ON THE DIAGNOSIS AND GENERALIZATION OF COMPOSITIONAL VISUAL REASONING

by

Zhuowan Li

A dissertation submitted to Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy

Baltimore, Maryland

January, 2024

Abstract

Computer vision is not only about recognizing visual signals, but also reasoning over perceived visual elements. This ability, termed **visual reasoning**, is typically studied by multimodal tasks like visual question answering and image captioning. Thanks to recent developments in multimodal vision-and-language, we are closer to achieving visual reasoning than ever. However, more efforts are still required, in order to build visual reasoning systems that are robust, interpretable and generalizable.

In this dissertation, I present my efforts towards visual reasoning, through both model diagnosis and enhancements. In the first part, I diagnose existing visual question answering models, including the end-to-end models and compositional models, and show the advantage of the latter. In the second part, I dive deeper into compositional models, proposing techniques for enhancing them with improved performance on real-world images. In the third part, I generalize visual reasoning onto a different task, image captioning, introducing a new setting of the task that requires strong reasoning to summarize and compare groups of images. With this dissertation, I showcase the advantages and disadvantages of compositional visual reasoning methods, which should be pursued in conjunction with non-compositional end-to-end models.

Thesis Committee

Alan Yuille (Primary Advisor)
Bloomberg Distinguished Professor
Department of Cognitive Science and Computer Science
Johns Hopkins University

Benjamin Van Durme (Co-Advisor)
Associate Professor
Department of Cognitive Science and Computer Science
Johns Hopkins University

Daniel Khashabi
Assistant Professor
Department of Computer Science
Johns Hopkins University

Acknowledgments

First and foremost, I thank my advisors, Prof. Alan Yuille and Prof. Benjamin Van Durme. I feel incredibly fortunate to be co-advised by two great advisers. I deeply appreciate Alan for the introducing me to the computer vision field since I was an undergraduate intern student, for encouraging me later to explore interdisciplinary research direction, vision-and-language, which turns into this thesis, for educating us with the dedication and passion for fundamental research problems, and for the long afternoon chats about research, career and life. I sincerely appreciate Ben for always being flexible and supportive of what I work on, for the consistent weekly meetings through years that keep my PhD journey smooth, and for encouraging me to explore both open science topics and pragmatic applications. I deeply appreciate that both advisors put students' interest and well-being at first.

I also thank my thesis committee member, Daniel Khashabi, and my GBO committee members, Justin Halberda, Vishal Patel, Anqi Liu, Rene Vidal, David Etter, for the questions, discussions, and feedbacks about my works.

I am grateful for all my collaborators. I thank Elias Stengel-Eskin for the discussions over multiple projects and the consistently patient helps in English writing. I thank Chenxi Liu and Cihang Xie for the mentoring and support

through years. I thank Adam Kortylewski for showing me how to drive a project. I thank Kate Sanders, David Etter and Reno Kriz for the feedbacks about my weekly updates. I appreciate the collaborations with my labmates Yixiao Zhang, Yingwei Li and Wufei Ma. I also thank the student advisees that I have worked with, including Vipul Gupta, Chenyu Zhang, Xingrui Wang, Varun Iyer, Chenyu Heidi Zhang, Xingrui Wang, Yuxuan Wang, Shitian Zhao, Yijiang Li.

I am fortunate to have three great internships in industry at Adobe, Meta, and Amazon. I thank my mentors Quan Tran, Long Mai, and Zhe Lin at Adobe, Dhruv Mahajan, Vignesh Ramanathan, Deepti Ghadiyaram and Filip Radenovic at Meta, Bhavan Jasani, Peng Tang and Shabnam Ghadar at Amazon. The experience in industry has expanded my perspective beyond academic research, playing a pivotal role in shaping my career.

I am lucky to be a part of two wonderful labs, spending time with the warm and talented lab members. Among those from CCVL: Lingxi Xie, Cihang Zie, Yuyin Zhou, Adam Kortylewski, Zongwei Zhou, Yaoyao Liu, Yi Zhang, Jieru Mei, Zihao Xiao, Chenglin Yang, Yutong Bai, Angtian Wang, Chen Wei, Jieneng Chen, Ju He, Prakhar Kaushik, Qihao Liu, Xiaoding Yuan, Runtao Liu, Bowen Li, Wufei Ma, Junfei Xiao, Jiahao Wang, Yu-Cheng Chou, Yixiao Zhang, Qihang Yu, Hongru Zhu, Fengze Liu, Yingwei Li, Qi Chen, Chenxu Luo, Yongyi Lu, Yingda Xia, Huiyu Wang, Qing Liu, Siyuan Qiao, Yuhui Zu, Zefan Li, Zhuotun Zhu, Weichao Qiu, Wei Shen, Yan Wang, Zhishuai Zhang, Chenxi Liu. Among those from BLab: Elias Stengel-Eskin, Kate Sanders, Nils Holzenberger, Marc Marone, Yunmo Chen, Guanghui Qin, Nathaniel Weir,

Orion Weller, Zhengping Jiang, Tongfei Chen, Sheng Zhang.

