ompositional **REP**resentation **E**valuation. We qualitatively and quantitatively

[1] We use biases and artifacts interchangeably in the paper.

verify through both human and automatic evaluations that SUGARCREPE effectively fixes these biases.

With SUGARCREPE, we *re*-evaluate recent methods proposed to improve compositionality. Specifically, we focus on one prominent approach that aims to improve compositionality through data augmentation. This method trains models by generating compositional hard negatives and injecting them within a training batch [Doveh et al., 2023; Yuksekgonul et al., 2023]. Unfortunately, we observe that the effectiveness of this simple data augmentation approach is hugely *overestimated* when evaluated on existing benchmarks, leading to limited improvements on SUGARCREPE. Finally, we evaluate a wide variety of 17 pretrained CLIP models [Radford et al., 2021a; Ilharco et al., 2021; Gadre et al., 2023], and find that current models still lack compositionality. Our results suggest that to improve compositionality, future work may need more innovative techniques.

4.2 Related Work

We situate our paper amongst existing work on vision-language compositionality, and debiasing datasets for model evaluation.

Evaluating vision-language compositionality. Recent works have introduced benchmarks to evaluate the compositionality of vision-language models [Radford et al., 2021a]; they find that current models exhibit little compositional understanding [Yuksekgonul et al., 2023; Thrush et al., 2022; Zhao et al., 2022; Ma et al., 2022; Ray et al., 2023] despite their remarkable performance on downstream tasks [Radford et al., 2021a; Li et al., 2022b; Singh et al., 2022; Alayrac et al., 2022; Wang et al., 2022a,c; Zhai et al., 2022]. Models have a hard time discerning between text containing the same words ordered differently [Thrush et al., 2022]. Models also fail to link objects to their attributes, or understand the relationship between objects [Zhao et al., 2022; Yuksekgonul et al., 2023; Ray et al., 2023]. Our work finds that many of the benchmarks used to evaluate compositionality have hackable biases; blind models that do not even look at the image outperform state-of-the-art vision-language models.

Improving vision-language compositionality. To enhance vision-language models' compositionality, new proposals suggest training strategies that utilize additional data, models, and/or losses [Yuksekgonul et al., 2023; Cascante-Bonilla et al., 2023; Ray et al., 2023; Doveh et al., 2023; Singh et al., 2023]. Amongst them, one prominent approach is to explicitly train the models to distinguish hard negatives from the

correct captions [Yuksekgonul et al., 2023; Doveh et al., 2023]. While these approaches appear to improve compositionality on benchmarks, it is unclear if these models achieve such improvements by actually acquiring compositional understanding or by exploiting biases in these datasets. We answer this question in our evaluation.

Debiasing dataset for faithful model evaluation. Several prior manuscripts have pointed out that biased datasets could lead to an overestimation of models' true capabilities [Gururangan et al., 2018]. They have proposed dataset de-biasing methods to enable more faithful model evaluations [Reif and Schwartz, 2023; Zellers et al., 2018; Sakaguchi et al., 2021; Le Bras et al., 2020; Park et al., 2021]. For instance, adversarial filtering [Zellers et al., 2018] iteratively trains an ensemble of classifiers on different training splits and uses them to filter out "easy" negatives for each instance. Building upon adversarial filtering, AFLite [Sakaguchi et al., 2021; Le Bras et al., 2020] filters data instances in a more light-weight manner without retraining a model at each iteration and leads to benchmarks that more accurately represent the underlying tasks. We use adversarial refinement to remove biases that creep into the generation of compositionality benchmarks.

4.3 Limit and biases of current compositionality benchmarks

A majority of existing compositionality benchmarks for vision-language models formulate the evaluation task as image-to-text retrieval [Zhao et al., 2022; Yuksekgonul et al., 2023; Ma et al., 2022]. We focus on these benchmarks and discuss others [Thrush et al., 2022; Ray et al., 2023] in Appendix C.2. Given an image, the model is probed to select text that correctly describes the image from a pool of candidates. Unlike standard retrieval tasks where the negative (incorrect) candidates differ a lot from the *positive* (correct) text, compositionality benchmarks intentionally design *hard negative* texts that differ minimally from the positive text, in order to test whether the model understands the fine-grained atomic concepts that compose the scene.

Existing hard negative generation process introduces undesirable biases. Existing benchmarks generate hard negative texts through rule-based programmatic procedures [Zhao et al., 2022; Yuksekgonul et al., 2023; Ma et al., 2022], which produce hard negatives by replacing a word of specific type (an object, attribute, or relation) in the original text, by swapping two words, or by shuffling the word order. We find that such procedures introduce unintentional biases in the generated hard negatives (see Table 4.1); specifically, we observe two major types of undesirable artifacts: (1) *nonsensical* artifacts, and (2) *non-fluent* artifacts.

Table 4.1: Existing compositionality benchmarks rely on procedurally-generated hard negatives which often do not make logical sense or are not fluent due to grammatical errors.

Dataset	Nonsensical Hard Negatives	Non-fluent Hard Negatives
CREPE [Ma et al., 2022]	Olives and grape inside a plate. Ground in a basket on the flowers. A hair wearing a necklace, with her lady on a table.	A door with panes not in a room; the door has windows. Right has word another word. There is a words. A shelf with books in something. There is no background.
ARO [Yuksekgonul et al., 2023]	The grass is eating the horse. A gray bathtub is looking at a white cat. Green ball with a remarkable chair behind a blue scene.	At brown cat a in looking a gray dog sitting is and white bathtub. Scene with remarkable a ball blue a green behind chair. Books the looking at people are.
VL-CheckList [Zhao et al., 2022]	Sheep is hardwood. Empty zebras. The bush speaking in the garden.	An man fishing a food from a wrapper using a paw at a open. It heaving at a city. An grouping subduing at a room access.

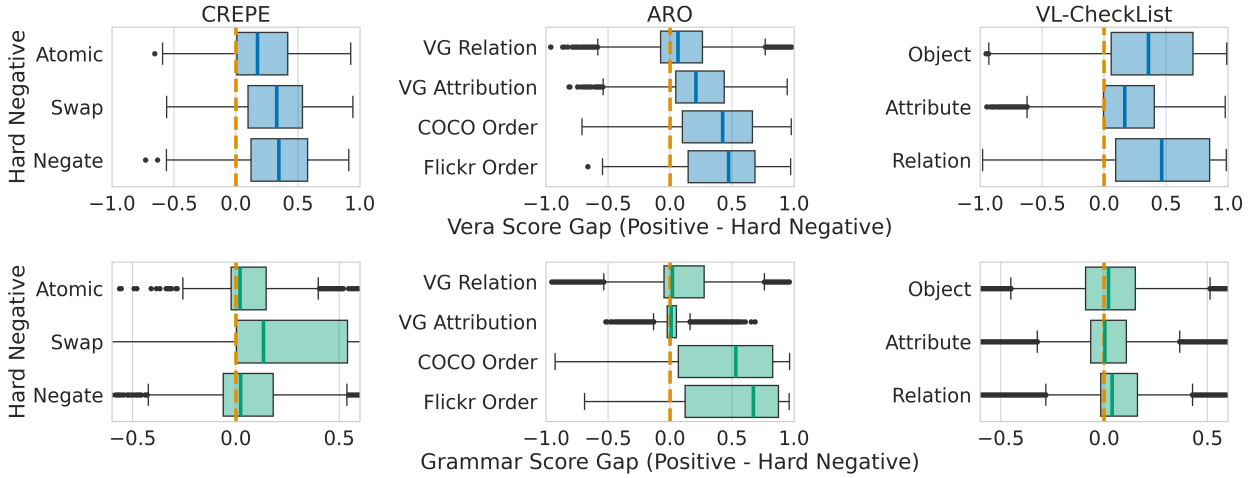


Figure 4.1: Top row: We define *Vera score gap* as the score difference between the positive and hard negative texts: $\mathrm{Vera}(T^{\mathrm{p}}) - \mathrm{Vera}(T^{\mathrm{n}})$. The entire Vera score gap distribution lies on the positive spectrum, indicating that the template-generated hard negative texts usually have low plausibility. Bottom row: Similarly, *Grammar score gap* is defined by: $\mathrm{Grammar}(T^{\mathrm{p}}) - \mathrm{Grammar}(T^{\mathrm{n}})$. On grammar score, we also find that the distribution largely rests on the positive side, suggesting that most hard negative texts in existing benchmarks exhibit grammatical errors.

In order to quantitatively measure these biases, we utilize Vera [Liu et al., 2023a], a plausibility estimation model, to characterize the nonsensical bias. Specifically, we define $\mathrm{Vera}(T)$ to be the plausibility score of a caption T, where a higher score suggests more sensical the caption is. Similarly, to capture the non-fluent bias, we leverage a grammar-check model [Morris et al., 2020] that assigns high scores, $\mathrm{Grammar}(T)$, to more grammatically correct texts. In Figure 4.1, we find that Vera and the grammar model assign higher scores to positive texts, suggesting that many hard negatives are nonsensical and not fluent.

Dataset biases render current compositionality benchmarks ineffective. Given the heavily-skewed score gaps, we show that blind models (i.e. , Vera and the grammar model) that simply select the higher-scoring texts as positives and admittedly do not possess any vision-language compositionality, can achieve

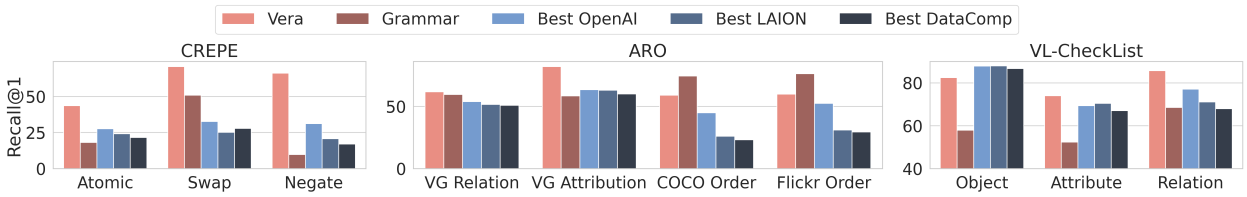


Figure 4.2: Blind commonsense Vera model and Grammar model outperform state-of-the-art CLIP models on nearly *all* existing benchmarks by exploiting the nonsensical and non-fluent artifacts. This suggests that existing benchmarks are hackable and ineffective in measuring compositionality.

state-of-the-art performances on existing benchmarks. We compare the the blind models against 17 pre-trained CLIP models from three sources: OpenAI's in-house WebImageText dataset [Radford et al., 2021a], LAION [Schuhmann et al., 2022b], and Datacomp [Gadre et al., 2023]. We plot the performances of the blind models and the best-performing CLIP models from each category (Figure 4.2). Blind models achieves state-of-the-art performances on 9 out of 10 existing benchmark tasks. We provide full evaluation results in Appendix C.4.1.

4.4 SUGARCREPE

We introduce SUGARCREPE, a new benchmark for faithful evaluation of vision-language models' compositionality based on the image-text pairs of COCO [Lin et al., 2014]. SUGARCREPE presents two key contributions over existing benchmarks: (1) it drastically reduces the two identified dataset biases (Sec. 4.4.1), and (2) it covers a broad range of fine-grained types of hard negatives (Sec. 4.4.2). We present a summary comparison on compositionality benchmarks in Appendix C.2.

4.4.1 SUGARCREPE generation workflow alleviates dataset biases

The generation procedure of SUGARCREPE consists of three main stages, centered around creating sensical and fluent hard negatives that close the distributional gaps to the positive texts, and ensuring a balanced distribution on the score gaps to make the final dataset robust to the identified biases.

Stage 1: Generate sensical and fluent hard negatives with a large language model. Observing the capability of modern large language models in generating fluent and plausible texts, we leverage ChatGPT [OpenAI, 2022] to generate hard negative texts where we explicitly instruct it to avoid commonsense

```
Given an input sentence describing a scene, your task
is to:
1. Locate the noun words in the sentence.
2. Randomly pick one noun word.
3. Replace the selected noun word with a new noun word
to make a new sentence.

The new sentence must meet the following three
requirements:
1. The new sentence must be describing a scene that is
as different as possible from the original scene.
2. The new sentence must be fluent and grammatically
correct.
3. The new sentence must make logical sense.

Here are some examples:

Original sentence: A man is in a kitchen making pizzas.
Nouns: ["man", "kitchen", "pizzas"]
Selected noun: man
New noun: woman
New sentence: A woman is in a kitchen making pizzas.

Original sentence: a woman seated on wall and birds
besides her
Nouns: ['woman', 'wall', 'birds']
Selected noun: wall
New noun: bench
New sentence: A woman seated on a bench and birds
besides her.
```

Figure 4.3: Example prompt (black) and actual hard negative (green) generated from ChatGPT.

Algorithm 1 Adversarial Refinement

Require: Text-only model M_1 and M_2; Number of grids K; A set of candidates $\mathcal{D} = \{I_i, T_i^{\text{p}}, T_i^{\text{n}}\}_{i\in[N]}$, where I_i, T_i^{p}, and T_i^{n} are i-th image, positive caption, and negative caption.

Ensure: A subset $\bar{\mathcal{D}} \subset \mathcal{D}$

Calculate the model score gap for each candidate $g_i^{(1)} = M_1(T_i^{\text{p}}) - M_1(T_i^{\text{n}})$ and $g_i^{(2)} = M_2(T_i^{\text{p}}) - M_2(T_i^{\text{n}})$
Split the 2D space $[-1, 1] \times [-1, 1]$ to $K \times K$ equal-size grids.
Place each candidate to a grid based on the score gaps $g_i^{(1)}$ and $g_i^{(2)}$.
Initialize $\bar{\mathcal{D}} = \{\}$
for each pair of grid (G_j, G_j^*) symmetric about the original point $(0, 0)$ **do**
    if $|G_j| > |G_j^*|$ **then**
        Sample $|G_j^*|$ candidates from G_j and put them to $\bar{\mathcal{D}}$.
        Put candidates in G_j^* to $\bar{\mathcal{D}}$.
    else
        Sample $|G_j|$ candidates from G_j^* and put them to $\bar{\mathcal{D}}$.
        Put candidates in G_j to $\bar{\mathcal{D}}$.

(logical) and fluency (grammatical) errors. To guide ChatGPT in re-writing a given positive text into its hard negative counterparts, we provide few-shot demonstrations written by the authors and leverage its in-context learning ability [Brown et al., 2020] to generalize to unseen texts. Figure 4.3 shows an example demonstration used and an actual hard negative generated. We detail all the prompt templates in Appendix C.3.2. Table 4.3 shows the comparisons between hard negatives generated from ChatGPT in SUGARCREPE and that from existing benchmarks.

Stage 2: Filter false negatives with human validation. A generated text is considered a valid hard negative only if it incorrectly describes the corresponding image. For example, given an image with a positive caption "a man and a child sitting on a sofa", a compositional change that replaces "child" with "girl" may still result in a correct caption. To ensure the validity of the hard negatives in SUGARCREPE, we filter out *false* negatives by manually examining the generated hard negatives and their corresponding images.

Stage 3: De-bias dataset with adversarial refinement. While ChatGPT yields more sensical and fluent text, there is no guarantee that the bias between positive and negative texts is negligible. Following dataset

de-biasing work [Zellers et al., 2018; Sakaguchi et al., 2021; Le Bras et al., 2020], we develop an adversarial refinement mechanism that maximally reduces the undesirably exploitable artifacts in SUGARCREPE. Specifically, our goal is to ensure that performance improvements on SUGARCREPE cannot be achieved by exploiting the identified nonsensical and non-fluent biases. To accomplish this, we characterize the biases again with the commonsense and grammar models [Liu et al., 2023a; Morris et al., 2020], and subsample the dataset to ensure symmetric score gap distributions on both the positive and negative sides, as shown in Figure 4.4. We note the symmetry around zero implies that the commonsense and grammar scores can no longer be used to infer the ground truth positive texts. We provide the adversarial refinement algorithm in Algorithm 1.

4.4.2 SUGARCREPE covers a broad range of hard negative types

To test different aspects of vision-language models' compositional understanding, we follow CREPE [Ma et al., 2022] to consider various *forms* of hard negatives, and follow VL-CheckList [Zhao et al., 2022] and ARO [Yuksekgonul et al., 2023] to consider different fine-grained *categories* of the atomic concepts. In total, SUGARCREPE covers 7 fine-grained types of hard negatives, as shown in Table 4.2. We introduce the dataset taxonomy below, starting from the *form* of the hard negatives to its different *finer-grained* variants.

The REPLACE form. Given a positive text describing a scene, we generate a REPLACE hard negative by replacing an atomic concept in the original text with a new concept that makes the text mismatch with the original scene. Based on the type of the atomic concept—object, attribute, or relation—we further categorize REPLACE hard negatives into REPLACE-OBJ, REPLACE-ATT, and REPLACE-REL.

The SWAP form. Different from REPLACE, SWAP does not introduce new concepts in the hard negatives, but a SWAP hard negative is generated by swapping two atomic concepts of the same category in the positive text. We further categorize SWAP into SWAP-OBJ and SWAP-ATT, and omit swapping two relationships since it generally results in nonsensical texts.

The ADD form. Similar to the REPLACE form, but instead of replacing an atomic concept with a new one, we generate an ADD hard negative by adding a new atomic concept to the positive text that makes it mismatch with the original scene. We only further categorize ADD into ADD-OBJ (adding object concept) and ADD-ATT (adding attribute concept), as adding new relationship concepts to the positive texts often

Table 4.2: We report the number of hard negative captions of all types in SUGARCREPE.

	REPLACE			SWAP		ADD	
	Object	**Attribute**	**Relation**	**Object**	**Attribute**	**Object**	**Attribute**
# negative captions	1,652	788	1,406	246	666	2,062	692

make them highly implausible.

Dataset overview. The final evaluation set of SUGARCREPE consists of $7,512$ examples, where the numbers for each fine-grained type are listed in Table 4.2. Each example is an image-to-text retrieval task composed of an image, a positive text, and a hard negative. On SUGARCREPE, random chance performance has an average accuracy of 50%. We note that ARO and CREPE additionally consider SHUFFLE (randomly shuffling words in a sentence) and NEGATE (adding negation keywords "no/not" to a sentence) hard negatives. We however omit them in SUGARCREPE as SHUFFLE is very unlikely to be plausible and fluent, and NEGATE introduces irreducible keyword artifacts [Ma et al., 2022]. [2]

4.5 Evaluations

In this section, we qualitatively and quantitatively compare SUGARCREPE to existing benchmarks (Sec. 4.5.1), re-evaluate recent methods proposed to improve compositionality of vision-language models (Sec. 4.5.2), and comprehensively evaluate a wide array of pretrained CLIP models (Sec. 4.5.3).

To systematically and fairly compare SUGARCREPE with existing benchmarks, we normalize the benchmarks by reproducing their data generation workflow using COCO [Lin et al., 2014] as in SUGARCREPE. We utilize source code from CREPE [Ma et al., 2022] to generate REPLACE, SWAP, NEGATE hard negatives and take SHUFFLE hard negatives released in ARO [Yuksekgonul et al., 2023]. We refer to this reproduced dataset as ARO+CREPE. In addition, we standardize the evaluation task as retrieving the correct caption from *two* possible choices, i.e. , a positive text and a hard negative. This normalization sets the positive texts fixed for all benchmarks, including SUGARCREPE.

[2] One can easily infer hard negatives from whether the text contains negation keywords "no/not".

Table 4.3: We present example positive texts and their hard negatives in ARO+CREPE (generated using existing procedures) and SUGARCREPE (generated with ChatGPT). SUGARCREPE brings significant improvements in commonsense and fluency.

Hard-Negative Type	Text Type	Commonsense	Fluency
REPLACE	Original	Two adult bears play fight in the water.	A man sitting in front of a laptop computer.
	ARO+CREPE	Two adult bears play fight in the soda.	A man sitting around front of a laptop computer.
	SUGARCREPE	A flock of ducks play fight in the water.	A man standing in front of a laptop computer.
SWAP	Original	A woman standing behind a fence looking at an elephant.	Man swinging tennis racket while group of people watches.
	ARO+CREPE	A fence standing behind a woman looking at an elephant.	Group swinging tennis racket while man of people watches.
	SUGARCREPE	An elephant standing behind a fence looking at a woman.	Group of people swinging tennis racket while man watches.
NEGATE / ADD	Original	A teddy bear next to a stuffed fish.	A red fire hydrant on a city sidewalk.
	ARO+CREPE	A teddy bear next to a stuffed fish. There is no teddy bear.	A red fire not hydrant on a city sidewalk.
	SUGARCREPE	A teddy bear and a stuffed fish and a robot toy.	A red fire hydrant and a trash can on a city sidewalk.

Table 4.4: We compare the commonsense and grammar scores on hard negatives in ARO+CREPE and SUGARCREPE. We report both their respective average scores and the ratio where SUGARCREPE has higher score than ARO+CREPE in pairwise comparison. Overall, SUGARCREPE has hard negatives with better commonsense and grammar.

		Average Score		
Hard-negative Type	Metric	ARO+CREPE	SUGARCREPE	Pairwise Better Ratio
REPLACE	Commonsense	37.46	50.21	77.71
	Grammar	76.79	88.96	86.85
SWAP	Commonsense	23.09	41.57	78.76
	Grammar	45.67	80.46	87.02
NEGATE / ADD	Commonsense	25.24	50.20	87.24
	Grammar	65.09	90.07	95.03

4.5.1 SUGARCREPE significantly reduces dataset biases

SUGARCREPE generates more sensical and fluent hard negatives. We validate that SUGARCREPE generates higher quality hard negative texts by leveraging ChatGPT than previous rule-based approaches. Qualitatively, in Table 4.3, we observe that the hard negatives in SUGARCREPE are more sensical and fluent compared to hard negatives in ARO+CREPE. We report human evaluation results in Appendix C.4.2 that show on an average of 35% of examples, hard negatives in SUGARCREPE have *strictly* higher quality than ARO+CREPE in terms of commonsense and fluency. For instance, on SWAP, humans judge that SUGARCREPE wins 68% over ARO+CREPE and ties on 28% of examples in terms of commonsense. Quantitatively, in Table 4.4, we compare the commonsense and grammar scores averaged over the hard negative texts in both ARO+CREPE and SUGARCREPE. We see SUGARCREPE has much higher average scores than ARO+CREPE. Additionally, pairwise comparisons show that SUGARCREPE has higher commonsense and grammar scores than ARO+CREPE on 86% of examples on average.

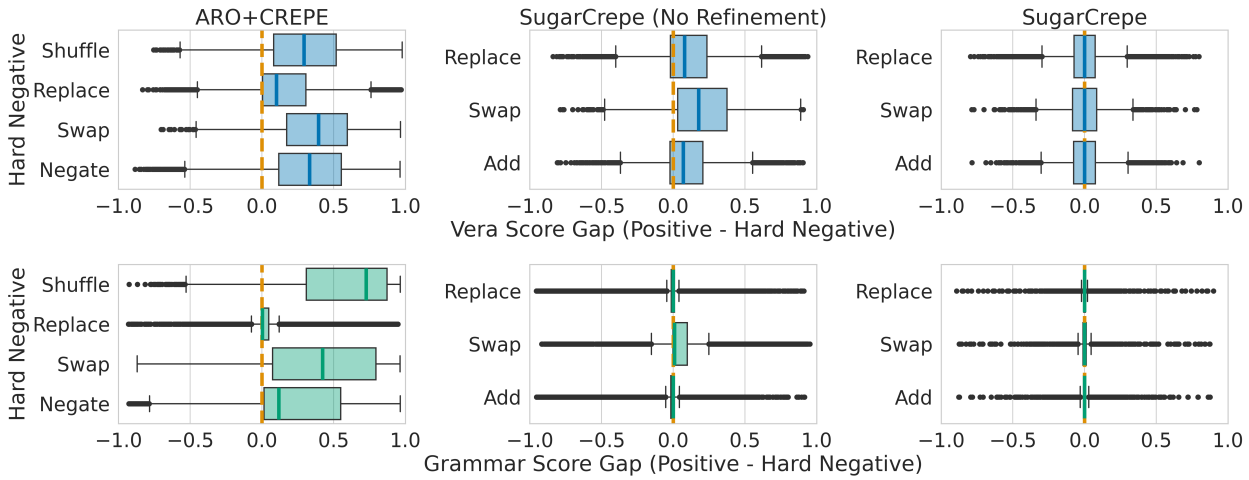


Figure 4.4: We compare the Vera (top row) and Grammar (bottom row) score gap distributions between ARO+CREPE (leftmost column), SUGARCREPE without adversarial refinement (middle), and SUGARCREPE (rightmost). Top row: We see that Vera score gap distribution shifts from the positive spectrum to more centered around zero from ARO+CREPE to SUGARCREPE without refinement. After adversarial refinement, we ensure the score gap distribution is centered around zero on SUGARCREPE. Bottom row: Similarly, from ARO+CREPE to SUGARCREPE, we see the Grammar score gap distribution shifts from the positive spectrum to centered around zero.

SUGARCREPE disentangles the identified exploitable biases. We show that the final SUGARCREPE evaluation set maximally reduces the identified biases that could be exploited undesirably to achieve improvements on a benchmark. Figure 4.4 visualizes the Vera/Grammar score gap distributions. We compare the distributions between ARO+CREPE and SUGARCREPE (before and after adversarial refinement). First, We see that by leveraging ChatGPT, the hard negative texts in SUGARCREPE already have lower biases than ARO+CREPE before adversarial refinement, i.e. , the score gap distribution is more centered around zero. Furthermore, we see that after adversarial refinement, the score gap distributions on the final SUGARCREPE evaluation set are symmetric around zero. This implies that the previously identified artifacts can no longer be exploited to infer the positive texts. As a result, we show that the previous commonsense and grammar attacks that are extremely successful on existing benchmarks do not work on SUGARCREPE. As shown in Table 4.6, these blind models now consistently rank the *last* on SUGARCREPE as compared to other pretrained CLIP models.

4.5.2 Re-evaluating recent methods for improving compositionality

Given the vulnerability of existing compositionality benchmarks, it is unclear whether recently proposed methods that show state-of-the-art performances on these benchmarks are indeed effective. Thus, we re-evaluate these methods with SUGARCREPE.

Hard negative augmented training. Specifically, we focus on evaluating one common *data-augmentation* approach considered in [Yuksekgonul et al., 2023; Doveh et al., 2023], where the core idea is to explicitly create hard negatives and train the model to distinguish them. We broadly refer to this training scheme as NEGCLIP following [Yuksekgonul et al., 2023]. We evaluate two NEGCLIP training schemes: finetuning and training from scratch. For finetuning, in addition to taking the model released in [Yuksekgonul et al., 2023], we finetune another three NEGCLIP models (using ViT-B/32 following [Yuksekgonul et al., 2023]) with three respective types of hard negatives (i.e. , REPLACE, SWAP, NEGATE) generated using CREPE's [Ma et al., 2022] source code. For training from scratch, we use RN50 as the base model and train variants of NEGCLIP by augmenting the training examples with different types of hard negatives. We perform both training and finetuning on COCO [Lin et al., 2014].

Improvements are overestimated due to unintentionally overfitting. In Table 4.5, we first see that NEGCLIP finetuned models show significant improvements on ARO+CREPE, boosting the performance more than 10% compared to standard CLIP finetuning on 11 out of 16 cases (highlighted in green). The lifts are especially large when the hard negative type used in finetuning matches that used in evaluation, where NEGCLIP finetuned models can achieve near human-level performances. For instance, by finetuning with REPLACE hard negatives, NEGCLIP reaches 94% on ARO+CREPE evaluated with REPLACE hard negatives (human performance is 95%). While the results on ARO+CREPE suggest that NEGCLIP is seemingly sufficient in equipping models with strong compositionality, we however see that the improvements brought by NEGCLIP are much smaller on SUGARCREPE. In fact, none of the improvements on SUGARCREPE is larger than 10%, and the best performing NEGCLIP finetuned models still have large gaps to human-level performances, e.g. , best NEGCLIP model lags behind human by 23% on SUGARCREPE's SWAP hard negatives. Similarly, when trained from scratch, we observe the same trend that NEGCLIP's improvements are much larger on ARO+CREPE than on SUGARCREPE. The improvements on ARO+CREPE are again most pronounced when the training and testing hard negative type matches.

We attribute the stark contrast in NEGCLIP's effectiveness on ARO+CREPE and SUGARCREPE to model's unintentional overfitting: The NEGCLIP models learned to exploit artifacts that can be used to easily distinguish hard negatives from positives on ARO+CREPE, instead of actually improving compositionality. Thus, when evaluated on SUGARCREPE where the artifacts are removed, the improvement from NEGCLIP drastically reduces. These results imply that NEGCLIP's effectiveness is overestimated on existing benchmarks, and we may still need further innovations to fundamentally improve a model's compositionality. [3]

Table 4.5: Re-evaluating hard negative augmented training shows that the method's improvements on existing benchmarks (ARO+CREPE) are hugely overestimated, particularly when the test hard negative type matches the one used in training, which can be attributed to overfitting the artifacts.
Color notations: Gains compared to standard CLIP (finetuned / from scratch) $> 10\%$.

			ARO+CREPE				SUGARCREPE		
Model	**Training**	**Hard Negative Used**	**REPLACE**	**SWAP**	**NEGATE**	**SHUFFLE**	**REPLACE**	**SWAP**	**ADD**
Human			95.33	100	99.33	96.00	98.67	99.50	99.00
ViT-B/32	Pretrained	N/A	75.71	71.58	76.89	72.06	80.76	63.27	75.09
	CLIP finetuned	N/A	77.06	68.81	61.19	63.04	84.76	70.83	85.58
	NEGCLIP finetuned	REPLACE	94.51	90.04	85.06	88.15	88.27	74.89	90.16
		SWAP	82.88	94.48	77.57	87.00	85.54	76.21	86.56
		NEGATE	77.24	68.91	99.54	64.28	84.97	70.29	85.84
		Released in [Yuksekgonul et al., 2023]	85.72	94.35	83.51	90.45	85.36	75.33	87.29
RN50	CLIP from scratch	N/A	69.93	59.96	55.36	68.78	69.54	60.33	67.63
	NEGCLIP from scratch	REPLACE	89.04	66.51	60.90	75.23	74.32	62.65	72.92
		SWAP	72.33	92.29	64.51	84.84	73.31	68.35	71.93
		NEGATE	70.09	60.29	99.45	69.03	72.74	60.89	70.47
		REP + SW + NEG	86.30	88.60	99.34	82.93	75.26	67.69	73.08

4.5.3 Comprehensive evaluations on existing pretrained vision-language models

We present four key findings in our evaluation over 17 pretrained CLIP models on SUGARCREPE, with results reported in Table 4.6 and visualized in Figure 4.5.

The best pretrained CLIP models demonstrate some compositional understanding but still have overall large rooms for improvements. Table 4.6 shows that the largest pretrained CLIP models, e.g. , OpenAI's RN50x64, LAION's xlm-roberta-large-ViT-H-14, and DataComp's ViT-L-14, achieve near-human performance on REPLACE-OBJ. However, on REPLACE-OBJ, smaller models pretrained on small datasets still suffer from big drops in performance — 23% and 43% respectively for DataComp's small and medium

[3] In Appendix C.4.3, we provide further results on training NEGCLIP with hard negatives filtered with our adversarial refinement mechanism.

Table 4.6: Our evaluation of pretrained CLIP models on SUGARCREPE shows that they demonstrate compositionality on some hard negatives but are far from human performance on others, especially on SWAP hard negatives or ones perturbing attributes and relations (also illustrated in Figure 4.5: lower overall performance on SWAP, and lower performances on attributes/relations compared to objects). We additionally evaluate recently introduced GPT-4V [OpenAI, 2023]. While it demonstrates strong results, there is still gap to human-level performance.

				REPLACE			SWAP		ADD		
Source	**Model**	**Data Size**	**Model Size (M)**	**Object**	**Attribute**	**Relation**	**Object**	**Attribute**	**Object**	**Attribute**	**Average**
Human				100	99	97	99	100	99	99	99
Text-only model	Vera [Liu et al., 2023a]			49.39	49.62	49.36	49.19	49.40	49.42	49.57	49.42
	Grammar [Morris et al., 2020]			50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00
OpenAI [Radford et al., 2021a]	RN50	400M	102	91.77	80.58	69.99	61.79	68.47	74.54	69.65	73.83
	RN101		120	92.49	83.88	67.07	56.50	65.92	75.46	70.09	73.06
	ViT-B-32		151	90.92	80.08	69.20	61.38	63.96	77.21	68.79	73.08
	ViT-B-32-negclip		151	92.68	85.91	76.46	75.20	75.38	88.80	82.80	82.46
	RN50x4		178	92.68	82.99	67.57	65.04	63.36	79.34	70.09	74.44
	RN50x16		291	93.46	82.11	69.20	63.01	65.77	80.70	75.87	75.73
	ViT-L-14		428	94.07	79.19	65.15	60.16	62.31	78.32	71.53	72.96
	RN50x64		623	94.49	83.50	70.63	61.79	66.67	83.27	73.99	76.33
LAION [Schuhmann et al., 2022b]	roberta-ViT-B-32	2B	212	92.86	84.90	72.40	63.01	71.02	87.34	79.91	78.78
	ViT-H-14		986	96.49	84.77	71.76	67.48	73.12	92.05	85.84	81.64
	ViT-g-14		1367	95.76	85.03	72.40	63.01	71.17	91.51	82.08	80.14
	ViT-bigG-14		2540	96.67	88.07	74.75	62.20	74.92	92.19	84.54	81.91
	xlm-roberta-base-ViT-B-32	5B	366	93.16	84.01	69.20	63.41	67.57	87.78	81.07	78.03
	xlm-roberta-large-ViT-H-14		1193	96.85	86.04	72.05	63.82	72.07	93.11	86.13	81.44
DataComp [Gadre et al., 2023]	`small`: ViT-B-32	13M	151	56.90	56.85	51.99	50.81	50.00	53.93	60.55	54.43
	`medium`: ViT-B-32	128M	151	77.00	69.54	57.68	57.72	57.06	66.73	64.88	64.37
	`large`: ViT-B-16	1B	150	92.68	79.82	63.94	56.10	57.66	84.34	78.61	73.31
	`xlarge`: ViT-L-14	13B	428	95.52	84.52	69.99	65.04	66.82	91.03	84.97	79.70
OpenAI	GPT-4V [OpenAI, 2023]			96.31	93.53	90.26	83.13	90.09	91.59	91.76	90.95

models — compared to humans. Additionally, on nearly all other hard negative types, there are clear gaps (larger than 10%) between the best model performances and human performances, showing an overall large room for improvements in current models' compositionality.

All models struggle at identifying SWAP hard negatives, regardless of their pertaining dataset and model size. Among the three types of hard negatives, SWAP hard negatives present the biggest challenge to the pretrained CLIP models, even though humans can easily tell them apart from the positive captions. We observe in Table 4.6 that all models demonstrate low performance on both SWAP-OBJ and SWAP-ATT hard negatives regardless of their pretraining dataset and model sizes, with the difference from human performance reaching from 27% to 50%.

Existing models are object-centric, struggling to compose attributes and relations. We find that existing pretrained models are a lot better at composing objects than attributes or relations (Table 4.6). This finding holds for both REPLACE and ADD hard negatives but not the most difficult SWAP negatives, where models perform equally poorly on both SWAP-OBJ and SWAP-ATT. On REPLACE hard negatives, even though most models achieve human-level performance on REPLACE-OBJ, they all suffer from a drop in

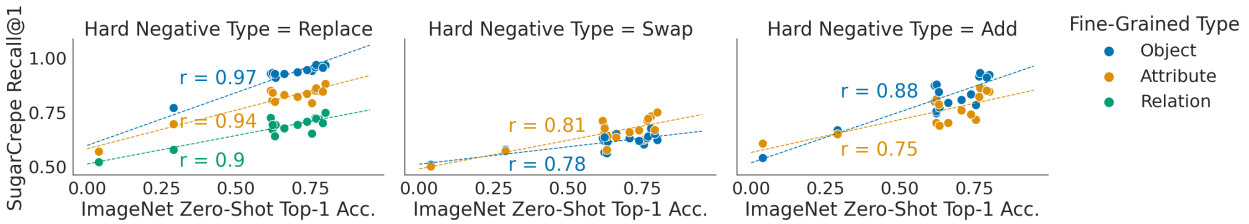


Figure 4.5: We plot pretrained vision-language models' zero-shot top-1 accuracy on ImageNet versus their retrieval recall@1 on SUGARCREPE, where r is the Pearson correlation coefficient. This plot suggests that models' ImageNet zero-shot accuracy positively correlates with their compositionality.

performance on REPLACE-ATT and REPLACE-REL, where the drop is as large as 15% and 29% respectively. Similarly, on ADD hard negatives, all models except for DataComp's small:ViT-B-32 experience a decrease in performance from ADD-OBJ to ADD-ATT, with the largest difference reaching 10%.

Models' performance on SUGARCREPE correlates with their ImageNet zero-shot accuracy. We show in Figure 4.5 that there is a positive correlation between models' performance on SUGARCREPE and their zero-shot accuracy on ImageNet. This correlation is moderate on SWAP-OBJ and ADD-ATT (Pearson correlation coefficient $r = 0.78$ and $r = 0.75$ respectively) and strong on all other hard negatives ($r > 0.8$).