Finally, my life has been shaped by my family and friends. I thank my parents for their unconditional love and support, and my boyfriend for his consistent companionship. I am grateful for all my friends, old and new, on either side of the Pacific Ocean, for the countless enjoyable times.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Computer vision is a research field involving a broad range of problems, which requires low-level recognition including image segmentation, classification, etc., and high-level intelligence that enables cognitive understanding of the perceived visual signals. In this dissertation, I study **visual reasoning**, a high-level intelligence that requires reasoning over the visual elements in an image, which I believe is crucial for building machines that can understand and think about the visual world like humans do.

To study visual reasoning, it is natural to leverage language, which contains

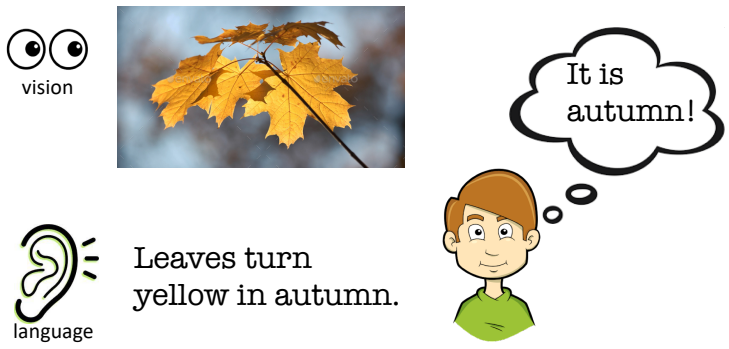


Figure 1.1: Language is a natural interface visual reasoning.

rich semantics and logic, as an interface. As an example, in Fig. 1.1, the knowledge "leaves turn yellow in autumn" enables the reasoning about seasons over the image showing the yellow leaves. Understanding both modalities, *i.e.*, vision and language, is crucial for reasoning about the visual world. In fact, multimodal understanding has been widely studied with various proxy tasks. Given a image, to describe it with natural language is called image captioning; to identify objects or regions specified by a language query is called referring expression understanding; to answer specific questions about it is called Visual Question Answering (VQA). In this dissertation, I focus on two tasks: visual question answering, which serves as a discriminative probe of the models' reasoning ability, and image captioning, which is a generative task that requires reasoning with free-form language.

There has been two lines of approaches for visual reasoning. One is the black-box neural networks trained end-to-end for solving the tasks directly, which performs well on standard benchmarks but is in short of interpretability and robustness. The other approach, the compositional method, favors modularity, interpretability, and data efficiency, but requires specifically-designed modules, thus is hard to generalize and achieve competitive performance on real-world data. In this dissertation, I first compare the two types of methods by systematically benchmarking and diagnosing their advantages and disadvantages. Subsequently, I dive deeper into the compositional reasoning methods, proposing techniques for enhancing their performance on real-world data.

Putting everything together, I study visual reasoning, represented in various tasks, from the perspective of diagnosis and model enhancement. I showcase the challenges in visual reasoning, and present the advantages and disadvantages of compositional and non-compositional methods, and suggest that compositional models, characterized by strong robustness and interpretability, be pursued in conjunction with non-compositional end-to-end models.

Next, I will introduce the outline of this thesis and illustrate how each part fits in the story.

1.1 Diagnosing Visual Question Answering

In Part I, I focus on the task of visual question answering, which is a discriminative task that queries the model to answer specific questions. I introduce methods for **model diagnosis**, enabling better understanding of existing models. Understanding the models is important for reliable applications, but this is challenging due to the black-box nature of neural networks. I do the diagnosis with two strategies: feature swapping and synthetic data. Using the two method, I aim to better understand, and systematically benchmark the robustness of current models, including how they perform on images with swapped objects, how they generalize to testing images that differs from training ones, how they answer various types of questions, like questions requiring 3D understanding of the images.

In Chapter 2, I propose to diagnose models by feature swapping, which

is a strategy to perturb the input features of VQA models. Using this strategy, I discover that current VQA models over-rely on visual context, *i.e.*, the irrelevant visual elements in the image, to answer the questions, thus are not robust to changes in the features of visual context. By swapping the features of the irrelevant visual elements, the model's predictions can be dramatically changed, leading to a significant performance drop. In additional to revealing the model's vulnerability in robustness, in this chapter, I also show that this feature swapping strategy can be applied in training time to improve the model robustness.

In Chapter 3, I introduce a synthetic benchmark, named Super-CLEVR, to benchmark the robustness of VQA models towards domain shifts. Domain shift, *i.e.*, the testing domain shifted from the training domain, is an important challenge for current models, but can hardly be studied due to the difficulty to control the distribution of real-world datasets. Alternatively, in this synthetic world, I control the distribution over a broad range of nuanced factors, including visual complexity, question redundancy, concept distribution, and concept compositionality. Different types of models, including the end-to-end models and the compositional models, are studied over these factors. The analysis shows that one compositional model, with probability injected, shows the best robustness.