4.6 Discussions

Our investigation reveals significant biases present in existing benchmarks for the compositional comprehension capability of vision-language models. The severity of this vulnerability is exemplified by text-only models without access to the image outperforming vision-language models. To address this, we introduce SUGARCREPE, a novel benchmark for evaluating the compositionality of vision-language understanding. Unlike previous benchmarks that relied on rule-based templates, we leverage large language models to generate less biased negatives and employ adversarial filtering mechanisms to minimize biases. Through reassessment of state-of-the-art models and recently proposed compositionality inducing mechanisms, we uncover a significant overestimation of their advancements, underscoring the need for further innovation.

4.6.1 Limitation and future work

Scope of the compositionality benchmarks and vision-language models. We focus our scope in this chapter on compositionality benchmarks formulated as image-to-text retrieval task. While this is currently

the most prevailing evaluation framework, future research can characterize compositionality evaluation as text-to-image retrieval problem, as in the initial efforts considered by [Ray et al., 2023; Thrush et al., 2022]. More importantly, we hope our work can guide future efforts in creating and ensuring faithful compositionality benchmarks in text-to-image form. In addition, we focus our evaluations on contrastively learned vision-language models [Radford et al., 2021a]. Future work should include and characterize the compositionality of modern generative vision-language models [Alayrac et al., 2022; Chen et al., 2022; Li et al., 2023c; Tschannen et al., 2023].

Potential biases imposed by language models. In this chapter, we identify *two* human interpretable dataset biases, the nonsensical and non-fluent biases, which may not cover all dataset artifacts that could possibly be exploited by a model. By leveraging ChatGPT in generating hard negatives, the generated captions may also exhibit hard to detect biases imposed by the language model, e.g., watermarks [Kirchenbauer et al., 2023]. Future work may utilize more sophisticated adversarial filtering techniques that train models to detect and remove spurious dataset artifacts beyond human comprehension [Zellers et al., 2019; Le Bras et al., 2020].

Shifts in language model behavior. Our work leverages ChatGPT to generate hard negatives. However, recent work has pointed out that the underlying model behind these APIs may change, resulting in model behavior shifts [Chen et al., 2023b; Liu et al., 2023c]. We discuss how this potential model behavior shift may affect our proposed dataset construction pipeline. Specifically, while there may be variances on the quality of the generated texts, we note that our employed adversarial refinement mechanism can ensure that the final evaluation set is free of the identified artifacts. In the case when ChatGPT improves and generates higher-quality captions, the refinement mechanism will filter out less examples and we can more efficiently create the final evaluation set. On the other hand, if ChatGPT degrades and shifts towards generating less fluent and plausible captions, the refinement mechanism will filter out more generated examples and we would need to generate more candidates in order to create an evaluation set of the same desired size. As a result, while the efficiency of the proposed dataset construction pipeline depends on quality of the language model used, our pipeline ensures the generated set does not contain the identified biases. In the large language model era, we see these capable models as productive tools one can leverage to efficiently process and create data. We do however deem careful validation mechanisms, such as our manual and automatic filtering

technique, necessary to ensure that the ultimate goal is properly achieved.

4.6.2 Societal impact

As vision-language models such as CLIP [Radford et al., 2021a] are becoming the foundation models for many downstream applications [Rombach et al., 2022; Ramesh et al., 2022], it is imperative to understand the limitations of these models to avoid misuses and undesirable outcomes [Cho et al., 2022; Bianchi et al., 2022]. Compositionality benchmarks probe a model's understanding of finer-grained concepts, and hence allow us to identify blind spots [Yuksekgonul et al., 2023; Zhao et al., 2022; Ma et al., 2022] of seemingly powerful models deemed by standard classification and retrieval benchmarks [Deng et al., 2009; Lin et al., 2014]. Our work further alleviates common artifacts in existing compositionality benchmarks that result in overestimation of a model's capability. We hope our proposed benchmark SUGARCREPE leads to more faithful assessment of a vision-language model's compositionality, and can hence guide more accurate usages of the models. Nevertheless, we note that strong performances on SUGARCREPE do not imply perfect models. We envision SUGARCREPE being one of the many benchmarks used to comprehensively understand the abilities of vision-language models from various aspects.

Chapter 5

Data for Improved Vision-Language Model: Instruction Tuning Enables Zero-Shot Conditional Image Representations

5.1 Introduction

In recent years, vision foundation models that are pretrained with large-scale datasets [Dosovitskiy, 2020; Chen et al., 2022; Radford et al., 2021b; Schuhmann et al., 2022a] have become the cornerstone for visual feature extraction, powering downstream applications ranging from classification [Dosovitskiy, 2020], segmentation [Caron et al., 2021], retrieval [Radford et al., 2021b], to multimodal large language models (MLLMs) [Ramesh et al., 2021; Li et al., 2022a; Liu et al., 2024a; Reid et al., 2024; McKinzie et al., 2024; Driess et al., 2023]. Despite the variety of pretraining schemes [Radford et al., 2021b; Caron et al., 2021; He et al., 2022; Oquab et al., 2023; El-Nouby et al., 2024], most commonly used vision foundation models, such as CLIP [Radford et al., 2021b], are designed to encode the rich information contained in (a patch of) an image into a single feature vector, wherein this *general* feature representation is expected to encapsulate all information that may be leveraged by various potential downstream tasks.

However, by aiming to extract *general-purpose* features that can serve as many downstream tasks as possible, image representations obtained from these task-agnostic vision foundation models may inevitably

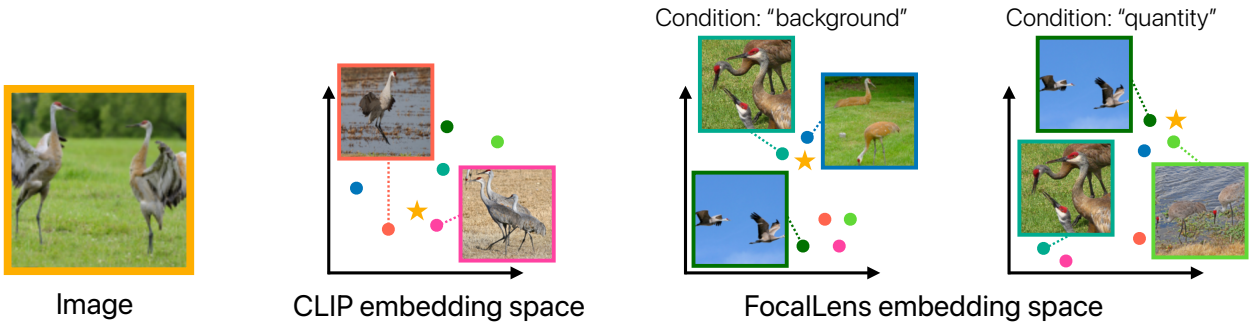


Figure 5.1: For a given image, the CLIP embedding space is static and structured based on overall semantics. However, FocalLens dynamically rearranges the embedding space based on the specified condition, bringing instances that are more similar under that condition closer together. We show the top-2 nearest neighbors for both CLIP and FocalLens embeddings (once conditioned on "background" and once on "quantity").

compromise relevant information that is *specific* to the downstream task of interest. For instance, CLIP models are known to produce image representations that capture the high-level semantics well [Radford et al., 2021b; Ramesh et al., 2021], but often struggle with understanding the finer-grained details and intrinsics of the image, such as attribute associations, spatial relationships, camera perspective, and so on [Vaze et al., 2023; Hsieh et al., 2024b; Tong et al., 2024b].

In this chapter, instead of aiming to learn a model that produces a fixed image representation in fulfilling different goals, we consider learning an *adaptive* vision foundation model that encodes an image differently conditioned on the downstream task of interest, allowing the resultant image representations to prioritize information relevant to the specified condition over other available semantics. Furthermore, as opposed to pre-defining the downstream tasks in a priori [Salehi et al., 2024; Wu et al., 2021], our goal is an *adaptive generalist* model that is able to adapt to broad potential use cases in a *zero-shot* fashion. Specifically, we consider utilizing free-form natural language texts as a rich and flexible interface to condition [1] the model given different downstream purposes, inspired by recent literature [Wei et al., 2021; Su et al., 2022; Liu et al., 2024a]. For instance, given a task of retrieving images of similar background scene to a given query image, by specifying through the text condition: "*What is the background in the image?*", we expect to guide the model in focusing more on the background features of the image, as illustrated in Figure 5.1.

We introduce *FocalLens*, a contrastive finetuning framework that transforms a pretrained vision-language model (VLM) into a text-conditioned vision encoder that is able to produce visual representations with better

[1] We use "condition (conditional)" and "adapt (adaptive)" interchangeably in this paper.

"focus" on the information relevant to the given instructions. Specifically, leveraging visual instruction tuning dataset [Dai et al., 2023; Liu et al., 2024a], in the format of (`instruction`, `image`, `output`), FocalLens aligns the visual representation of `image` to better adhere to `instruction`, using the corresponding `output` to guide the alignment. To demonstrate this approach, we apply FocalLens to representative pretrained MLLM and vision encoder: LLaVA [Liu et al., 2024a] and CLIP [Radford et al., 2021b], and name the resultant text-conditioned vision encoder models *FocalLens-MLLM* and *FocalLens-CLIP* respectively, as illustrated in Figure 5.2.

Through extensive evaluations on over 60 tasks, we observe that FocalLens models demonstrate a strong ability to condition representations based on the given text instructions, significantly outperforming existing baselines like CLIP. On average, FocalLens achieves up to 9 points higher performance, with even greater improvements on specific tasks, for image-image retrieval tasks. In addition, when used in downstream applications, FocalLens's conditional image representations further lead to clear gains compared to existing baselines. For instance, on image-text retrieval benchmarks, we show an average improvements of 5 and 10 points respectively on SugarCrepe [Hsieh et al., 2024b] and MMVP-VLM [Tong et al., 2024b], comparing favorably to other CLIP models that are much larger (up to 2.5×) in size. On image classification, FocalLens also shows superior performances than CLIP, especially in low-data regime. Finally, further qualitative study showcases various intriguing application scenarios that can be supported by FocalLens.

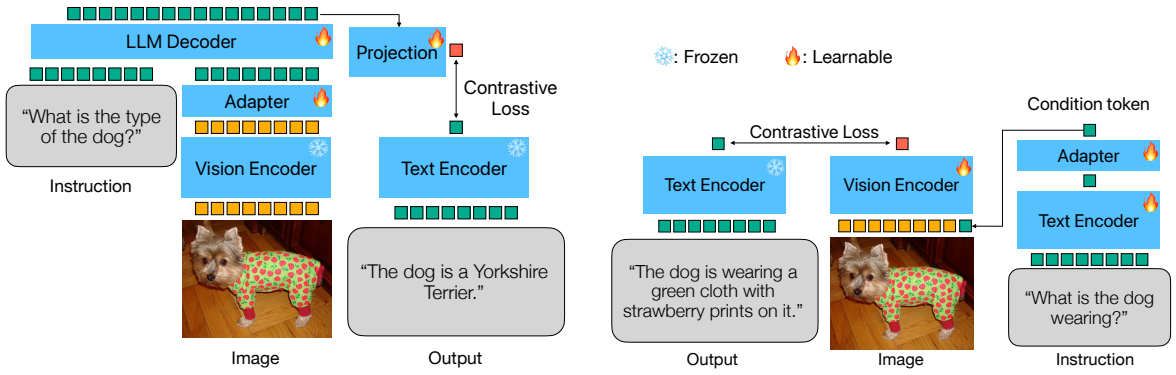
5.2 Related work

Foundation models for vision encoding. Modern vision foundation models trained on web-scale datasets [Dosovitskiy, 2020; Jia et al., 2021; Schuhmann et al., 2022a; Oquab et al., 2023] are used as the common underlying visual feature extractor to produce image representations that drive various downstream applications [Radford et al., 2021b; Ramesh et al., 2021; Kirillov et al., 2023; Zhou et al., 2022]. While there are many pretraining objectives [Oquab et al., 2023; He et al., 2022; El-Nouby et al., 2024; Radford et al., 2021b], existing schemes typically train the vision models to produce a single "general" image representation that hopefully captures all relevant information contained in the given image, or utilize information derived from diverse captions to help learning more discriminative image features [Lavoie et al., 2024]. Nonetheless, as an image naturally contains rich and dense information, a fixed and general-purpose representation may

not sufficiently pronounce information relevant to specific downstream contexts of interest [Kar et al., 2024; Wang et al., 2024a; Tong et al., 2024b; Hsieh et al., 2024b]. Our work aims to learn vision encoder that is capable of extracting different representations from a single image conditioned on downstream use cases at test-time, different from universal image embedding approaches that aim to learn a universal model for different domains without explicit conditioning [Google Research, 2023; Ypsilantis et al., 2023].

Conditional vision representations. Implicit and task-specific conditioning of visual features have been studied in the literature [Liu et al., 2024a; Dai et al., 2023; Tong et al., 2024a; Eftekhar et al., 2023; Vani et al., 2024; Chameleon Team, 2024]. For instance, the hidden representations in MLLMs may be interpreted as a type of conditional image representation, where the visual features are fused with text instructions for producing different output responses. Nonetheless, conditional visual representations considered in prior work are designed specifically to their model and respective applications, *e.g.*, generative conversations [Dai et al., 2023] and embodied AI [Eftekhar et al., 2023]. In this chapter, we are interested in conditional visual representations that may be used for various downstream applications, such as classifications, image-image or image-text retrieval.

Vision-language joint representation learning. There is a rich literature in vision-language (joint) representation learning [Lu et al., 2019; Li et al., 2019; Kim et al., 2021; Radford et al., 2021b; Jiang et al., 2024a]. Our work is related as we aim for a model that can comprehend both images and natural language conditions. Concurrent to ours, recent works [Jiang et al., 2024a,b] consider MLLM's output space as a universal representation space for both vision and language inputs. Nonetheless, in addition to the MLLM-based approach, we study an alternative promising CLIP-based approach with comprehensive analysis which leads to various performance benefits. Relatedly, composed image retrieval [Wu et al., 2021; Saito et al., 2023; Zhang et al., 2024] considers developing models of underlying similar capabilities that generate image embeddings given both image and text. However, different from our goal to use text conditioning to extract downstream-specific *intrinsic* visual features, their goal is to *extrinsically* "compose" semantics from both texts and images, largely towards image-retrieval purposes.



(a) FocalLens-MLLM **(b)** FocalLens-CLIP

Figure 5.2: FocalLens is applied to two vision-language models to extract text-conditioned visual features: **(a)** modifying Llava-like VLMs, which already have text-conditioning capabilities, to produce a global visual feature, and **(b)** modifying ViT [Dosovitskiy, 2020] based CLIP-like VLMs, which already produce a global visual feature, to condition their output feature based on a text condition.

5.3 Conditional embeddings via instruction contrastive tuning

Our goal is to develop an adaptive vision foundation model that is capable of encoding an image into tailored embeddings conditioned on the downstream task of interest, as specified through natural language texts.

We consider the visual instruction tuning data [Liu et al., 2024a], which covers diverse tasks, and has demonstrated great generalization of MLLMs in different benchmarks. The visual instruction tuning data is in the triplet format of (`image`, `instruction`, `output`). For instance, given an image of "a Yorkshire Terrier wearing a green cloth", the `output` is "The dog is wearing a green cloth with strawberry prints on it" with the `instruction` "What is the dog wearing?". Alternatively, when the instruction is "What is the type of the dog", the output is "The dog is a Yorkshire Terrier" correspondingly. MLLMs [Dai et al., 2023; Liu et al., 2024a] leverage the triplet instruction tuning for text generation: given (`image`, `instruction`), generating `output`. Instead, we propose to utilize contrastive learning [Radford et al., 2021b] on the triplet instruction tuning data. Specifically, given an image encoder conditioned on the *instruction*, we match the output embedding with a text embedding of *output*. We call the proposed method as *FocalLens*, which leverage instruction tuning data to *contrastively* tune the pretrained image encoder, such that it can better focus on desired information and generalize to diverse downstream tasks. We explore tuning two different representative vision-language models with FocalLens: MLLMs (Sec. 5.3.1) and CLIP (Sec. 5.3.2).

5.3.1 FocalLens with MLLMs

MLLMs [Liu et al., 2024a; Dai et al., 2023] generate textual responses regarding an image based on the given input text instructions. Given (`instruction`, `image`), the goal is to generate `output`. However, as the original model objective is text generation rather than producing explicit representation for downstream tasks, the conditional visual information may be dispersed throughout the model, and there is no direct access to them by design.

In FocalLens, instead of training the MLLM to generate `output` given (`image`, `instruction`) as in the original auto-regressive objective, we append a special indicator token `<eos_token>` to MLLM's input sequence, and consequently train the indicator's output token to align with the CLIP text embedding of the targeted output in a constrative manner. Here, we use an off-the-shelf frozen CLIP text encoder to obtain the target output embedding. With the contrastive objective, we encourage the model to condense information relevant to the image, with the specified instructions, into a single output representation. We show the overall model architecture in Figure 5.2a.

5.3.2 FocalLens with CLIP

Unlike MLLMs, CLIP models by design generate image representations [Radford et al., 2021b], where these image embeddings are already widely utilized in a variety of downstream tasks [Ramesh et al., 2021; Liu et al., 2024a]. However, CLIP models are inherently limited to producing a fixed representation for each image, regardless of the downstream task of interest. Although strong in capturing high-level semantics, these general visual features are shown to lack various aspects of fine-grained image details that can be critical for downstream tasks [Hsieh et al., 2024b; Tong et al., 2024b]. To tackle this, we propose to make CLIP's vision encoder task-aware, such that it is able to adapt its representations based on specific requirements, thereby capturing specific aspects of the image essential for different applications.

To incorporate natural language instructions into CLIP's vision encoder, we consider first converting `instruction` into a "condition text embedding", which is then treated as an additional token that is fed into the image encoder alongside the standard image tokens and the CLS token. Afterwards, the model is trained as in standard CLIP using a contrastive loss, aligning the resultant text-conditioned image representations with their corresponding textual outputs. By instruction tuning, we aim to allow the vision encoder to generalize

to a broad range of scenarios of interest that can be described via natural language at test-time [Wei et al., 2021; Su et al., 2022]. We illustrate the FocalLens-CLIP training setup in Figure 5.2b.

5.4 Experiments

In this section, we first demonstrate the benefits of conditional image representations (Sec. 5.4.1) over the generic representations produced by CLIP, using a toy dataset. We then extensively evaluate FocalLens models' capability in characterizing downstream conditions on a variety of tasks, compared to existing baselines (Sec. 5.4.2). By zooming in on FocalLens-CLIP, we demonstrate that its conditional image representations improve performance across a range of downstream tasks, including image-text retrieval, image classification, and image-image retrieval (Sec. 5.4.4).