While image is a 2D capture of the world, it is important that the world is fundamentally in 3D. Chapter 4 expands the Super-CLEVR benchmark with 3D-aware questions, which requires 3D understanding of the images

to answer, such as the 3D poses of objects, occlusion relationships, and part-whole object structures. Various models are evaluated on Super-CLEVR-3D to access their ability for answering questions about the 3D world from the 2D images. Moreover, this benchmark motivates a neural symbolic method, for 3D VQA, which will be described in Chapter 4 as well.

1.2 Visual Question Answering with Compositional Models

Motivated by the diagnosis findings in Part I showing merits of compositional methods, in Part II, I dive deeper into the compositional models for visual question answering. I introduce several techniques to improve existing compositional models for stronger performance on real-world data.

In Chapter 5, I generalize the synthetically successful compositional models onto real-world images. Existing compositional models performs great on toy synthetic world containing clean images with simple objects, but do not generalize to real-world images, which are far more complex and contains countless visual concepts, *e.g*. objects, attributes, and relationships. With analysis, I discover two limiting factors: the long-tail distribution of real-world visual concepts, and the unequal importance of reasoning steps in real questions. Two simple yet effective techniques, *i.e*., calibrating the concepts and operations, are proposed to address the two limiting factors respectively. With these techniques, the model performance is greatly enhanced on real-world benchmark.

Recently, empowered by large language models (LLMs), compositional

reasoning has been brought to a new stage by using LLMs as reasoning planners. However, the method still suffer from errors sourced from either LLM reasoning planner, or from the visual modules that the execute the reasoning program. In Chapter 6, I introduce a "plug-and-play" method to correct the reasoning results through introspective verification. Specifically, a mixture-of-expert of verification modules are devised to validate predictions after each reasoning step, subsequently calibrating the visual module predictions and refining the reasoning trace planned by LLMs, leading to improved performance over a wide range of visual reasoning tasks.

Visual representations provides the foundation for compositional reasoning. In Chapter 7, I conduct a detailed analysis of the visual representations in pretrained large models. I compare the visual features in pretrained vision models, and multi-modal (vision-and-language) models, by probing the features on a broad spectrum of visual tasks. The probing tasks include label prediction tasks like object name and attribute prediction, which requires semantic understanding of the visual elements, and dense prediction tasks like detection and segmentation, which requires localized understanding of the image. The probing results suggest that multimodal representations encode stronger semantics information, while vision-only representations are stronger in localization. The probing results serve as a preliminary study to guide the usage of foundation models for visual tasks.

1.3 Image Captioning with Contrastive Reasoning

Part III moves on to the generative image captioning task. I introduce a new setting of the image captioning task that requires reasoning. While traditional image captioning generates text descriptions for a single image, in Chapter 8, I introduce a much more challenging setting: describe a group of images, in the contrast of more context images. The challenges of new setting come from reasoning for comparison of the similarities and differences between images: the models is required to describe the similarity of a group of images, and contrast them with the context images to summarize the differences. This task can be potentially grounded to real-world applications, like suggesting personalized search queries for image search engines. I created benchmarks for this new setting based on datasets of traditional image captioning, and proposed models based on attention and feature contrasting to solve the task.

Part I

Diagnosing Visual Question Answering

Chapter 2

SwapMix: Diagnosing and Regularizing the Over-Reliance on Visual Context in Visual Question Answering

While VQA has progressed rapidly, previous works raise concerns about robustness of current VQA models. In this chapter, we study the robustness of VQA models from a novel perspective: visual context. We suggest that the models over-rely on the visual context, i.e., irrelevant objects in the image, to make predictions. To diagnose the models' reliance on visual context and measure their robustness, we propose a simple yet effective perturbation technique, SwapMix. SwapMix perturbs the visual context by swapping features of irrelevant context objects with features from other objects in the dataset. Using SwapMix we are able to change answers to more than 45% of the questions for a representative VQA model. Additionally, we train the models with perfect sight and find that the context over-reliance highly depends on the quality of visual representations. In addition to diagnosing, SwapMix can

also be applied as a data augmentation strategy during training in order to regularize the context over-reliance. By swapping the context object features, the model reliance on context can be suppressed effectively. Two representative VQA models are studied using SwapMix: a co-attention model MCAN and a large-scale pretrained model LXMERT. Our experiments on the popular GQA dataset show the effectiveness of SwapMix for both diagnosing model robustness, and regularizing the over-reliance on visual context. The code for our method is available at https://github.com/vipulgupta1011/swapmix

2.1 Introduction

In recent years, VQA performance is greatly boosted by different techniques including intra- and inter-modality attentions [1, 3], large scale multi-modal pretraining [2, 4, 5], etc. However, previous works study the robustness of VQA models and show that the models may exploit language prior [6, 7, 8], statistical bias [9, 10] or dataset shortcuts [11, 12] to answer questions.