Setup. We train FocalLens models with the visual instruction tuning data used in LLaVA [Liu et al., 2024a]. The dataset contains around 150k examples, wherein 60k examples are multi-turn conversations and thus can be treated as multiple triplets of (`image`, `instruction`, `output`), where the image remains the same. During training, we expand conversation data within batches to encourage models to output different representations given the same image but different instructions. For FocalLens-MLLM, we follow the training recipe of LLaVA [Liu et al., 2024a] to obtain a base MLLM before further training with the proposed contrastive loss. For FocalLens-CLIP, we initialize the base CLIP model with OpenAI's CLIP-ViT-L-14-336 [Radford et al., 2021b], which is also the underlying vision encoder used in LLaVA. We initialize the additional text encoder for instructions to have the same weight as the original text encoder.

For contrastive instruction tuning, given a batch of triplet instruction data $(\boldsymbol{x}_{\text{img}}^{(i)}, \boldsymbol{x}_{\text{ins}}^{(i)}, \boldsymbol{y}^{(i)})$, where $\boldsymbol{y}^{(i)}$ is the expected output for sample i, we form the pair-wise similarity matrix S, such that

$$S_{i,j} = \phi(\boldsymbol{x}_{\text{img}}^{(i)}, \boldsymbol{x}_{\text{ins}}^{(i)})^T \mathcal{T}(\boldsymbol{y}^{(j)}), \tag{5.1}$$

where ϕ is the encoding process that produce the conditional image embedding from both image $\boldsymbol{x}_{\text{img}}$ and instruction $\boldsymbol{x}_{\text{ins}}$, and $\mathcal{T}$ is the (frozen) text encoder that generates the target embedding from $\boldsymbol{y}$. We apply scaled Softmax to the rows of similarity matrix and compute the contrastive loss following CLIP [Radford

et al., 2021b]. We report further training details in Sec. D.3. In addition, we report all prompts used for conditioning FocalLens models during evaluation in Sec. D.4.

Image-image retrieval as an evaluation protocol. We consider the common image-image retrieval evaluation to measure the quality of image representations produced from different vision encoders [Google Research, 2023; Caron et al., 2021]. Specifically, given a query image, image-image retrieval tasks the model to retrieve other images from a gallery that are "similar" to the query image. We are especially interested in the scenario wherein the very definition of "similar" changes as the downstream tasks vary [Vaze et al., 2023]. To facilitate such evaluations, we adopt datasets where we may define various similarities between images based on *test-time* interest determined through a text condition. We introduce these datasets in the following sections. For each dataset, when not otherwise specified, we report mean Average Precision (mAP) as the evaluation metric.

5.4.1 Conditional representations better characterize task-specific details

We empirically validate the benefits of having the flexibility to encode an image based on the given condition of interest over using a fixed representation when downstream purpose varies, as considered in most prevailing vision encoding paradigms [Radford et al., 2021b; Caron et al., 2021]. Here, we restrict ourselves to a toy dataset to demonstrate the idea, and we shall expand our studies in the following sections.

A toy ColorShape dataset. ColorShape is a synthetic dataset where each image contains a certain colored shape. There are in total 4 different colors and shapes respectively. We generate 500 different images with random position and size of the object for each combination of color and shape. At test-time, we may define the intent for retrieval based on different aspects. Specifically, we may group each image into different categories based on either only its color, only its shape, or both. Figure 5.3 shows some examples from the dataset.

The pretrained CLIP model [Radford et al., 2021b] serves as the standard encoder baseline where the image representations are fixed even when the test-time condition varies. For the conditional vision encoders, we consider both FocalLens-MLLM and FocalLens-CLIP models discussed in Sec. 5.3. We show their retrieval performances on the ColorShape dataset when the test-time condition varies.

Non-adapative image representations overlook specific aspects of images. From Table 5.1, on

the simple ColorShape dataset, CLIP yields almost perfect retrieval performances when we define image categories based on both color and shape. However, in the context where we are specifically interested in categorizing images based only on the color, CLIP's performance drops significantly to 57 mAP point. On the other hand, when we define similarity based only on shape, CLIP achieves relatively better performances at 90 mAP point. Combining the results, while CLIP can produce *general* representation that is strong at grouping objects of certain shape and color together, its overall representation space is biased towards the "shape" of objects, and much less discriminative over the "color" aspect. This also echos the observations made in recent works [Tong et al., 2024b; Hsieh et al., 2024b], suggesting that CLIP's representation, while powerful for general tasks, may overlook fine-grained details such as color, highlighting a need for approaches to better adapt and capture the nuanced visual characteristics, depending on the task at hand.

Conditional image representations better capture information relevant to the downstream task. In Table 5.1, as opposed to CLIP model, the conditional image representations produced from both adaptive vision encoders, the MLLM-based and the CLIP-based model, achieve much more balanced (and superior) results than CLIP's representation when the downstream condition varies. When averaged across three different scenarios ("color", "shape", and "both"), both conditional vision encoders improve over 10 mAP point compared to CLIP. The conditional CLIP-based model also always outperforms CLIP, when evaluated separately on the three respective conditions.

In addition to using discrete color labels (e.g., "red", "blue") to define image similarity, we also consider a more sophisticated setup where image similarity is measured based on L2 distance in RGB space. Specifically, in this Continuous Color variant, we assign randomly sampled RGB colors to the objects. During evaluation, our goal is to retrieve images with colors closer to that of the query image. We compute the rank correlation between the similarity measured in the model's image representation space and the ground-truth similarity defined in RGB space. In this setup, both FocalLens models significantly outperform CLIP as show in the last column of Table 5.1.

5.4.2 FocalLens improves image representations across benchmarks

Using the ColorShape toy dataset, we validated the benefits of adapting image representations for downstream tasks. We now compare FocalLens to existing vision encoders and relevant baselines across a comprehensive

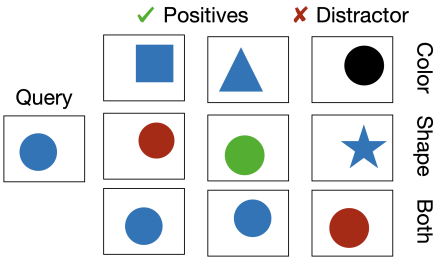


Figure 5.3: ColorShape examples with a query image, three conditions, and corresponding positives and distractors.

Model	ColorShape				Cont. Color
	Color	Shape	Both	Avg.	
CLIP (task-agnostic)	57.10	90.24	99.36	82.23	0.158
FocalLens-MLLM	**99.94**	82.56	98.92	**93.80**	**0.560**
FocalLens-CLIP	87.28	**93.51**	**99.99**	93.59	0.405

Table 5.1: Image-image retrieval results on ColorShape dataset. Conditional representations from FocalLens better capture the given conditions compared to the task-agnostic representations of CLIP.

set of evaluation benchmarks.

Evaluation benchmarks. We consider a total of 49 different tasks across 4 coarse-grained categories in our evaluation suite as briefly described below. We include dataset details in Sec. D.1.

- **CelebA-Attribute** [Liu et al., 2015]: CelebA is a dataset consisting of celebrity face images. Each face image is associated with various properties spanning from the hair color of the person, the eyebrow shape, to whether the person is wearing eyeglasses, and so on. We vary the downstream condition of interest across different properties for retrieval. For instance, when conditioned on "eyeglasses" with a query image showing a person is (not) wearing eyeglasses, the model is tasked to retrieve other face images with (without) eyeglasses. We manually select a total of 29 different properties that can be objectively labeled, and exclude more subjective properties such as "attractiveness" or "young". We notice that the class within each attribute may be imbalanced, resulting in high mAP even with random guess. We thus report scaled performances w.r.t. random guess by: $\frac{p-r}{1-r}$, where p is the original mAP and r is the random guess mAP.

- **GeneCIS** [Vaze et al., 2023]: GeneCIS presents various image retrieval tasks for evaluating conditional image similarity. Given a query image ("a white laptop") and a condition ("color"), the goal is to retrieve the most similar image (another "white laptop") from a gallery that contains implicitly similar distractors with wrong conditions (e.g., "a black laptop"). We report the "Focus attribute" and "Focus object" tasks from GeneCIS. As each query image contains only a single positive in the gallery, we report Recall@3 following prior work [Zhang et al., 2024].

Table 5.2: Results on CelebA-Attribute and GeneCIS.

	CelebA-Attribute					GeneCIS		
Model	Blond Hair	Smiling	Wavy Hair	Lipstick	Avg. 29 tasks	Attribute	Object	Avg.
CLIP	6.20	8.68	7.54	41.45	13.59	43.10	25.81	34.46
InstructBLIP	21.03	21.71	13.91	34.64	16.19	**47.00**	34.03	40.52
MagicLens	8.24	9.98	10.76	54.12	13.42	39.00	35.50	37.25
FocalLens-MLLM	25.76	**34.43**	**17.61**	**68.07**	**22.67**	45.35	30.20	37.78
FocalLens-CLIP	**32.22**	22.11	16.89	62.50	21.32	**43.30**	**43.72**	**43.51**

Table 5.3: Results on ImageNet-Subset and fine-grained classification datasets.

	ImageNet-Subset					Fine-grained classification datasets				
Model	Ball	Cat	Dog	Fish	Avg. 14 tasks	Flower	Car	Aircraft	Food	Avg.
CLIP	64.63	53.00	16.55	61.79	51.03	**83.87**	45.14	**25.96**	58.66	53.41
InstructBLIP	66.44	51.22	9.60	59.16	47.67	80.26	25.97	13.47	54.32	43.51
MagicLens	68.10	50.14	17.28	58.84	46.36	74.88	23.95	17.55	**65.13**	45.38
FocalLens-MLLM	**78.99**	53.24	29.25	57.40	52.34	43.92	18.59	14.73	50.93	32.04
FocalLens-CLIP	70.01	**56.80**	**33.15**	**65.37**	**55.29**	80.23	**54.72**	21.44	64.16	**55.14**

- **ImageNet-Subset** [Deng et al., 2009]: In addition to the above benchmarks with specific downstream conditions of interest, we as well evaluate our models on standard ImageNet classes, where the condition corresponds to the image "classes" as defined by ImageNet. Specifically, we create 14 different retrieval sub-tasks based on coarse-grained categories from WordNet [Miller, 1995] hierarchy (e.g., ball, bird, dog, etc.). In each task (e.g., dog), the goal is to retrieve images (from all dog images) with the same type of instance (same breed of dog) as the query image.

- **Fine-grained classification datasets**: Similar to ImageNet, we incorporate 4 finer-grained classification datasets, including Oxford Flowers [Nilsback and Zisserman, 2008], Stanford Cars [Krause et al., 2013], FGVC Aircraft [Maji et al., 2013], and Food-101 [Bossard et al., 2014].

Baselines. We consider CLIP [Radford et al., 2021b] as the task-agnostic vision encoder model. We also compare to models that are able to generate conditional visual representations, including the Q-former used in InstructBLIP [Li et al., 2023d; Dai et al., 2023], and MagicLens [Zhang et al., 2024] that is designed specifically for composed image-retrieval with open-ended instructions. We include details of the baselines in Sec. D.2.

FocalLens improves significantly over existing baselines given specific downstream conditions. From Table 5.2, both variants of FocalLens provide significant gains over the task-agnostic CLIP baseline on CelebA-Attribute and GeneCIS, when there are specific conditions to respect. We see an overall gain of 9 points on CelebA-Attribute. Looking more closely at the individual conditions on CelebA-Attribute (complete results reported in Sec. D.5), we observe that when the condition of interest is "smiling", we see a significant gap of 26 points between CLIP and FocalLens, where the gap is as large as 48 points on certain attributes. Similarly on the GeneCIS benchmark, by specifying the attribute such as color or certain object to focus on, FocalLens improves over CLIP by an average of 9 points.

On CelebA-Attribute and GeneCIS, we also see FocalLens models demonstrate outperforming (or favorable) results when compared to prior task-aware vision encoders (i.e., InstructBLIP and MagicLens), that are also given the downstream condition of interest when generating the image representations. Specifically, FocalLens-CLIP achieves the best overall performances, winning over the stronger InstructBLIP baseline by 5 and 3 points respectively on CelebA-Attribute and GeneCIS, validating the effectiveness of our proposed strategy.

FocalLens maintains or improves over existing baseline on generic conditions. In Table 5.3, we compare model performances on ImageNet-Subset and the fine-grained classification datasets, where the downstream goal is generic instance classification. First, CLIP model demonstrates competitive performances on both ImageNet-Subset and fine-grained classification tasks, showing that its embeddings are indeed strong at representing generic features when it comes to standard "type" classification. In contrast, InstructBLIP and MagicLens suffer performance drops on both ImageNet-Subset and fine-grained tasks. On the other hand, we see FocalLens (especially FocalLens-CLIP) maintains comparable performances to CLIP on fine-grained datasets and attains even better performances on ImageNet-Subset. We explain the improvement on ImageNet by that conditioning FocalLens with instructions such as "What is the type of dog?" helps the model to better focus on the specific object of interest but not other potential distractors in the image (e.g., the "toy" besides the dog).

5.4.3 Comparative analysis of FocalLens variants

Both FocalLens-MLLM and FocalLens-CLIP yield promising results in the experiments. One major difference between FocalLens-MLLM and FocalLens-CLIP is their underlying pretrained models' output modality. Specifically, the original MLLM model in FocalLens-MLLM is trained to autoregressively produce textual outputs, while CLIP's vision encoder is trained to produce image embeddings. We are thus interested in understanding whether this difference affects the underlying characteristics of the output representations in FocalLens-MLLM and FocalLens-CLIP.

To test this, we consider downstream conditions that require visual features beyond semantic concepts that are describable by text. In particular, on CelebA, instead of considering conditions such as whether the person is wearing glasses or not, which is answerable in simple words ("yes" or "no"), we consider a *fuzzy* condition where the image similarity is defined by the *identity* of the person. Textual representations that do not carry visual information may fail at achieving good performance on this task, as identity is hardly describable through natural language.

In Table 5.4, we observe that FocalLens-MLLM suffers from a clear performance gap compared to FocalLens-CLIP. This suggests that FocalLens-MLLM may rely more on MLLM's original textual output modality, which is limited for tasks requiring rich visual information. Similar observations are also hinted by its relatively low performance on fine-grained classification results in Table 5.3. In contrast, FocalLens-CLIP, with its underlying model being a vision encoder, is better suited for tasks requiring richer visual detail. Based on this observation, we focus on FocalLens-CLIP for the remainder of the experiments.

Table 5.4: Comparison between FocalLens-MLLM and FocalLens-CLIP on fuzzy conditions with CelebA-Identity.

Model	CelebA-Identity
FocalLens-MLLM	14.48
FocalLens-CLIP	**46.84**

5.4.4 FocalLens representations improve downstream applications

In addition to evaluations based only on image representations, we show how image representations produced from FocalLens-CLIP can drive improvement on downstream tasks including image-text retrieval and image

Table 5.5: Image-Text Retrieval on SugarCrepe for vision-language compositionality evaluation.

	SugarCrepe							
Model	Replace-obj	Replace-att	Replace-rel	Swap-obj	Swap-att	Add-obj	Add-att	Avg.
OpenAI ViT-L-14 [Radford et al., 2021b]	94.49	80.58	66.78	64.08	62.46	80.74	74.27	74.77
OpenAI RN50x64 [Radford et al., 2021b]	94.49	83.50	70.63	61.79	66.67	83.27	73.99	76.33
LAION ViT-g-14 [Schuhmann et al., 2022a]	**95.76**	**85.03**	72.40	63.01	**71.17**	**91.51**	82.08	**80.14**
FocalLens-CLIP	95.64	84.51	**75.53**	**65.30**	66.36	86.12	**83.09**	79.51

Table 5.6: Image-Text Retrieval on MMVP-VLM.

	MMVP-VLM									
Model	Orientation	Presence	State	Quantity	Spatial	Color	Structure	Text	Camera	Avg.
OpenAI ViT-L-14 [Radford et al., 2021b]	6.7	20.0	26.7	6.7	13.3	33.3	**46.7**	20.0	13.3	20.7
MetaCLIP ViT-H-14 [Xu et al., 2023a]	6.7	13.3	**60.0**	13.3	6.7	53.3	26.7	13.3	**33.3**	25.2
EVA01 ViT-g-14 [Sun et al., 2023a]	6.7	26.7	40.0	6.7	13.3	**66.7**	13.3	13.3	20.0	23.0
FocalLens-CLIP	6.7	**33.3**	33.3	**40.00**	**26.7**	**66.7**	20.0	**26.7**	20.0	**30.4**

classification in a low-data regime where only a small amount of downstream task data is available for training.

Image-text retrieval. A prevailing usage of image representations is to enable cross-modality retrieval. Here, we include two image-text prediction benchmarks, where the goal is to predict the correct textual description of a given image. Specifically, we adopt SugarCrepe [Hsieh et al., 2024b] and MMVP-VLM [Tong et al., 2024b]. SugarCrepe presents challenging hard-negative text distrators along with a positive description for the model to select from, where existing models are shown to struggle with. Similarly, MMVP-VLM particularly collects examples with visual patterns where CLIP vision encoder are shown to fall short.

In Table 5.5 on SugarCrepe, we compare FocalLens-CLIP to several standard CLIP models of different sizes, and trained with different data sizes. First, compared to the underlying CLIP model used in FocalLens-CLIP (i.e., OpenAI ViT-L-14), FocalLens-CLIP achieves around 4.7 point improvements on average, with consistent improvements across all different sub-tasks with individual gains up to 9 points on Replace-rel and Add-att. Interestingly, the two sub-tasks test the model's capability in understanding fine-grained relationships and attributes in the image, where standard CLIP models struggle the most [Hsieh et al., 2024b]. This suggests FocalLens-CLIP's image representations are able to better characterize fine-grained visual details. Furthermore, by scaling up the model size from 428M to 623M, the RN50x64 model still underperform our smaller FocalLens-CLIP model (551M for both image and text encoders). On the other hand, FocalLens-CLIP shows competitive performances compared to the 2.5$\times$ bigger ViT-g-14 model trained

on 5× more data.

From Table 5.6 on MMVP-VLM, we see FocalLens-CLIP significantly outperforms the baseline ViT-L-14 model consistently across all sub-tasks, by an average of 9.7 points. Furthermore, we note that our FocalLens-CLIP model also compares favorably to the much larger ViT-H-14 (1.8× larger) and ViT-g-14 (2× larger) on individual sub-tasks, where FocalLens-CLIP achieves the best overall performance with a lead of 5.2 point.