While previous works studied VQA robustness from the perspective of language context, in this work, we study the robustness of VQA models from a different view: visual context. The visual context refers to the background in the image or the irrelevant objects that are not needed during the reasoning process to answer the question. For example, in Figure 2.1, the tennis ball is irrelevant for the question "What color is the women's dress", so we say it is a context object. Ideally, a model with real perception and reasoning ability should be robust to the irrelevant context. However, in our work, we find that VQA models are vulnerable to context changes, which suggests the models'

over-reliance on the irrelevant context in the image.

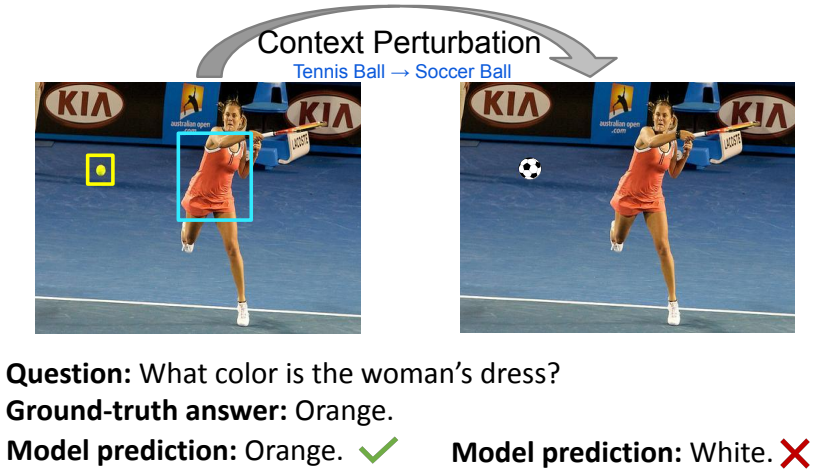


Figure 2.1: VQA models over-rely on visual context. By swapping features of irrelevant context objects, we can perturb the model prediction. Here the tennis ball (in yellow box) is an irrelevant context object for the question. Changing feature of the tennis ball to feature of soccer ball results in change in model prediction.

To study the role of visual context, we propose a simple perturbation strategy named *SwapMix*, which perturbs the visual context by swapping features of context object with features from another object in the dataset. We first identify the visual features corresponding to irrelevant objects in the image, then randomly swap them with feature vectors of another similar object from the dataset. For example, in Figure 2.1, the tennis ball is a context object for the given question, so we swap tennis ball feature vector with a feature vector of soccer ball. The swapping confuses the model to mis-recognize the color of the dress. In the swapping process, we carefully control the swapped objects to ensure that the new object is compatible to the scene (*e.g.*, we don't want to change the ball into a car).

Surprisingly, by perturbing the irrelevant context, more than 45% of the

correct answers get changed. This reveals that VQA models highly rely on the context in the image, thus are vulnerable to context perturbations. The model may utilize shortcut correlations in the visual context to make predictions. We diagnose two representative VQA models: MCAN [1] as representative for attention-based models, and LXMERT [2] as representative for large-scale pretrained models. Our experiments show that LXMERT is much more robust to context perturbations, which indicates that large-scale pretraining may increase model robustness.

We further find that the context over-reliance highly depends on the quality of visual representations: a perfect sighted model relies much less on context. We achieve this by replacing the visual representations [1] with the ground-truth object and attribute encoding, which can be viewed as gold visual representation that provides the model the perfect sight. By studying this perfectly sighted model, we can exclude the influence of imperfect visual perception, thus purely focus on the reliance on relevant objects in the reasoning process. Our results shows that by providing VQA models with the perfect visual encoding, the answer changes are greatly reduced from 45.0% to 16.4% (for MCAN model). This suggests that models trained with perfect visual representations are more robust and that the context over-reliance largely comes from the imperfection of visual perception features.

In addition to diagnosing context over-reliance, SwapMix can also be used as a data augmentation technique during training. In training, we randomly swap a part of the context features with other object features from the dataset.

[1] Majority of VQA models use object features extracted by pretrained object detectors as visual representation.

This forces the model to focus more on relevant objects in the image and less on irrelevant context. Our empirical results show that by applying *SwapMix* in training, the model robustness improves by more than 40% and effective accuracy improves by more than 5% on GQA dataset [13].

Our main contributions in this paper are three-fold. First, we are the first to study VQA robustness from the perspective of visual context. With our simple context perturbation strategy named SwapMix, we benchmark robustness of two representative VQA models and find their over-reliance on visual context. Second, we find that a perfect sighted model relies much less on visual context. We provide models with perfect visual encodings and observe the improvement in model robustness. Third, we define 2 metrics, context reliance and effective accuracy and shows improvement by using SwapMix as data augmentation technique.

2.2 Related Works

Visual Question Answering. The most common approach for VQA is to first extract visual features using convolution neural networks and question features using LSTM[14], then fuse them together to make answer predictions [15]. Multiple works have shown the effectiveness of attention in VQA [16, 17, 18, 19, 20, 21, 22]. BAN[23] proposes bilinear attention that utilizes vision and language information. MCAN [1] is a co-attention model which uses self-attention and guided-attention units to model the intra-modal and inter-modal interactions between visual and question input. OSCAR [5] uses object tags in images as anchors to improve alignment between modalities. LXMERT

[2] is a large-scale Transformer [24] model that consists of three encoders: an object relationship encoder, a language encoder, and a cross-modality encoder. In concurrent work, [25] proposes feature swapping for domain adaptation from synthetic to real data.