Linear probing in low-data regime. We evaluate the performance of FocalLens-CLIP in a linear probing setup, where only a small amount of downstream task data is available for training. We use the largest dataset in ImageNet-Subset introduced in Sec. 5.4.2, focusing on different dog breeds (a total of 118 classes). In the low-data setup [Henaff, 2020; Luo et al., 2017; Vemulapalli et al.], we assume there are k instances available for each class for training and consider $k = 5, 10, 15$. We freeze the backbone and replace the CLIP projection layer with a linear layer to perform 118-way classification. The linear probe is trained for 100 epochs following prior works like [Liu et al., 2024b]. We sweep over learning rates from 1e-2 to 1e-4 in steps of 2.5e-3 and report the performance of the best checkpoint. We compare FocalLens-CLIP to OpenAI ViT-L-14 in this setup, as shown in Figure 5.4. In the extreme setting, where only 5 instances per class is available to train the linear probe, FocalLens-CLIP outperforms CLIP-ViT-L by 5.3%. This result further reinforces our observation that conditional image representations are more efficient in extracting information relevant to downstream tasks.

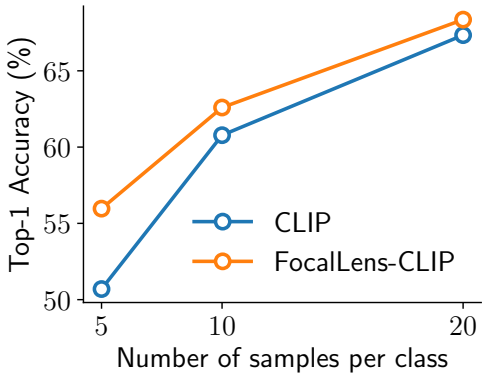


Figure 5.4: Linear probing results comparing CLIP and FocalLens-CLIP.

Qualitative analysis on conditional image-retrieval. We qualitatively compare the top-k images retrieved by using FocalLens-CLIP's conditional image embeddings with those retrieved by standard CLIP, specifically when given various downstream conditions. For this qualitative study, we treat all images in the 14 coarse-grained categories considered in ImageNet-Subset as the gallery for retrieval. In Figure 5.5, we showcase several intriguing examples across various aspects of conditioning FocalLens-CLIP captures. In the top-left example, we consider a scenario where we are interested in retrieving images of similar background

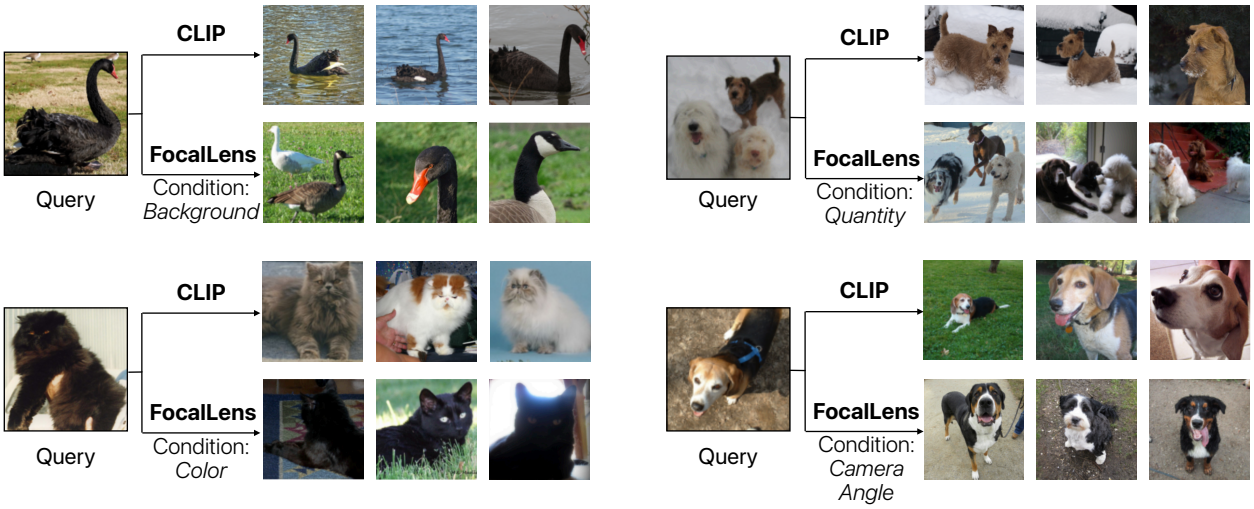


Figure 5.5: Comparison between CLIP and FocalLens-CLIP on conditional image retrieval.

to the query image. Given the query image of "a goose on a grassy field", although the images retrieved by CLIP do all contain goose, all images have the background of water instead of grassy field. Conversely, we see images retrieved by FocalLens-CLIP all have similar grassy background as expected. Similarly, in the top-right, we see FocalLens-CLIP faithfully reflects the interested condition of quantity, retrieving images with 3 dogs as in the query image, whereas images retrieved by CLIP is largely based on their instance type (same species of dog), and cannot reflect the downstream interest. More examples demonstrate that color or even implicit visual features such as camera angle can also be characterized by FocalLens-CLIP.

5.5 Discussion

In this chapter, we introduced FocalLens, a zero-shot conditional visual embedding model that focuses the representation on specific aspects of the image described in the given text. FocalLens is trained using existing visual instruction tuning datasets to align the conditional image representation with the textual description. Experiments on a comprehensive set of tasks, including image-to-image retrieval, image classification, and image-to-text retrieval, demonstrate that FocalLens matches or exceeds the performance of state-of-the-art models.

Limitations. Although experiments demonstrate that FocalLens can be effectively trained using visual instruction tuning datasets, model performance could be enhanced by designing customized datasets for this

task, which we leave for future study. Moreover, the relatively small scale of the visual instruction tuning datasets may hinder alignment accuracy for highly specialized concepts that are entirely absent from the dataset.

Chapter 6

Conclusion

This thesis addresses two central challenges in the development of modern large-scale AI systems: the efficient deployment and adaptation of large models, and the effective curation of data for evaluating and improving model capabilities. The work spans both the model and data axes across two major modalities—natural language processing and computer vision.

On the model side, we propose practical strategies to deploy and adapt large language models (LLMs) in resource-constrained settings. Through methods such as distillation and attention calibration, the thesis demonstrates how smaller models can achieve strong performance without costly data annotations, and how models can effectively adapt to new knowledge domains leveraging external input contexts.

On the data side, we focus on vision-language models (VLMs) and introduce approaches to benchmark and improve their compositional reasoning abilities. We reveal limitations in existing benchmarks, and present a new dataset for more reliable evaluation. Furthermore, by strategically leveraging vision instruction tuning data, we propose a new vision-language model training framework that produces context-aware image representations able to adapt to different downstream needs, ultimately providing ubiquitous performance gains on various tasks.

Together, these studies offer a holistic view of how to scale AI systems in a more efficient, effective, and reliable manner. By addressing challenges at the intersection of model and data, we envision this thesis contributes toward making powerful AI more accessible across real-world applications.

Bibliography

Priyanka Agrawal, Chris Alberti, Fantine Huot, Joshua Maynez, Ji Ma, Sebastian Ruder, Kuzman Ganchev, Dipanjan Das, and Mirella Lapata. 2022. Qameleon: Multilingual qa with only 5 examples. *arXiv preprint arXiv:2211.08264*.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

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*.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew R Gormley. 2023. Unlimiformer: Long-range transformers with unlimited length input. *arXiv preprint arXiv:2305.01625*.

Lucas Beyer, Xiaohua Zhai, Amélie Royer, Larisa Markeeva, Rohan Anil, and Alexander Kolesnikov. 2022. Knowledge distillation: A good teacher is patient and consistent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10925–10934.

Federico Bianchi, Pratyusha Kalluri, Esin Durmus, Faisal Ladhak, Myra Cheng, Debora Nozza, Tatsunori Hashimoto, Dan Jurafsky, James Zou, and Aylin Caliskan. 2022. Easily accessible text-to-image generation amplifies demographic stereotypes at large scale. *arXiv preprint arXiv:2211.03759*.

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of the ACL Workshop on Challenges & Perspectives in Creating Large Language Models*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2021. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. 2014. Food-101–mining discriminative components with random forests. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part VI 13*, pages 446–461. Springer.

Léon Bottou. 2014. From machine learning to machine reasoning. *Machine learning*, 94(2):133–149.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. *Advances in Neural Information Processing Systems*, 31.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660.

Paola Cascante-Bonilla, Khaled Shehada, James Seale Smith, Sivan Doveh, Donghyun Kim, Rameswar Panda, Gül Varol, Aude Oliva, Vicente Ordonez, Rogerio Feris, and Leonid Karlinsky. 2023. Going beyond nouns with vision & language models using synthetic data.

Chameleon Team. 2024. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*.

Howard Chen, Ramakanth Pasunuru, Jason Weston, and Asli Celikyilmaz. 2023a. Walking down the memory maze: Beyond context limit through interactive reading. *arXiv preprint arXiv:2310.05029*.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023b. How is chatgpt's behavior changing over time? *arXiv preprint arXiv:2307.09009*.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. 2020. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255.

Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. 2022. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*.

Jaemin Cho, Abhay Zala, and Mohit Bansal. 2022. Dall-eval: Probing the reasoning skills and social biases of text-to-image generative transformers. *arXiv preprint arXiv:2202.04053*.

Noam Chomsky and Morris Halle. 1965. Some controversial questions in phonological theory. *Journal of linguistics*, 1(2):97–138.

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*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*.

MJ Cresswell. 1973. Logics and languages.

Wenliang Dai, Junnan Li, D Li, AMH Tiong, J Zhao, W Wang, B Li, P Fung, and S Hoi. 2023. Instructblip: Towards general-purpose vision-language models with instruction tuning. arxiv 2023. *arXiv preprint arXiv:2305.06500*, 2.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, K. Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Anuj Diwan, Layne Berry, Eunsol Choi, David Harwath, and Kyle Mahowald. 2022. Why is winoground hard? investigating failures in visuolinguistic compositionality. *arXiv preprint arXiv:2211.00768*.

Yue Dong, Chandra Bhagavatula, Ximing Lu, Jena D Hwang, Antoine Bosselut, Jackie Chi Kit Cheung, and Yejin Choi. 2021. On-the-fly attention modulation for neural generation. *arXiv preprint arXiv:2101.00371*.

Alexey Dosovitskiy. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Sivan Doveh, Assaf Arbelle, Sivan Harary, Eli Schwartz, Roei Herzig, Raja Giryes, Rogerio Feris, Rameswar Panda, Shimon Ullman, and Leonid Karlinsky. 2023. Teaching structured vision & language concepts to vision & language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2657–2668.

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*.

Ainaz Eftekhar, Kuo-Hao Zeng, Jiafei Duan, Ali Farhadi, Ani Kembhavi, and Ranjay Krishna. 2023. Selective visual representations improve convergence and generalization for embodied ai. *arXiv preprint arXiv:2311.04193*.

Jacob Eisenstein, Daniel Andor, Bernd Bohnet, Michael Collins, and David Mimno. 2022. Honest students from untrusted teachers: Learning an interpretable question-answering pipeline from a pretrained language model. *arXiv preprint arXiv:2210.02498*.

Alaaeldin El-Nouby, Michal Klein, Shuangfei Zhai, Miguel Angel Bautista, Alexander Toshev, Vaishaal Shankar, Joshua M Susskind, and Armand Joulin. 2024. Scalable pre-training of large autoregressive image models. *arXiv preprint arXiv:2401.08541*.

Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. 2023. Eva: Exploring the limits of masked visual representation learning at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19358–19369.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. *arXiv preprint arXiv:2301.12726*.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. 2023. Datacomp: In search of the next generation of multimodal datasets. *arXiv preprint arXiv:2304.14108*.

Gemini Team. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Behrooz Ghorbani, Orhan Firat, Markus Freitag, Ankur Bapna, Maxim Krikun, Xavier Garcia, Ciprian Chelba, and Colin Cherry. 2021. Scaling laws for neural machine translation. *arXiv preprint arXiv:2109.07740*.

Google Research. 2023. Introducing the google universal image embedding challenge. Accessed: 2024-09-30.

Madeleine Grunde-McLaughlin, Ranjay Krishna, and Maneesh Agrawala. 2021. Agqa: A benchmark for compositional spatio-temporal reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*.

Braden Hancock, Antoine Bordes, Pierre-Emmanuel Mazare, and Jason Weston. 2019. Learning from dialogue after deployment: Feed yourself, chatbot! *arXiv preprint arXiv:1901.05415*.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2021. Self-attention attribution: Interpreting information interactions inside transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12963–12971.

Peter Hase and Mohit Bansal. 2021. When can models learn from explanations? a formal framework for understanding the roles of explanation data. *arXiv preprint arXiv:2102.02201*.

Junqing He, Kunhao Pan, Xiaoqun Dong, Zhuoyang Song, Yibo Liu, Yuxin Liang, Hao Wang, Qianguo Sun, Songxin Zhang, Zejian Xie, et al. 2023. Never lost in the middle: Improving large language models via attention strengthening question answering. *arXiv e-prints*, pages arXiv–2311.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009.

Olivier Henaff. 2020. Data-efficient image recognition with contrastive predictive coding. In *International conference on machine learning*, pages 4182–4192. PMLR.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2022. Large language models are reasoning teachers. *arXiv preprint arXiv:2212.10071*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Cheng-Yu Hsieh, Si-An Chen, Chun-Liang Li, Yasuhisa Fujii, Alexander Ratner, Chen-Yu Lee, Ranjay Krishna, and Tomas Pfister. 2023a. Tool documentation enables zero-shot tool-usage with large language models. *arXiv preprint arXiv:2308.00675*.

Cheng-Yu Hsieh, Yung-Sung Chuang, Chun-Liang Li, Zifeng Wang, Long Le, Abhishek Kumar, James

Glass, Alexander Ratner, Chen-Yu Lee, Ranjay Krishna, and Tomas Pfister. 2024a. Found in the middle: Calibrating positional attention bias improves long context utilization.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023b. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*.

Cheng-Yu Hsieh, Pavan Kumar Anasosalu Vasu, Fartash Faghri, Raviteja Vemulapalli, Chun-Liang Li, Ranjay Krishna, Oncel Tuzel, and Hadi Pouransari. 2025. Focallens: Instruction tuning enables zero-shot conditional image representations. *arXiv preprint arXiv:2504.08368*.

Cheng-Yu Hsieh, Jieyu Zhang, Zixian Ma, Aniruddha Kembhavi, and Ranjay Krishna. 2024b. Sugarcrepe: Fixing hackable benchmarks for vision-language compositionality. *Advances in neural information processing systems*, 36.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2022. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795.

Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. 2021. Openclip.

Fotis Iliopoulos, Vasilis Kontonis, Cenk Baykal, Gaurav Menghani, Khoa Trinh, and Erik Vee. 2022. Weighted distillation with unlabeled examples. In *Advances in Neural Information Processing Systems*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022a. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022b. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*.

Theo MV Janssen and Barbara H Partee. 1997. Compositionality. In *Handbook of logic and language*, pages 417–473. Elsevier.

Jingwei Ji, Ranjay Krishna, Li Fei-Fei, and Juan Carlos Niebles. 2020. Action genome: Actions as compositions of spatio-temporal scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10236–10247.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR.

Huiqiang Jiang, Qianhui Wu, , Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *ArXiv preprint*, abs/2310.06839.

Ting Jiang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and Fuzhen Zhuang. 2024a. E5-v: Universal embeddings with multimodal large language models. *arXiv preprint arXiv:2407.12580*.

Ziyan Jiang, Rui Meng, Xinyi Yang, Semih Yavuz, Yingbo Zhou, and Wenhu Chen. 2024b. Vlm2vec: Training vision-language models for massive multimodal embedding tasks. *arXiv preprint arXiv:2410.05160*.

He Junqing, Pan Kunhao, Dong Xiaoqun, Song Zhuoyang, Liu Yibo, Liang Yuxin, Wang Hao, Sun Qianguo, Zhang Songxin, Xie Zejian, et al. 2023. Never lost in the middle: Improving large language models via attention strengthening question answering. *arXiv preprint arXiv:2311.09198*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Oğuzhan Fatih Kar, Alessio Tonioni, Petra Poklukar, Achin Kulshrestha, Amir Zamir, and Federico Tombari. 2024. Brave: Broadening the visual encoding of vision-language models. *arXiv preprint arXiv:2404.07204*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

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.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. *arXiv preprint arXiv:2301.10226*.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 2013. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Samuel Lavoie, Polina Kirichenko, Mark Ibrahim, Mahmoud Assran, Andrew Gordon Wildon, Aaron Courville, and Nicolas Ballas. 2024. Modeling caption diversity in contrastive vision-language pretraining. *arXiv preprint arXiv:2405.00740*.

Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew Peters, Ashish Sabharwal, and Yejin Choi. 2020. Adversarial filters of dataset biases. In *International Conference on Machine Learning*, pages 1078–1088. PMLR.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Dacheng Li, Rulin Shao, Anze Xie, Ying Sheng, Lianmin Zheng, Joseph E. Gonzalez, Ion Stoica, Xuezhe Ma, and Hao Zhang. 2023a. How long can open-source llms truly promise on context length?

Dacheng Li, Rulin Shao, Anze Xie, Eric P Xing, Joseph E Gonzalez, Ion Stoica, Xuezhe Ma, and Hao Zhang. 2023b. Lightseq: Sequence level parallelism for distributed training of long context transformers. *arXiv preprint arXiv:2310.03294*.

G Li, N Duan, Y Fang, M Unicoder-VL Gong, D Jiang, and M Unicoder-VL Zhou. 2019. A universal encoder for vision and language by cross-modal pre-training. arxiv 2019. *arXiv preprint arXiv:1908.06066*.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023c. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023d. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022a. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. 2022b. BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 12888–12900. PMLR.

Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. 2023e. Symbolic chain-of-thought distillation: Small models can also" think" step-by-step. *arXiv preprint arXiv:2306.14050*.

Kevin J Liang, Weituo Hao, Dinghan Shen, Yufan Zhou, Weizhu Chen, Changyou Chen, and Lawrence Carin. 2020. Mixkd: Towards efficient distillation of large-scale language models. *arXiv preprint arXiv:2011.00593*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Alexander Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew Arad

Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue WANG, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri S. Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Andrew Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*. Featured Certification, Expert Certification.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024a. Visual instruction tuning. *Advances in neural information processing systems*, 36.

Jiacheng Liu, Wenya Wang, Dianzhuo Wang, Noah A Smith, Yejin Choi, and Hannaneh Hajishirzi. 2023a. Vera: A general-purpose plausibility estimation model for commonsense statements. *arXiv preprint arXiv:2305.03695*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023b. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*.

Xingbin Liu, Jinghao Zhou, Tao Kong, Xianming Lin, and Rongrong Ji. 2024b. Exploring target representations for masked autoencoders. In *International Conference on Learning Representations*.