Biases and Robustness in VQA. Despite the prosperity in the development of VQA, multiple previous works show bias in VQA models. [9] points out the generalization incapacity of VQA models. [26] shows bias reliance of VQA models. [27] discover and enumerate explicitly biases learned by the model. Many work show that the models exploit language prior [6, 7, 8], statistical bias [9, 10] or dataset shortcuts [11, 12] to answer questions. There are many approaches to mitigate the bias in models. [6] introduces a method that reorganizes the VQA v2 dataset. Some works use question-only model: [28] introduces training as an adversarial game between the base model and a question-only adversary, while [29] adds a question-only branch to do joint training with the base model, and omits it at test time. CSS [30] generates counterfactual samples during training, which improves the visual-explainable ability. [31, 32] leverage the important visual information by humans to focus on selected regions during training. [33] designed a two-stage model, the first stage trains only on biases, and the second stage focuses on the other patterns of the dataset. In addition to decreasing modal biases, there are lot of work on measuring biases more accurately and efficiently. MUTANT [34] and GQA-OOD [11] use out-of-distribution (OOD) generalization. Early work like [35] provided a soft measure score based on a lexical set. [36] measures the performance of the models based on both the baseline questions and the

CLOSURE test, indicating that the gap between these two measurements is the behavior of generalization. [37] measures bias in VQA by finding counter-examples from validation set with their proposed rules and use the mined counter-examples to evaluate model. Different from the above previous works, our work is the first to study the reliance on visual context of VQA models by generating new examples.

Context in computer vision. Contextual information is important for computer vision. For object recognition, early work by [38] introduced a context-based model using place categorization to simplify object recognition, [39] studied how context influences object recognition, and recent work by [40] modified a global context model to enhance performance. Moreover, [41] demonstrate that object detection models rely too much on contextual information when objects are occluded, and resolve this using a compositional generative model [42, 43] that separates the object and context in the representation. For scene graphs, [44, 45] introduce a hierarchical context model to generate a scene graph, and [46] augment the node features of scene graphs with contextual information. For segmentation, [47] presents multi-scale contextual representation with context modules, which leverage the global image representations to estimate local affinity of sub-regions, and [48] introduces a switchable context network to improve the performance of semantic segmentation of RGB-D images. In the field of VQA, [49] add a visual context based attention that takes into account the previously attended visual content.

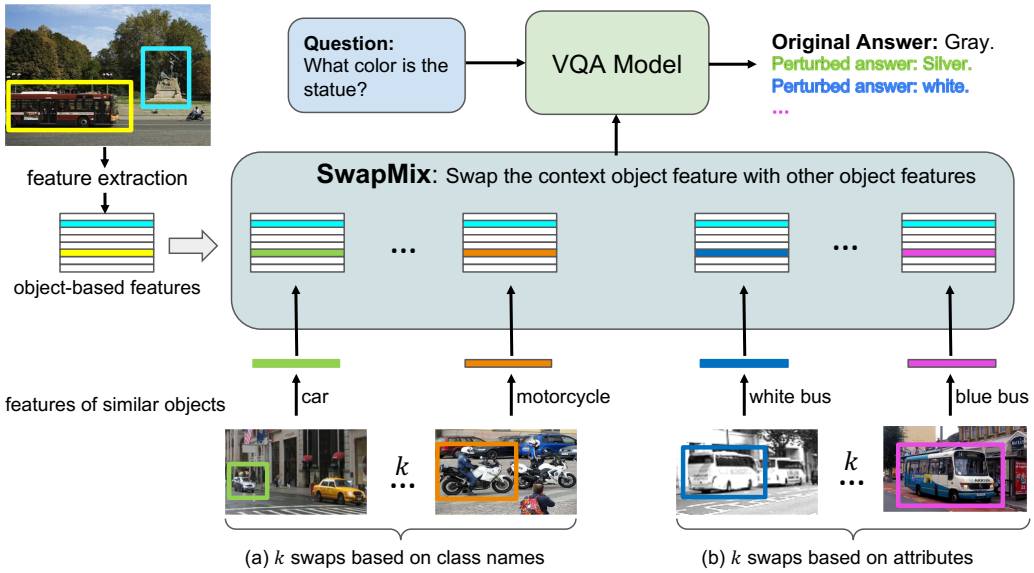


Figure 2.2: Overview of our method. Given an image and a question, we first find context object (*e.g.* red bus in the yellow box) using the reasoning steps of the question. Then we swap the context object feature with other similar object features in the dataset. We perform k swaps based on (a) object class names and (b) object attributes each. The model's reliance on context can be evaluated with the percentage of answer changes when context gets perturbed.

2.3 Method

VQA models are not robust to minor perturbations. In this section, we provide a simple perturbation technique that measures the reliance of VQA models on visual context, i.e. irrelevant objects image. We swap features corresponding to irrelevant objects in the image with other objects from the dataset. In an ideal scenario, changing the context objects in the image should not affect the model's prediction, while in our experiments, we found that VQA models rely heavily on the context and are not robust to small perturbations.

We name our method, SwapMix, which performs perturbations on visual context to diagnose the robustness of the model. SwapMix can also be used as a data augmentation technique during training to improve the robustness and effective accuracy of the model. We first define what visual context is, then introduce VQA models with perfect sight which leads to interesting diagnosing findings, next describe how we perform SwapMix, and finally talk about how to apply SwapMix as a training strategy.

2.3.1 Definition of Visual Context

Here we clarify the definition of visual context and provide formulation for the problem.

$$f = Model(V, Q)$$

Here, V represents the visual representation and Q represents the question input. A widely-used visual representation is the object-based features [50] extracted by pretrained object detector Faster RCNN [51]. In this case, $V \in \mathbb{R}^{n \times d}$ is a set of object features, where n is the number of objects in the image and d is the dimension of the feature vector for each object.

Among the n objects in the image, there are some irrelevant objects that are not needed in the reasoning process of question answering. For a fully robust model, changing the context, C should not change model's prediction as shown in Figure 2.1. We refer to those irrelevant objects as visual context and denote visual context by C. $C \in \mathbb{R}^{m \times d}$ is a subset of V. It contains feature vectors corresponding to m irrelevant objects. Each row of the context C, denoted as $\mathbf{c}_i$, is a feature vector corresponding to an irrelevant object.

The context objects are identified using the question reasoning steps. For example, in order to answer the question “What color is the statue in front of the trees", we need to first find the tree, then find the statue in front of the tree and finally query its color. The GQA dataset [13] provides the ground-truth reasoning steps for each question, as well as the selected objects after each step. We use those reasoning steps to filter out all the relevant and irrelevant objects for the question. Then Intersection-over-Union (IoU) ratio is used to match the predicted objects with the ground-truth ones.

2.3.2 VQA Model with Perfect Sight

We conjecture that the model robustness is related to the quality of visual perception. The majority of the current VQA models use the object features described above as visual input to the model. The features are extracted by a pretrained off-the-shelf object detector which is not updated in VQA training. These pre-extracted features may contain a large amount of noise and miss out on important information that is required to answer the question. In this case, the model may be forced to learn unreasonable data correlations from irrelevant context to predict the answers correctly, which reduces the robustness of the model.

Therefore, to study the influence of visual perception imperfection, we train a model with perfect sight and compare its behavior with model trained with commonly used detected features. The models with perfect sight are trained using the scene graph annotations in GQA dataset. We replace the object features with the encoding of ground-truth object annotations. More

specifically, for each object, we encode its annotated class label and attributes into one-hot encodings, which are then encoded using GloVe [52] embeddings and finally converted to inner dimension d with a FC layer. The object bounding box coordinates are also converted to same dimension using a FC layer. The final representation of an object i is the average of three parts: $\mathbf{c}_i = Avg(\mathbf{o}_i, \mathbf{a}_i, \mathbf{b}_i)$. $\mathbf{o}_i, \mathbf{a}_i, \mathbf{b}_i \in \mathbb{R}^{1\times d}$ are encodings for object class label, attributes and bounding box coordinates respectively.

2.3.3 SwapMix

Now we introduce our proposed context perturbation strategy: SwapMix. The overall idea is shown in Figure 2.2. First, we describe the broad idea about the method and then we go into details on how we select candidates for context swapping, how we perform context swapping in terms of object class labels and attributes, and finally, how we apply SwapMix as a training strategy to improve robustness of the models.

After discovering the context objects as described in Section 2.3.1, we swap their features with other object features from the dataset. For each context object, we perform two types of context swapping based on (a) class label and (b) attributes. In (a) we swap the object feature with the feature of an object from a different class. For example, we change a *bus* into a *car*. In (b) we swap the object feature with feature of an object from same class but with different attributes. For example, we change a *red bus* with *yellow bus*. For both (a) and (b), we perform k feature swaps per context object, therefore altogether we have $2k$ swaps for each context object. We perform context swapping

iteratively for each irrelevant object in the image and measure the percentage of answer changes.

We control the swapping process to make sure that the new object is compatible with the image. For example, we may want to change a *bus* into a *car*, but we don't want to change it into a *computer* because a *computer* parking on the roadside is unnatural. swapped feature always corresponding to an object from the dataset. The swapped feature resembles a real object and thus this perturbation is equivalent to replacing the irrelevant object with another object in the image. Next, we will provide more details on how we choose candidate features to swap with.

To better describe our feature swapping strategy, we denote each context object feature (each row of context C) as $\mathbf{c}_i(o; a_1, a_2, \ldots)$. Here o is the class label for the object i, and $a_1, a_2, \ldots$ are its attributes. Each object belongs to a unique object class while it can have an arbitrary number of attributes. For example in Figure 2.2, feature corresponding to the large red bus can be written as $\mathbf{c}(bus; red, large)$.