Yang Liu, Yuanshun Yao, Jean-Francois Ton, Xiaoying Zhang, Ruocheng Guo Hao Cheng, Yegor Klochkov, Muhammad Faaiz Taufiq, and Hang Li. 2023c. Trustworthy llms: a survey and guideline for evaluating large language models' alignment. *arXiv preprint arXiv:2308.05374*.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. 2016. Visual relationship detection with language priors. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 852–869. Springer.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Zelun Luo, Yuliang Zou, Judy Hoffman, and Li F Fei-Fei. 2017. Label efficient learning of transferable representations acrosss domains and tasks. *Advances in neural information processing systems*, 30.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Zixian Ma, Jerry Hong, Mustafa Omer Gul, Mona Gandhi, Irena Gao, and Ranjay Krishna. 2022. Crepe: Can vision-language foundation models reason compositionally? *arXiv preprint arXiv:2212.07796*.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2022. Teaching small language models to reason. *arXiv preprint arXiv:2212.08410*.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. 2018. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. 2013. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*.

Brandon McKinzie, Zhe Gan, Jean-Philippe Fauconnier, Sam Dodge, Bowen Zhang, Philipp Dufter, Dhruti Shah, Xianzhi Du, Futang Peng, Floris Weers, et al. 2024. Mm1: Methods, analysis & insights from multimodal llm pre-training. *arXiv preprint arXiv:2403.09611*.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing english math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Smitha Milli, Ludwig Schmidt, Anca D Dragan, and Moritz Hardt. 2019. Model reconstruction from model explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 1–9.

Sewon Min, Suchin Gururangan, Eric Wallace, Hannaneh Hajishirzi, Noah A. Smith, and Luke Zettlemoyer. 2024. SILO language models: Isolating legal risk in a nonparametric datastore. In *ICLR*.

Sewon Min, Kalpesh Krishna, Xinxi 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. In *EMNLP*.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126.

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*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Maria-Elena Nilsback and Andrew Zisserman. 2008. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pages 722–729. IEEE.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*.

OpenAI. 2022. Chatgpt.

OpenAI. 2023. Gpt-4v(ision) system card.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. 2023. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*.

Dong Huk Park, Samaneh Azadi, Xihui Liu, Trevor Darrell, and Anna Rohrbach. 2021. Benchmark for compositional text-to-image synthesis. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

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.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, et al. 2020. Kilt: a benchmark for knowledge intensive language tasks. *arXiv preprint arXiv:2009.02252*.

Alexander Peysakhovich and Adam Lerer. 2023. Attention sorting combats recency bias in long context language models. *arXiv preprint arXiv:2310.01427*.

Ofir Press, Noah Smith, and Mike Lewis. 2021. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*.

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.

Mahima Pushkarna, Andrew Zaldivar, and Oddur Kjartansson. 2022. Data cards: Purposeful and transparent dataset documentation for responsible ai. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 1776–1826.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021a. Learning

transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021b. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

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(140):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, Florence, Italy. Association for Computational Linguistics.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

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.

Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Inbal Magar, Omri Abend, Ehud Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. Parallel context windows for large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6383–6402, Toronto, Canada. Association for Computational Linguistics.

Mathieu Ravaut, Shafiq Joty, Aixin Sun, and Nancy F Chen. 2023. On position bias in summarization with large language models. *arXiv preprint arXiv:2310.10570*.

Arijit Ray, Filip Radenovic, Abhimanyu Dubey, Bryan A. Plummer, Ranjay Krishna, and Kate Saenko. 2023. Cola: How to adapt vision-language models to compose objects localized with attributes?

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Yuval Reif and Roy Schwartz. 2023. Fighting bias with bias: Promoting model robustness by amplifying dataset biases. *arXiv preprint arXiv:2305.18917*.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695.

Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. 2017. Right for the right reasons: Training differentiable models by constraining their explanations. *arXiv preprint arXiv:1703.03717*.

Kuniaki Saito, Kihyuk Sohn, Xiang Zhang, Chun-Liang Li, Chen-Yu Lee, Kate Saenko, and Tomas Pfister. 2023. Pic2word: Mapping pictures to words for zero-shot composed image retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19305–19314.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Mohammadreza Salehi, Mehrdad Farajtabar, Maxwell Horton, Fartash Faghri, Hadi Pouransari, Raviteja Vemulapalli, Oncel Tuzel, Ali Farhadi, Mohammad Rastegari, and Sachin Mehta. 2024. Clip meets model zoo experts: Pseudo-supervision for visual enhancement. In *Transactions on Machine Learning Research (TMLR)*.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. 2022a. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade W Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa R Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. 2022b. LAION-5b: An open large-scale dataset for training next generation image-text models. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface. In *Advances in Neural Information Processing Systems*.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023a. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR.

Weijia Shi, Sewon Min, Maria Lomeli, Chunting Zhou, Margaret Li, Victoria Lin, Noah A Smith, Luke Zettlemoyer, Scott Yih, and Mike Lewis. 2023b. In-context pretraining: Language modeling beyond document boundaries. *arXiv preprint arXiv:2310.10638*.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2023c. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. 2022. Flava: A foundational language and vision alignment model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15638–15650.

Harman Singh, Pengchuan Zhang, Qifan Wang, Mengjiao Wang, Wenhan Xiong, Jingfei Du, and Yu Chen.

2023. Coarse-to-fine contrastive learning in image-text-graph space for improved vision-language compositionality.

Ryan Smith, Jason A Fries, Braden Hancock, and Stephen H Bach. 2022a. Language models in the loop: Incorporating prompting into weak supervision. *arXiv preprint arXiv:2205.02318*.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022b. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Suraj Srinivas and François Fleuret. 2018. Knowledge transfer with jacobian matching. In *International Conference on Machine Learning*, pages 4723–4731. PMLR.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, and Tao Yu. 2022. One embedder, any task: Instruction-finetuned text embeddings. *arXiv preprint arXiv:2212.09741*.

Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. 2023a. Eva-clip: Improved training techniques for clip at scale. *arXiv preprint arXiv:2303.15389*.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023b. Is ChatGPT good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937, Singapore. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

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*.

Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. 2022. Winoground: Probing vision and language models for visio-linguistic compositionality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5238–5248.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. 2024a. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *arXiv preprint arXiv:2406.16860*.

Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. 2024b. Eyes wide shut? exploring the visual shortcomings of multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9568–9578.

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*.

Michael Tschannen, Manoj Kumar, Andreas Steiner, Xiaohua Zhai, Neil Houlsby, and Lucas Beyer. 2023. Image captioners are scalable vision learners too. *arXiv preprint arXiv:2306.07915*.

Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2022. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). *arXiv preprint arXiv:2206.10498*.

Ankit Vani, Bac Nguyen, Samuel Lavoie, Ranjay Krishna, and Aaron Courville. 2024. Sparo: Selective attention for robust and compositional transformer encodings for vision. *arXiv preprint arXiv:2404.15721*.

Shikhar Vashishth, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. 2019. Attention interpretability across nlp tasks. *arXiv preprint arXiv:1909.11218*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Sagar Vaze, Nicolas Carion, and Ishan Misra. 2023. Genecis: A benchmark for general conditional image similarity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6862–6872.

Raviteja Vemulapalli, Hadi Pouransari, Fartash Faghri, Sachin Mehta, Mehrdad Farajtabar, Mohammad Rastegari, and Oncel Tuzel. Knowledge transfer from vision foundation models for efficient training of small task-specific models. In *Forty-first International Conference on Machine Learning*.

Haoxiang Wang, Pavan Kumar Anasosalu Vasu, Fartash Faghri, Raviteja Vemulapalli, Mehrdad Farajtabar, Sachin Mehta, Mohammad Rastegari, Oncel Tuzel, and Hadi Pouransari. 2024a. Sam-clip: Merging vision foundation models towards semantic and spatial understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3635–3647.

Junke Wang, Dongdong Chen, Zuxuan Wu, Chong Luo, Luowei Zhou, Yucheng Zhao, Yujia Xie, Ce Liu, Yu-Gang Jiang, and Lu Yuan. 2022a. Omnivl: One foundation model for image-language and video-language tasks. *arXiv preprint arXiv:2209.07526*.

Peifeng Wang, Aaron Chan, Filip Ilievski, Muhao Chen, and Xiang Ren. 2022b. Pinto: Faithful language reasoning using prompt-generated rationales. *arXiv preprint arXiv:2211.01562*.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Want to reduce labeling cost? gpt-3 can help. *arXiv preprint arXiv:2108.13487*.

Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. 2022c. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. 2022d. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. 2023. How far can camels go? exploring the state of instruction tuning on open resources. *Advances in Neural Information Processing Systems*, 36:74764–74786.

Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, and Tomas Pfister. 2024b. Chain-of-table: Evolving tables in the reasoning chain for table understanding. In *International Conference on Learning Representations*.

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, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Peter West, Chandra Bhagavatula, Jack Hessel, Jena D Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2021. Symbolic knowledge distillation: from general language models to commonsense models. *arXiv preprint arXiv:2110.07178*.

Sarah Wiegreffe, Ana Marasović, and Noah A. Smith. 2021. Measuring association between labels and free-text rationales. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10266–10284, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hui Wu, Yupeng Gao, Xiaoxiao Guo, Ziad Al-Halah, Steven Rennie, Kristen Grauman, and Rogerio Feris. 2021. Fashion iq: A new dataset towards retrieving images by natural language feedback. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 11307–11317.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv*.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. 2023. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*.

Hu Xu, Saining Xie, Xiaoqing Ellen Tan, Po-Yao Huang, Russell Howes, Vasu Sharma, Shang-Wen Li, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. 2023a. Demystifying clip data. *arXiv preprint arXiv:2309.16671*.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. 2023b. Retrieval meets long context large language models. *arXiv preprint arXiv:2310.03025*.

Xiaohan Xu, Chongyang Tao, Tao Shen, Can Xu, Hongbo Xu, Guodong Long, and Jian-guang Lou. 2023c. Re-reading improves reasoning in language models. *arXiv preprint arXiv:2309.06275*.

Nikolaos-Antonios Ypsilantis, Kaifeng Chen, Bingyi Cao, Mário Lipovskỳ, Pelin Dogan-Schönberger, Grzegorz Makosa, Boris Bluntschli, Mojtaba Seyedhosseini, Ondřej Chum, and André Araujo. 2023. Towards universal image embeddings: A large-scale dataset and challenge for generic image representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11290–11301.

Mert Yuksekgonul, Federico Bianchi, Pratyusha Kalluri, Dan Jurafsky, and James Zou. 2023. When and why vision-language models behave like bags-of-words, and what to do about it? In *International Conference on Learning Representations*.

Omar Zaidan, Jason Eisner, and Christine Piatko. 2007. Using "annotator rationales" to improve machine learning for text categorization. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 260–267, Rochester, New York. Association for Computational Linguistics.

Eric Zelikman, Yuhuai Wu, and Noah D Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *arXiv preprint arXiv:2203.14465*.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. *arXiv preprint arXiv:1808.05326*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. 2022. Lit: Zero-shot transfer with locked-image text tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18123–18133.

Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. 2020. Side-tuning: a baseline for network adaptation via additive side networks. In *European Conference on Computer Vision*, pages 698–714. Springer.

Kai Zhang, Yi Luan, Hexiang Hu, Kenton Lee, Siyuan Qiao, Wenhu Chen, Yu Su, and Ming-Wei Chang. 2024. Magiclens: Self-supervised image retrieval with open-ended instructions. *arXiv preprint arXiv:2403.19651*.

Qingru Zhang, Chandan Singh, Liyuan Liu, Xiaodong Liu, Bin Yu, Jianfeng Gao, and Tuo Zhao. 2023. Tell your model where to attend: Post-hoc attention steering for llms. *arXiv preprint arXiv:2311.02262*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Ye Zhang, Iain Marshall, and Byron C. Wallace. 2016. Rationale-augmented convolutional neural networks for text classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 795–804, Austin, Texas. Association for Computational Linguistics.

Tiancheng Zhao, Tianqi Zhang, Mingwei Zhu, Haozhan Shen, Kyusong Lee, Xiaopeng Lu, and Jianwei Yin. 2022. Vl-checklist: Evaluating pre-trained vision-language models with objects, attributes and relations. *arXiv preprint arXiv:2207.00221*.

Lianmin Zheng, Zhuohan Li, Hao Zhang, Yonghao Zhuang, Zhifeng Chen, Yanping Huang, Yida Wang, Yuanzhong Xu, Danyang Zhuo, Joseph E Gonzalez, et al. 2022. Alpa: Automating inter-and intra-operator parallelism for distributed deep learning. *arXiv preprint arXiv:2201.12023*.

Chong Zhou, Chen Change Loy, and Bo Dai. 2022. Extract free dense labels from clip. In *European Conference on Computer Vision*, pages 696–712. Springer.

Chapter A

Appendix: Efficient Language Model Deployment

A.1 Experiment detail

A.1.1 Implementation

We perform our experiments on cloud A100×16 GPU instances. We train the T5 models with the following hyperparameters, using publicly available packages from `https://github.com/huggingface/transformers`:

- T5-Base (220M) and T5-Large (770M): We train the models with learning rate $= 5\times10^{-5}$, batch size $=$ 64, max input length $= 1024$, for a maximum of 10000 steps.
- T5-XXL (11B): We train the models with learning rate $= 5\times10^{-5}$, batch size $= 32$, max input length $=$ 1024, for a maximum of 4000 steps.

We report all the results over 4 random runs, and include the standard error in the presented plots.

A.1.2 Datasets

We provide more detailed descriptions on the datasets used in our experiments. We include the sources from which we obtain the datasets as well as their original sources released from the authors. We refer readers to

these sources for their license or terms for use and/or distribution. To the best of our knowledge, the datasets used do not contain information that names or uniquely identifies individual people or offensive content.

- e-SNLI: The dataset was originally released in [Camburu et al., 2018], and made publicly available at https://github.com/OanaMariaCamburu/e-SNLI. We obtain the dataset from https://huggingface.co/datasets/esnli.
- ANLI: The dataset was originally released in [Nie et al., 2020], and made publicly available at https://github.com/facebookresearch/anli. We obtain the dataset from https://huggingface.co/datasets/anli. We use the R1 split in our experiments.
- CQA: The dataset was originally released in [Talmor et al., 2019], and made publicly available at https://www.tau-nlp.sites.tau.ac.il/commonsenseqa. It was then augmented with human-labeled explanations by [Rajani et al., 2019], which is available at https://github.com/salesforce/cos-e. We obtain the dataset used in our experiments from https://huggingface.co/datasets/cos_e.
- SVAMP: The dataset was originally released in [Patel et al., 2021]. We obtain the dataset from https://github.com/arkilpatel/SVAMP.
- ASDiv: The dataset was originally released in [Miao et al., 2020]. We obtain the dataset from https://github.com/chaochun/nlu-asdiv-dataset.

For each dataset, we randomly subsample 10% of the original training set to serve as validation set when validation set is not originally provided. For CQA, we use the original validation set to serve as our test set since the ground-truth labels are not available for the original test set. We provide the dataset statistics in Table A.1.

Table A.1: Dataset statistics used in our experiments.

Dataset	Train	Validation	Test
e-SNLI	549,367	9,842	9,824
ANLI	16,946	1,000	1,000
CQA	8,766	975	1,221
SVAMP	720	80	200

Chapter B

Appendix: Effective Language Model Adaptation

B.1 Multi-doc QA datasets

We use NaturalQuestions [Kwiatkowski et al., 2019][1] (released in Apache-2.0 license) and SynthWiki [Peysakhovich and Lerer, 2023][2] to conduct the experiments. Both datasets contains question-answer pairs, a gold document contains the answer, and $K-1$ distractor documents, where $K = 10$ and 20.

The NaturalQuestions dataset is the subset with 2655 queries selected by Liu et al. [2023b][3] where the annotated long answer is a paragraph. The $k-1$ distractor passages are Wikipedia chunks retrieved by Contriever [Izacard et al., 2022a] that are most relevant to the query but do not contain any of the annotated answers in NaturalQuestions. The distractor documents are presented in the context in order of decreasing relevance.

The SynthWiki dataset [Peysakhovich and Lerer, 2023] is a synthetic multi-doc QA dataset with 990 entries. All the documents in SynthWiki are GPT-4 generated Wikipedia paragraphs for fictional people, thus it can minimize the knowledge contamination issue from pre-training and ensure the LLMs can only use information from the provided context. The distractor documents are randomly sampled and randomly

[1] https://github.com/google-research-datasets/natural-questions
[2] https://github.com/adamlerer/synthwiki
[3] https://github.com/nelson-liu/lost-in-the-middle

ordered in SynthWiki.

NaturalQuestions is collected from public English Wikipedia articles and SynthWiki is collected by GPT-4 automatic generation of English fake Wikipedia articles. These two dataset should not contain any information that names or uniquely identifies individual people or offensive content. We ensure that the use of these two datasets was consistent with their intended purpose for academic research and in accordance with their specified licensing agreements.

B.2 Implementation details

In our experiments, we utilize `tulu-2-7b` and `Vicuna-7b-v1.5-16k` as the base models. Both models consist of 32 decoder layers, each with 32 attention heads. In applying attention calibration method to intervene model attention, we apply only to the last 16 decoder layers (and all of their attention heads). We find that intervening early layers may lead to unstable generation. We leave finding the best set of attention heads to intervene as future directions [Zhang et al., 2023].

In the experiments, we find attention calibration to be robust to the temperature term t in Eq. 3.5. We set $t = 5\text{e}{-5}$ for all experiments.

B.3 Additional experiment results

Different model formulations. To approximate equation 3.1, in addition to linear models as shown in equation 3.2, we also investigate log-linear models, which is defined as

$$\log \text{Attn}(x^{\text{doc}}, k) = \text{rel}(x^{\text{doc}}) + \text{bias}(k) + \epsilon, \tag{B.1}$$

where ϵ is a noise. We compute rank correlation as described in Sec. 3.3. The result is shown in Table B.1. The log-linear model and linear are competitive to each other, which all result in rank correlation above 0.75.

Table B.1: Rank correlations of linear and log-linear models.

Model form of f	Rank correlation
Linear	0.76
Log-linear	0.75

Table B.2: Our proposed attention intervention by calibrated attention stably improves models' RAG performances compared to existing re-ordering based baselines.

			Gold position in 10 documents				Gold position in 20 documents			
Dataset	Model	Method	1st	5th	10th	Avg.	1st	10th	20th	Avg.
NaturalQuestion	Vicuna	Vanilla attention	74.35	54.83	52.01	60.39	71.93	47.34	50.65	56.64
		Calibrated attention	70.84	62.61	55.78	63.07	66.40	56.19	51.75	58.11
		Attention sorting	72.54	59.54	63.12	65.06	69.37	56.91	62.41	62.89
		Prompt reordering	-	-	-	64.63	-	-	-	58.68
		LongLLMLingua-r_k	-	-	-	63.95	-	-	-	59.92
		LongLLMLingua-r_k + Cal.	-	-	-	66.17	-	-	-	62.22
	Tulu	Vanilla attention	70.50	48.81	49.26	56.19	56.94	35.32	46.59	46.28
		Calibrated attention	71.52	57.13	63.54	64.06	57.17	43.08	61.5	53.91
		Attention sorting	62.52	56.43	63.2	60.71	45.57	43.12	45.04	44.57
		Prompt reordering	-	-	-	58.77	-	-	-	44.64
		LongLLMLingua-r_k	-	-	-	56.39	-	-	-	43.90
		LongLLMLingua-r_k + Cal.	-	-	-	61.31	-	-	-	47.34
SynthWiki	Vicuna	Vanilla attention	65.15	48.68	68.58	60.80	53.73	43.63	60.20	52.52
		Calibrated attention	68.58	53.83	74.14	65.52	57.77	51.21	68.78	59.25
		Attention sorting	67.37	64.14	67.57	66.36	60.60	51.55	61.31	57.82
		Prompt reordering	-	-	-	70.20	-	-	-	62.22
		LongLLMLingua-r_k	-	-	-	70.50	-	-	-	62.42
		LongLLMLingua-r_k + Cal.	-	-	-	73.43	-	-	-	66.96
	Tulu	Vanilla attention	92.22	81.51	94.34	89.35	80.40	60.30	95.75	78.81
		Calibrated attention	92.92	87.77	95.25	91.98	82.22	75.15	96.14	84.50
		Attention sorting	92.92	92.82	93.83	93.19	94.04	93.53	95.05	94.20
		Prompt reordering	-	-	-	94.04	-	-	-	95.55
		LongLLMLingua-r_k	-	-	-	94.04	-	-	-	95.45
		LongLLMLingua-r_k + Cal.	-	-	-	94.44	-	-	-	95.75

Experiment tables. Table B.2 shows the exact numbers in our experiments.

B.4 Compute and inference details

In the experiments, we use the Huggingface Transformer package[4] with the two models: Tulu-2-7B[5] and Vicuna-7B-v1.5-16k[6] both contains 7B parameters. We run the experiments with two NVIDIA A100 GPUs. The inference time is roughly 1 to 3 hours on both datasets. We run our experiments with all greedy decoding without any non-deterministic factor, so we only need to run the experiments for once. Our method is a pure inference method, so there is no need to do training or hyperparameter searching.

[4] https://github.com/huggingface/transformers
[5] https://huggingface.co/allenai/tulu-2-7b
[6] https://huggingface.co/lmsys/vicuna-7b-v1.5-16k

Chapter C

Appendix: Data for Reliable Vision-Language Model Development

C.1 Implementation details

C.1.1 Hardware information

All experiments are run on a machine with an Intel(R) Xeon(R) CPU E5-2678 v3 with a 512G memory and two 48G NVIDIA RTX A6000 GPUs.

C.1.2 Dataset sources

We obtain all existing datasets from their original sources released by the authors. We refer readers to these sources for the dataset licenses. To the best of our knowledge, the data we use does not contain personally identifiable information or offensive content.

- CREPE [Ma et al., 2022]: We obtain CREPE dataset from its official repository [1].
- ARO [Yuksekgonul et al., 2023]: We obtain ARO dataset from its official repository [2].
- VL-CheckList [Zhao et al., 2022]: We obtain VL-CheckList dataset from its official repository [3].

[1] https://github.com/RAIVNLab/CREPE
[2] https://github.com/mertyg/vision-language-models-are-bows
[3] https://github.com/om-ai-lab/VL-CheckList

- COCO [Lin et al., 2014]: We obtain COCO from its official project website [4].

C.1.3 Software configuration

Models. We detail the sources of the pretrained models we use in the paper, and the hyper-parameters used in training our own models.

- Vera model [Liu et al., 2023a]: We obtain pretrained Vera model released by its author [5].
- Grammar model [Morris et al., 2020]: We obtain the Grammar model released by the authors [6].
- All pretrained CLIP models: We obtain all pretrained CLIP models' weights from OpenCLIP [7].
- NEGCLIP models: We obtain weights for pretrained NEGCLIP released by the authors [8]. For training from scratch and finetuning, we train RN50 and ViT-B/32 based on OpenCLIP codebase and set hyper-parameters as the following: number of warmup steps is 1000, batch size is 256, learning rate is 1e-4, weight decay is 0.1, number of epochs is 30. We augment the original CLIP loss with hard negative captions following NEGCLIP [Yuksekgonul et al., 2023].

Evaluations. We base our evaluation framework on OpenCLIP [Ilharco et al., 2021]. We follow all default hyper-parameters used for evaluating models.

C.2 Vision-language compositionality benchmarks

We provide an overview of existing vision-language compositionality benchmarks below, with Table C.1 summarizing the dataset comparisons.

C.2.1 Image-to-text formulation

A majority of current benchmarks formulate the evaluation task as image-to-text retrieval problem. These benchmarks generate hard negative texts procedurally through rule-based templates, where each benchmark

[4] https://cocodataset.org/
[5] https://huggingface.co/liujch1998/vera
[6] https://huggingface.co/textattack/distilbert-base-uncased-CoLA
[7] https://github.com/mlfoundations/open_clip
[8] https://github.com/mertyg/vision-language-models-are-bows

considers different types of hard negatives.

VL-Checklist [Zhao et al., 2022]. VL-CheckList aims at evaluating vision-language models' understanding of different objects, attributes, and relationships. It contains REPLACE hard negatives generated by replacing atomic parts of the positive texts with other foils. VL-CheckList further breaks the hard negatives down into more granular categories based on the type of the replaced atomic part, i.e. , object, attribute, or relationship.

ARO [Yuksekgonul et al., 2023]. ARO focuses on models' understanding of different relationships, attributes, and order information. It considers SWAP and SHUFFLE hard negatives. SWAP hard negatives are generated by swapping two words in the positive texts; on the other hand, SHUFFLE hard negatives are generated by shuffling words in the positive texts. ARO further divides SWAP hard negatives into attribute or relationship type.

CREPE [Ma et al., 2022]. CREPE is a large-scale evaluation benchmark that includes three types of hard negatives: REPLACE, SWAP and NEGATE. REPLACE and SWAP hard negatives are generated as in VL-CheckList and ARO. In addition, NEGATE hard negatives are generated by adding negation keywords (i.e. , *not* or *no*) to the original positive texts. The hard negatives are not further divided into fine-grained types (object, attribute, or relations).

C.2.2 Text-to-image formulation

Complementary to image-to-text formulation, compositionality can as well be evaluated by probing a model to select an image that best matches a given text description, against other hard negative images as distractors. Unlike hard negative texts, hard negative images are more difficult to obtain and thus current text-to-image compositionality benchmarks are smaller at scale.

Winoground [Thrush et al., 2022]. Winoground is a small dataset manually curated by human annotators. Each example in the dataset contains two images and two matching captions, where both captions contain identical words that appear in different orders. Note that Winoground can be used for either image-to-text or text-to-image retrieval. While the original intention for Winoground is to evaluate vision-language compositionality, recent work [Diwan et al., 2022] has pointed out that solving the tasks in Winoground requires not just compositional vision-language understanding, but additionally a suite of other abilities such

Table C.1: Summary on vision-language compositionality benchmarks. SUGARCREPE considers image-to-text formulation to enable larger scale evaluation set. In addition, SUGARCREPE considers a wide range of hard negative types. SHUFFLE and NEGATE are omitted as they introduce inevitable biases discussed in Sec. 4.4.2.

			Hard Negative Text Type				
Benchmark	**Task Formulation**	**Scale**	**SHUFFLE**	**REPLACE**	**SWAP**	**NEGATE**	**ADD**
VL-CheckList [Zhao et al., 2022]	Image-to-Text	> 1000		✓			
ARO [Yuksekgonul et al., 2023]	Image-to-Text	> 1000	✓		✓		
CREPE [Ma et al., 2022]	Image-to-Text	> 1000		✓	✓	✓	
Winoground [Thrush et al., 2022]	Image-to-Text / Text-to-Image	400			✓		
Cola [Ray et al., 2023]	Text-to-Image	210			N/A		
SUGARCREPE	Image-to-Text	> 1000		✓	✓		✓

as commonsense reasoning, or distinguishing visually difficult images.

Cola [Ray et al., 2023]. Cola tests a vision-language model's ability to select an image that correctly matches a given caption, against another distractor image with the same objects and attributes but in the wrong composition. The image pairs are mined from existing datasets. As a result, the final evaluation set is relatively small in size (210 examples in total).

We deem text-to-image evaluation as important as image-to-text evaluation. Future work can explore approaches to generate or mine compositional hard negative images at scale, as preliminarily explored in [Ray et al., 2023; Yuksekgonul et al., 2023].

C.3 SUGARCREPE

C.3.1 Taxonomy

Figure C.1 shows the taxonomy of SUGARCREPE. We first categorize the hard negatives based on their forms: REPLACE, SWAP, and ADD. We then further divide each type of hard negatives into finer-grained sub-categories based on the type (object, attribute, or relation) of the atomic concept altered. SUGARCREPE covers a total of 7 fine-graind hard negative types.

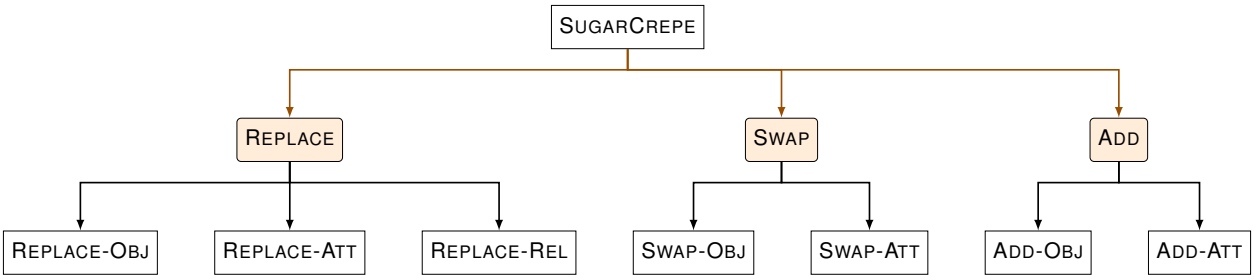


Figure C.1: Taxonomy of hard negatives considered in SUGARCREPE.

C.3.2 Hard negative generation procedure and templates

To generate hard negatives in SUGARCREPE, we come up with three different prompt templates for the three hard negative types considered: REPLACE, SWAP, and ADD. Each template consists of task instruction for generating the corresponding type of hard negatives and several (7 or more) few-shot demonstrations. We describe the general generation procedure and example prompt templates below and refer readers to our dataset repository for the full prompts used [9] .

Generating REPLACE hard negatives. To best leverage ChatGPT's capabilities, we devise a three-step workflow to generate REPLACE hard negatives: (1) We prompt ChatGPT in locating the desired atomic concepts (e.g. , objects) in the sentence; (2) We prompt ChatGPT to generate a new concept to replace a randomly selected old concept; (3) We let ChatGPT compose a new sentence by replacing the old concept with the new one. For steps (1) and (3), we prompt ChatGPT with a temperature of 0.0 to get stable outputs. For step (2), however, we diversify the outputs by prompting ChatGPT with a higher temperature of 1.5. With this design, we are able to generate diverse REPLACE hard negatives. Figure C.2 shows the example templates and outputs for REPLACE hard negatives.

Generating SWAP hard negatives. To generate SWAP hard negatives, which do not require any new concepts, we simply prompt ChatGPT once with a temperature of 0.0. Unlike REPLACE, SWAP hard negatives are only possible when there are at least two atomic concepts of the same category, i.e. , either object or attribute. Thus, our prompt first queries ChatGPT whether it is possible to swap two atomic concepts in the input sentence to generate a new description. Only if the answer is yes, will ChatGPT then proceed to identify two swappable concepts and compose the corresponding new sentence by swapping the two concepts. Figure C.3 shows the example templates and outputs for SWAP hard negatives.

[9] https://github.com/RAIVNLab/sugar-crepe

Generating ADD hard negatives. Similar to the REPLACE, we also employ a three-step prompting procedure to generate ADD hard negatives. The only difference in the procedure is that we prompt ChatGPT to add the generated new concept to the original caption, instead of using it to replace an old concept. Figure C.4 shows the example templates and outputs for ADD hard negatives.

C.3.3 Adversarial refinement

We detail the adversarial refinement procedure below. Given a text model M, we denote its output score for the positive and negative caption of i-th image as $M(p_i)$ and $M(n_i)$. If $M(p_i) > M(n_i)$, then the model could identify the correct caption for the i-th image without referring to it. For a test set to be unattackable given the text model M, the expectation of M's identifying the correct caption should be as close to random guess as possible; in particular, we hope that $E_i[M(p_i) > M(n_i)] = 0.5$. To achieve this for both the grammar model M_1 and plausibility model M_2, we first calculate the score difference $g_i^{(1)} = M_1(p_i) - M_1(n_i)$ and $g_i^{(2)} = M_2(p_i) - M_2(n_i)$, where the range of both $g^{(1)}$ and $g^{(2)}$ is $[-1, 1]$. Then we split the 2D space of the joint range of $g^{(1)}$ and $g^{(2)}$ into 100×100 equal grids, and for each pair of symmetric grids, e.g. , $\{(g^{(1)}, g^{(2)})|g^{(1)} \in (0.02, 0.04], g^{(2)} \in (-0.04, 0.06]\}$ and $\{(g^{(1)}, g^{(2)})|g^{(1)} \in (-0.02, -0.04], g^{(2)} \in (0.04, -0.06]\}$, we preserve the same number of data for both grids, therefore we ensure that for the resultant set, $E_i[M_1(p_i) > M_1(n_i)] = 0.5$ and $E_i[M_2(p_i) > M_2(n_i)] = 0.5$.

C.3.4 Dataset construction cost

We provide a high-level overview to the cost used to build SUGARCREPE by utilizing OpenAI's ChatGPT API for generating hard negatives. In building SUGARCREPE, we use approximately 40 API calls to generate hard negatives for each COCO test caption, including all different fine-grained types of hard negatives. This amounts to a total of $25,000 \times 40 = 1,000,000$ API calls to ChatGPT. With each API call costing around $\$0.0005$, it took roughly $\$500$ to build SUGARCREPE.

C.3.5 Dataset information

We host SUGARCREPE on Github [10]. The data card [Pushkarna et al., 2022] for SUGARCREPE, containing detailed dataset documentation, is available at the dataset repository [11]. We provide a summary below.

Dataset documentation. SUGARCREPE is a benchmark for faithful vision-language compositionality evaluation. Given an image, a model is required to select the positive text that correctly describes the image, against another hard negative text distractor that differs from the positive text only by small compositional changes. Each example consists of three fields:

- `filename`: The id to an image
- `caption`: Positive text correctly describing the image
- `negative_caption`: Hard negative text incorrectly describing the image

Maintenance plan. We are committed to maintain the dataset to address any technical issues. We actively monitor issues in the repository.

Licensing. We license our work using MIT License [12]. All the source data we use is publicly released by prior work [Lin et al., 2014].

C.4 Detailed evaluation results

C.4.1 Full evaluation results on existing benchmarks

We provide the full evaluation results over 17 pretrained CLIP models as well as 2 text-only models, Vera [Liu et al., 2023a] and the Grammar model [Morris et al., 2020], on existing compositionality benchmarks in Table C.2. We see that the text-only models, arguably without any vision-language compositionality, outperform most of the pretrained CLIP models, achieving state-of-the-art performances on many benchmark tasks. This implies that current benchmarks fail to faithfully reflect a model's vision-language compositionality.

[10] https://github.com/RAIVNLab/sugar-crepe
[11] https://github.com/RAIVNLab/sugar-crepe/blob/main/data_card.pdf
[12] https://github.com/RAIVNLab/sugar-crepe/blob/main/LICENSE

Table C.2: Blind models (i.e. , Vera and Grammar model) outperform all 17 existing pretrained CLIP models on nearly all existing benchmark tasks. This implies that current benchmarks fail to faithfully measure a model's vision-language compositionality.

		CREPE			ARO				VL-Checklist		
Source	**Model**	**Atomic**	**Swap**	**Negate**	**VG-Relation**	**VG-Attribution**	**COCO-Order**	**Flickr30K-Order**	**Object**	**Attribute**	**Relation**
Text-only model	Vera [Liu et al., 2023a]	43.70	70.80	66.15	61.71	82.59	59.81	63.52	82.48	73.99	85.72
	Grammar [Morris et al., 2020]	18.15	50.88	9.77	59.55	58.38	74.33	76.26	57.95	52.35	68.50
OpenAI [Radford et al., 2021a]	RN50	26.47	28.32	31.25	53.87	63.37	44.89	52.46	86.85	68.30	75.95
	RN101	27.63	32.74	12.50	52.43	62.93	29.86	39.34	86.44	67.93	71.75
	RN50x4	26.24	28.32	9.51	51.59	62.27	29.39	34.56	87.23	68.74	73.81
	ViT-B-32	22.31	26.55	28.78	51.12	61.33	37.14	47.18	87.00	68.80	77.04
	RN50x16	26.36	29.65	9.38	52.13	62.71	29.95	34.26	86.95	69.34	76.83
	RN50x64	26.82	30.09	23.57	51.00	62.56	40.54	46.74	87.71	68.61	74.97
	ViT-L-14	26.36	25.66	24.74	53.34	61.50	36.11	45.08	87.86	68.27	75.89
LAION [Schuhmann et al., 2022b]	ViT-H-14	23.70	25.22	16.54	50.33	62.93	25.79	30.96	85.39	68.46	71.13
	ViT-g-14	23.70	24.78	20.70	51.60	61.20	25.59	30.10	86.07	69.43	71.03
	ViT-bigG-14	23.58	24.78	17.97	51.61	61.89	25.24	30.22	84.66	67.80	66.48
	roberta-ViT-B-32	22.66	21.24	20.31	47.46	62.00	24.77	30.76	85.71	68.82	65.90
	xlm-roberta-base-ViT-B-32	21.16	20.80	12.76	47.93	59.73	23.85	30.32	86.06	70.41	63.01
	xlm-roberta-large-ViT-H-14	24.16	23.89	20.05	46.14	57.84	26.05	31.00	87.89	70.25	63.89
DataComp [Gadre et al., 2023]	`small`: ViT-B-32	13.64	27.88	14.84	50.83	50.17	13.35	14.02	68.72	58.80	57.00
	`medium`: ViT-B-32	16.42	20.35	11.33	50.45	54.04	16.44	16.26	78.43	63.53	62.94
	`large`: ViT-B-16	18.15	17.26	17.06	48.82	53.21	21.49	26.44	84.73	65.72	64.81
	`x-large`: ViT-L-14	21.62	22.57	16.28	48.54	60.03	23.19	29.52	86.66	67.01	67.93

C.4.2 SUGARCREPE human evaluation

To compare the quality of the hard negatives generated in SUGARCREPE to those in current benchmarks (i.e. , ARO+CREPE), we randomly sample 100 examples for each of the hard negative types: REPLACE, SWAP, and NEGATE / ADD. Each example is organized to consist of (1) the original positive text, (2) its hard negative in ARO+CREPE, and (3) its hard negative in SUGARCREPE. For each example, a human user rates whether the hard negative in ARO+CREPE or that in SUGARCREPE is better (or tie) in terms of commonsense and grammatical correctness, respectively. Note that we compare NEGATE in ARO+CREPE to ADD in SUGARCREPE, as both hard negatives are intended to probe a model's understanding of the *existence or not* of an atomic concept. Table C.3 shows that hard negatives in SUGARCREPE are much more sensical and fluent than that in ARO+CREPE across all three different types. For instance, SUGARCREPE has 68% more sensical and 46% more fluent hard negatives than ARO+CREPE on SWAP.