2.3.3.1 Swapping the context class.

We swap the object feature with the feature of an object from a different class. For example in Figure 2.2 we swap the *bus* to *car*, *motorcycle*, etc. This type of feature swapping is similar to putting an object of a different class in place of the irrelevant context object in the image. The context swapping helps us understand the dependency of the VQA model on irrelevant objects in the visual input.

To ensure that the swapped class is in the similar domain as object of interest, we only swap the object into similar classes. To achieve this, for each context object, we find the k nearest classes to its class name. More specifically, we compute cosine similarities between GloVe embeddings of class names and pick the top k class labels. Additionally, we set a threshold (0.5) to filter out the classes with small similarities [2]. In this way, we get the candidates for swapping each context object and ensure the selected matching classes are in the similar domain. For example, for the class *car*, its top-5 similar classes are: *truck, motorcycle, vehicle, taxi, bus*.

For each context object $\mathbf{c}_i(o; a_1, a_2, \ldots)$ with class o, we select the k nearest class labels to o for swapping as described above. For each of the k classes, we randomly pick one object from the dataset that belongs to that class. This results in k candidate features for swapping: $\hat{\mathbf{c}}_i^j(o_j; a_1^j, a_2^j, \ldots)$ where $j = \{1, 2....k\}$. Then by swapping $\mathbf{c}_i$ to each of the $\hat{\mathbf{c}}_i^j$, we get k perturbations for each irrelevant context object.

To perturb the perfectly sighted models trained with the encoding of ground truth object class and attributes, a straight-forward way is to simply modify the one-hot encoding for the class label. However, this might cause the object names to be incompatible with attributes, such as *pink elephant* or *green basketball*. Therefore, to ensure that the swapped context corresponds to a real object, we pick a random object of the swapped class from the dataset and use its attributes to generate one-hot encodings for the swapped object

[2] For cases where the number of matching classes with threshold 0.5 are less than k, we select random classes from the datasets to make number of swaps exactly k. We also tried not padding with random classes. The experimental results are similar for the two settings.

attributes.

2.3.3.2 Swapping the context attributes.

To further study the context reliance, we change the object attributes while keeping the object class unchanged. The object feature is swapped with the feature of an object from the same class but with different attributes. For example in Figure 2.2, the *red bus* can be changed to *orange bus*, *yellow bus*, etc. Compared with object class swapping, attribute swapping can be viewed as a more controlled perturbation that helps reveal the models' reliance on the context in more detail.

For each context object $\mathbf{c}_i(o; a_1, a_2, \ldots)$, we swap it with k objects of same class label but different attributes. To get the k swapping candidates, we randomly select k objects from the list of objects belonging to same class with different attributes from the dataset [3]. This results in $\hat{\mathbf{c}}_i^j(o; a_1^j, a_2^j, \ldots)$ where $j = \{k+1, k+2, \ldots, 2k\}$. The object o remains the same across all context swaps.

To perturb the model with perfect sight, we just change the one-hot encodings for the attributes. We pick top k attributes which are similar to attribute of interest using GloVe similarity. For example, the attribute *black* can be swapped with: *blue, green, red, purple, yellow*.

Algorithm. Let $\mathbf{c}_j \in \mathbb{R}^{1\times d}$ be the context corresponding to j^{th} irrelevant object. The aim is to swap $\mathbf{c}_j$ with context $\mathbf{c}_p(o^p; a_1^p, a_2^p, ...)$ belonging to an object p from the dataset. We define a matrix for swapping, $S \in \mathbb{R}^{m\times d}$, where

[3] If the list of objects belonging to same class is less than k, let's say k', we perform only k' feature swapping for that object.

each row of S is equal to $\mathbf{c}_p$. We perform feature swapping to convert C to C^p with the following operation :

$$C^p = C \odot P + S \odot P^c$$

$P \in \{0,1\}^{m \times d}$ is the perturbation matrix and $\odot$ is Hadamard product [53], also known as element wise matrix multiplication. All the rows of P are 1's except j^{th} row corresponding to c_j. P^c is complementary matrix of P, P^c = J - P, where J is the matrix with each entry being 1. Thus, all entries of P^c are 0's except for j^{th} row. Effectively, we modify the context C to C^p by changing context of j^{th} row, from $\mathbf{c}_j$ to $\mathbf{c}_p$.

Summary. For each context object, we get k swaps for its class labels and another k swaps for attributes. Thus $2k$ context swaps are performed for each irrelevant object. To generate one perturbation for an image, we only perturb one context object at a time. Given m context objects in the image, we perform $m * 2k$ perturbations. This is detailed testing on the model to check if it depends on context for predictions. The results of these $m * 2k$ perturbations are used to measure the robustness of the model.

2.3.4 SwapMix as a training strategy.

We can further use SwapMix to improve the robustness of the model. We use SwapMix during training to augment the training images. The model sees a new version of the image at every epoch based on context swapping. Using SwapMix with training, we force the model to pay less attention to context, C, and focus on relevant objects in the image to answer the questions.