C.4.3 Additional NEGCLIP results

In this section, we conduct preliminary experiments to answer whether models' performances on SUGARCREPE would increase hugely if the models are trained with hard negatives generated through the same procedure as how we create SUGARCREPE. Since generating hard negatives for training with ChatGPT would incur substantial cost, we create hard negatives using a proxy method. In particular, we start with

Table C.3: Human evaluation results on the comparisons between hard negatives in ARO+CREPE and SUGARCREPE. We report the counts (out of 100 sampled examples) that the human user considers better or tie, w.r.t. both commonsense and grammatical correctness.

		Human counts of better examples		
Hard-negative Type	Evaluation	ARO+CREPE	SUGARCREPE	Tie
REPLACE	Commonsense	11	29	60
	Grammar	4	33	63
SWAP	Commonsense	4	68	28
	Grammar	4	46	50
NEGATE / ADD	Commonsense	1	26	73
	Grammar	1	35	64

template-generated hard negatives on the COCO training set and apply our adversarial refinement technique to remove the biases. We use this adversarially refined dataset for NEGCLIP training. We show the results in Table C.4. While we observe that the method improves over vanilla CLIP training without hard negatives, it performs similarly to NegCLIP and does not saturate the performance on SugarCrepe. This suggests that while the adversarial refinement mechanism prevents SugarCrepe from being attacked as an evaluation benchmark, leveraging the approach alone for training does not saturate the performance on SugarCrepe. Future work may characterize how LLMs could be used to generate better hard negatives for training to genuinely improve vision-language models' compositionality.

Table C.4: Model performances on SUGARCREPE when trained with hard negatives generated through similar procedure as how SUGARCREPE is created.

		SUGARCREPE		
Model	Hard negative	REPLACE	SWAP	ADD
CLIP without hard negatives	N/A	69.54	60.33	67.63
NEGCLIP with template hard negatives	REPLACE	74.32	62.65	72.92
NEGCLIP with adversarial refined hard negatives	REPLACE	73.37	61.40	72.84
NEGCLIP with template hard negatives	SWAP	73.31	68.35	71.93
NEGCLIP with adversarial refined hard negatives	SWAP	72.07	65.13	69.68
NEGCLIP with template hard negatives	NEGATE	72.74	60.89	70.47
NEGCLIP with adversarial refined hard negatives	NEGATE	72.70	60.75	68.70

```
Given an input sentence describing a scene, your task
is to:
1. Locate the noun words in the sentence.
2. Randomly pick one noun word.
3. Replace the selected noun word with a new noun word
to make a new sentence.

The new sentence must meet the following three
requirements:
1. The new sentence must be describing a scene that is
as different as possible from the original scene.
2. The new sentence must be fluent and grammatically
correct.
3. The new sentence must make logical sense.

Here are some examples:

Original sentence: A man is in a kitchen making pizzas.
Nouns: ["man", "kitchen", "pizzas"]
Selected noun: man
New noun: woman
New sentence: A woman is in a kitchen making pizzas.

Original sentence: a woman seated on wall and birds
besides her
Nouns: ['woman', 'wall', 'birds']
Selected noun: wall
New noun: bench
New sentence: A woman seated on a bench and birds
besides her.
```

(a) REPLACE-OBJ.

```
Given an input sentence describing a scene, your task
is to:
1. Locate the adjective words describing objects in the
sentence. If there are no adjective words, return an
empty list.
2. Randomly pick one adjective word.
3. Replace the selected adjective word with a new
adjective word to make a new sentence.

The new sentence must meet the following three
requirements:
1. The new sentence must be describing a scene that is
as different as possible from the original scene.
2. The new sentence must be fluent and grammatically
correct.
3. The new sentence must make logical sense.

Here are some examples:

Original sentence: a blue bike parked on a side walk.
Adjectives: ["blue"]
Selected adjective: blue
New adjective: red
New sentence: a red bike parked on a side walk.

Original sentence: The kitchen is clean and ready for
us to see.
Adjectives: ["clean", "ready"]
Selected adjective: clean
New adjective: dirty
New sentence: The kitchen is dirty and ready for us to
see.
```

(b) REPLACE-ATT.

```
Given an input sentence describing a scene, your task
is to:
1. Find any action or spatial relationships between two
objects in the sentence. If there are no such
relationships, return an empty list.
2. Randomly pick one relationship.
3. Replace the selected relationship with a new
relationship to make a new sentence.

The new sentence must meet the following three
requirements:
1. The new sentence must be describing a scene that is
as different as possible from the original scene.
2. The new sentence must be fluent and grammatically
correct.
3. The new sentence must make logical sense.

Here are some examples:

Original sentence: The dining table near the kitchen
has a bowl of fruit on it.
Relationships: ["near", "on"]
Selected relationship: near
New relationship: far from
New sentence: The dining table far from the kitchen has
a bowl of fruit on it.

Original sentence: A couple of buckets in a white room.
Relationships: ['in']
Selected relationship: in
New relationship: outside
New sentence: A couple of buckets outside a white room.
```

(c) REPLACE-REL.

Figure C.2: Example prompt templates (black) and outputs (green) from ChatGPT for REPLACE hard negatives.

```
Given an input sentence describing a scene, your task
is to first locate two swappable noun phrases in the
sentence, and then swap them to make a new sentence.
The new sentence must meet the following three
requirements:
1. The new sentence must be describing a different
scene from the input sentence.
2. The new sentence must be fluent and grammatically
correct.
3. The new sentence must make logical sense.

To complete the task, you should:
1. Answer the question of whether generating such a new
sentence is possible using Yes or No.
2. Output the swappable noun phrases.
3. Swap them to make a new sentence.

Here are some examples:

Input: A cat resting on a laptop next to a person.
Is it possible to swap noun phrases in the input
sentence to generate a new sentence that is different
from the input sentence and makes logical sense? Yes
Swappable noun phrases: laptop, person
Output: A cat resting on a person next to a laptop.

Input: A plate of donuts with a person in the
background.
Is it possible to swap noun phrases in the input
sentence to generate a new sentence that is different
from the input sentence and makes logical sense? Yes
Swappable noun phrases: a plate of donuts, a person
Output: A person with a plate of donuts in the
background.
```

(a) SWAP-OBJ.

```
Given an input sentence describing a scene, your task
is to first locate two swappable adjectives in the
sentence describing different objects, and then swap
them to make a new sentence.
The new sentence must meet the following three
requirements:
1. The new sentence must be describing a different
scene from the input sentence.
2. The new sentence must be fluent and grammatically
correct.
3. The new sentence must make logical sense.

To complete the task, you should:
1. Answer the question of whether generating such a new
sentence is possible using Yes or No.
2. Output the swappable adjectives.
3. Swap them to make a new sentence.

Here are some examples:

Input: A girl in a pink shirt holding a blue umbrella.
Is it possible to swap attributes in the input sentence
to generate a new sentence that is different from the
input sentence and makes logical sense? Yes
Swappable attributes: pink, blue
Output: A girl in a blue shirt holding a pink umbrella.

Input: A girl with a green shirt brushing her teeth
with a blue toothbrush.
Is it possible to swap attributes in the input sentence
to generate a new sentence that is different from the
input sentence and makes logical sense? Yes
Swappable attributes: green, blue
Output: A girl with a blue shirt brushing her teeth
with a green toothbrush.
```

(b) SWAP-ATT.

Figure C.3: Example prompt templates (black) and outputs (green) from ChatGPT for SWAP hard negatives.

```
Given an input sentence describing a scene, your task
is:
1. Find the objects in the sentence.
2. Randomly pick one object.
3. Generate a new object that's not in the sentence.
4. Add the new object next to the selected object to
make a new sentence.

The new sentence must meet the following three
requirements:
1. The new sentence must describe a clearly new and
different scene.
2. The new sentence must be fluent and grammatically
correct.
3. The new sentence must make logical sense.

Here are some examples:

Original sentence: An elephant standing under the shade
of a tree.
Objects: ["elephant", "shade of a tree"]
Selected object: elephant
New object: squirrel
New sentence: An elephant and a squirrel standing under
the shade of a tree.

Original sentence: A bench at the beach next to the sea
Objects: ['bench', 'beach', 'sea']
Selected object: bench
New object: umbrella
New sentence: An umbrella and a bench at the beach next
to the sea.
```

(a) ADD-OBJ.

```
Given an input sentence describing a scene, your task
is:
1. Find the objects in the sentence.
2. Randomly pick one object.
3. Generate a new plausible but uncommon attribute for
this object that's not in the sentence.
4. Add the new attribute next to the selected object to
make a new sentence.

The new sentence must meet the following three
requirements:
1. The new sentence must describe a clearly new and
different scene.
2. The new sentence must be fluent and grammatically
correct.
3. The new sentence must make logical sense.

Here are some examples:

Original sentence: A large white airplane and a person
on a lot.
Objects: ["airplane", "person"]
Selected object: airplane
New attribute: blue
New sentence: A large white and blue airplane and a
person on a lot.

Original sentence: three people riding horses on a
beach
Objects: ['three people', 'horses', 'beach']
Selected object: three people
New attribute: elderly
New sentence: Three elderly people riding horses on a
beach.
```

(b) ADD-ATT.

Figure C.4: Example prompt templates (black) and outputs (green) from ChatGPT for ADD hard negatives.

Chapter D

Appendix: Data for Improved Vision-Language Model

D.1 Datasets

CelebA-Attribute. There are a total of 40 different binary attributes in CelebA dataset [Liu et al., 2015], from which we select 29 attributes we consider objective, including: “Arched Eyebrows”, “Bags Under Eyes”, “Bald”, “Bangs”, “Black Hair”, “Blond Hair”, “Brown Hair”, “Gray Hair”, “Blurry”, “Bushy Eyebrows”, “Double Chin”, “Eyeglasses”, “Goatee”, “Male”, “Mouth Slightly Open”, “Mustache”, “No Beard”, “Oval Face”, “Pale Skin”, “Rosy Cheeks”, “Sideburns”, “Smiling”, “Straight Hair”, “Wavy Hair”, “Wearing Earrings”, “Wearing Hat”, “Wearing Lipstick”, “Wearing Necklace”, “Wearing Necktie”.

ImageNet-Subset. The ImageNet dataset [Deng et al., 2009] is organized according to the nouns in the WordNet hierarchy [Miller, 1995] and consists of 1000 classes. To evaluate the performance of conditioned representations, we form multiple subsets of ImageNet using the intermediate nodes from the WordNet hierarchy. We list all the ImageNet subsets we created in Table D.1.

Table D.1: ImageNet-Subset datasets and number of classes per each.

Node Name	Dog	Bird	Musical Instrument	Snake	Fish	Monkey	Ball	Car	Edible Fruit	Beetle	Cat	Spider	Bag	Piano
Num classes	118	59	28	17	16	13	10	10	10	8	7	6	5	2

D.2 Baselines

CLIP. We consider CLIP as a task-agnostic vision encoder baseline. In all experiments, we use OpenAI's CLIP-ViT-L-patch14-336 released checkpoint [Radford et al., 2021b]. The model size is 428M including both vision and text encoder. We consider the same model checkpoint in FocalLens-MLLM and FocalLens-CLIP.

InstructBLIP. InstructBLIP [Dai et al., 2023] is a MLLM that connects a frozen vision encoder, CLIP [Fang et al., 2023], to a large language model (LLM) decoder to enable multi-modal capabilities. Specifically, it adopts an instruction-aware Q-former architecture [Li et al., 2023d] as the connector. The Q-former takes in as input the image embedding extracted from the underlying vision encoder, along with tokenized text instructions. Through cross-attention design, the Q-former outputs multiple instruction-aware image tokens to be fed into the LLM decoder. In our experiments, we average over all image tokens to obtain the image representation used in our evaluations. We use the same instructions as in FocalLens for conditioning InstructBLIP.

MagicLens. MagicLens [Zhang et al., 2024] is a model trained specifically for composed image retrieval with a web-scale 36M-sized dataset. The model takes in both a reference image and natural language text to produce image representations that composes the semantics from both the input image and text. In our experiments, we condition MagicLens model using the same text instructions used for FocalLens.

D.3 Experiment details

Computation resource. We train FocalLens models on single node machines with 8 A100 GPUs.

Hyperparameters. For contrastive training with FocalLens, we report the hyperparameters used in Table D.2.

Table D.2: Training hyperparameters.

Model	Batch size	Epoch	Learning rate	Weight decay	Warmup ratio
FocalLens-MLLM	384	2	2e-5	0.	0.03
FocalLens-CLIP	2048	20	2e-5	0	0.03

D.4 Instructions used for different tasks

Here, we detail the instructions we use for different tasks for conditioning FocalLens and other instruction-aware baselines.

Table D.3: Instructions and templates used for different datasets and conditions.

Dataset	Condition	Instruction
ColorShape	Color	What is the color of the object in the image?
	Shape	What is the shape of the object in the image?
	Both	What is the color and shape of the object in the image?
CelebA-Attribute	Noun attributes (e.g., Arched Eyebrows)	Does the person in the image have {attribute}?
	Adjective attributes (e.g., Bald)	Is the person in the image {attribute}?
CelebA-Identity	-	Gender, age, eye color, hair color, face shape, facial hair of the person.
GeneCIS	Focus attribute	Focus on the {attribute}.
	Focus object	Is there {object}?
ImageNet-Subset	category (e.g., dog)	What type of {category} is in the image?
Fine-grained datasets	category (e.g., flower)	What type of {category} is in the image?
SugarCrepe	Replace-obj	Focus on the presence of objects in the image.
	Replace-att	Focus on the color, patterns and other attributes of the objects in the image.
	Replace-rel	What are the relationships between the objects in the image?
	Swap-obj	What are the actions, states, colors, patterns and relationships of the objects in the image?
	Swap-att	What kind of objects are in the image?
	Add-obj	What is not in the image?
	Add-att	What is not in the image?
MMVP-VLM	Orientation	Describe the orientation, position, or the direction of the object.
	Presence	Focus on the presence of objects in the image.
	State	Focus on the specific state or the condition of the objects in the image.
	Quantity	Focus on the quantity of the objects in the image.
	Spatial	Describe the spatial relationship and the positions of the objects in the image.
	Color	Focus on the color of the objects in the image.
	Structural	Describe the state of the objects in the image.
	Text	Focus on the texts on the objects in the image.
	Camera	Describe the perspective and view from which the photo is taken.

D.5 Full experiment results

D.5.1 CelebA-Attribute full results

We report full CelebA-Attribute results in Table D.4.

D.5.2 ImageNet-Subset full results

We report full ImageNet-Subset results in Table D.5.

Table D.4: Full results on CelebA-Attribute.

Model	Arched Eyebrows	Bags Under Eyes	Bald	Bangs	Black Hair	Blond Hair	Blurry	Brown Hair	Bushy Eyebrows	Double Chin
CLIP	8.13	12.00	24.52	2.86	7.96	6.20	5.52	-0.58	11.98	18.35
InstructBLIP	7.12	8.35	27.40	4.95	9.50	21.03	14.67	-0.81	3.73	11.01
MagicLens	11.32	12.10	15.14	2.44	7.48	8.24	8.95	-3.22	6.75	13.88
FocalLens-MLLM	15.15	14.98	19.23	4.38	17.95	25.76	6.14	4.44	6.88	15.37
FocalLens-CLIP	13.38	13.00	26.68	8.19	10.24	32.22	11.03	5.53	9.99	15.94
Model	Eyeglasses	Goatee	Gray Hair	Male	Mouth Slightly Open	Mustache	No Beard	Oval Face	Pale Skin	Rosy Cheeks
CLIP	17.84	20.16	24.19	54.55	4.72	20.92	27.64	1.63	3.22	-3.15
InstructBLIP	41.83	16.17	22.56	43.66	12.87	19.16	23.75	0.77	2.73	-3.45
MagicLens	15.52	11.28	20.13	64.56	6.04	13.50	27.52	1.83	1.98	1.95
FocalLens-MLLM	47.72	20.96	22.40	96.82	33.41	19.30	34.30	1.66	1.36	5.85
FocalLens-CLIP	24.90	29.04	23.86	95.04	10.82	26.59	41.80	0.94	4.58	-0.90
Model	Sideburns	Smiling	Straight Hair	Wavy Hair	Wearing Earrings	Wearing Hat	Wearing Lipstick	Wearing Necklace	Wearing Necktie	
CLIP	18.21	8.68	3.47	7.54	7.32	17.60	41.45	-0.67	21.81	
InstructBLIP	12.10	21.71	3.17	13.91	13.51	45.11	34.64	1.94	36.56	
MagicLens	11.54	9.98	2.84	10.76	10.92	21.19	54.12	3.58	16.97	
FocalLens-MLLM	20.02	34.43	4.50	17.61	21.54	34.32	68.07	5.05	37.86	
FocalLens-CLIP	32.35	22.11	2.81	16.89	12.39	33.58	62.50	3.07	29.80	

Table D.5: Full results on ImageNet-Subset.

Model	Bag	Ball	Beetle	Bird	Car	Cat	Dog
CLIP	55.61	64.63	51.84	66.72	57.73	53.00	16.55
InstructBLIP	60.13	66.44	51.10	45.86	60.54	51.22	9.60
MagicLens	53.22	68.10	43.37	51.69	54.15	50.14	17.28
FocalLens-MLLM	63.95	78.99	41.44	54.14	54.46	53.24	29.25
FocalLens-CLIP	59.44	70.01	46.88	64.62	61.84	56.80	33.15
Model	Fruit	Fish	Monkey	Music Instrument	Piano	Snake	Spider
CLIP	60.95	61.79	37.79	39.18	61.97	32.03	54.61
InstructBLIP	49.74	59.16	27.96	41.44	66.17	26.45	51.61
MagicLens	57.40	58.84	26.82	41.18	57.40	25.74	43.76
FocalLens-MLLM	65.98	57.40	34.81	57.83	57.14	29.47	54.69
FocalLens-CLIP	69.78	65.37	38.30	61.29	60.60	32.06	53.93