During training, we swap the feature vectors belonging to context with other feature vectors from the dataset. We identify the context and perform context swapping based on (a) class label and (b) attributes in the same way as explained in the above sections. We perform context swapping on some irrelevant objects. For every irrelevant object, we decide to swap a feature with a probability of $p = 0.5$. If selected for context swapping, we decide if we have to perform context swapping based on the class label with a probability of $p = 0.5$, otherwise, we perform context swapping based on attributes.

SwapMix training can be performed on both models trained with Faster-RCNN features and model with perfect sight. We add a new function in the data loading part of training, which changes the context in the image. As we do context swapping for every image during data loading, the training time increases by a factor of 1.4 times. In our analysis, we show that both the robustness of the model and the effective accuracy increases using SwapMix on both FasterRCNN and Perfect Sight embeddings.

	MCAN			**LXMERT**		
	Acc.	Context Reliance	Effective Acc.	Acc.	Context Reliance	Effective Acc.
Faster RCNN	70.55	45.05	38.77	83.78 [4]	10.10	75.32
Perfect Sight	90.34	16.40	75.53	91.58	18.85	74.31
Faster RCNN + SwapMix	61.04	26.94	44.61	83.72	7.31	77.60
Perfect Sight + SwapMix	88.10	11.65	77.83	91.45	17.34	75.59

Table 2.1: Results for diagnosing the context reliance for MCAN [1] and LXMERT [2] models. We study models trained with both FasterRCNN features and perfect sight embeddings. Here *Context Reliance* is the percentage of correctly-answered questions that are successfully perturbed by SwapMix; *Effective Acc.* is the context-robust accuracy.

2.4 Experiment

2.4.1 Dataset and Experiment Setup

Dataset. Our experiments are based on the GQA dataset [13]. GQA train split contains 72140 images with 943k questions and val split contains 10243 images with 132k questions. The dataset provides annotated scene graphs for each image and ground-truth reasoning steps for each question. We leverage the reasoning steps to identify visual context and leverage the scene graph annotation to train models with perfect sight. We train the models on GQA train set and test them on GQA val test. GQA also has a test-dev split and a test split, which are not used in our work because they do not have scene graph and reasoning step annotation.

Models. Among the many different VQA models, in this work, we focus on two representative models: MCAN [1] and LXMERT [2]. MCAN is a representative of attention-based models, which contains self-attention and guided-attention units to model the intra-modal and inter-modal interactions between visual and question input. LXMERT is a representative of large-scale pretrained models which can be then finetuned to solve a set of downstream tasks.

Implementation Details. We use the official released code for both MCAN and LXMERT models. We finetune both MCAN and LXMERT pre-trained

[4]We note that LXMERT finetuned with Faster RCNN features has higher accuracy and shows high robustness towards SwapMix perturbation. This is because we test both the models on GQA val split and LXMERT was pretrained with five large vision-language datasets where it has seen images in GQA val set during pretraining. This is reported in the codebase. Therefore the LXMERT results with Faster RCNN features needs to be viewed with cautious.

models using FasterRCNN features on the GQA train set using the default hyperparameters as described by respective authors. For training models with perfect sight, we get ground-truth object names and attributes from scene graph annotation in GQA dataset. MCAN with perfect sight takes a total of 50 epochs to converge and LXMERT model takes 6 epochs. For LXMERT, we use the object features provided by its authors in the official codebase. For MCAN, we used the object features released with GQA dataset.

Evaluation metrics. We introduce 2 new metrics to evaluate the model robustness, *context reliance* and *effective accuracy*. As explained in Section 2.3.3, we apply $2mk$ perturbations for each question where m is the number of irrelevant objects in the image and $2k$ is the number of feature swaps per irrelevant object. We consider a question relying on context if its answer changes for any for the $2mk$ perturbations. Based on this definition, *context reliance* is the percentage of context-relying questions that are originally answered correctly. *Effective Accuracy* is the percentage of questions that are consistently predicted correctly and survive all $2mk$ SwapMix perturbations. Mathematically, it can be written as *effective Acc* = $\sum_i^N q_i / N$, where N is the total number of questions in the dataset and q_i is defined as:

$$q_i = \begin{cases} 0, & \text{if } gt \neq Model(V^j, Q) \text{ for any perturbation j} \\ 1, & \text{otherwise.} \end{cases}$$

		MCAN		LXMERT	
		Class Rel.	Attr Rel.	Class Rel.	Attr Rel.
k=5	Faster RCNN	32.52	28.41	7.29	7.61
	Faster RCNN + SwapMix	18.19	17.10	5.78	5.94
	Perfect Sight	11.11	3.16	14.57	1.34
	Perfect Sight + SwapMix	7.64	2.36	12.89	1.32
k=10	Faster RCNN	39.40	34.47	8.18	8.81
	Faster RCNN + SwapMix	22.18	20.91	6.27	6.58
	Perfect Sight	15.52	3.71	18.82	1.35
	Perfect Sight + SwapMix	10.89	2.72	17.28	1.36

Table 2.2: Detailed analysis of reliance on context. Here we measure the reliance on (a) class names and (b) attributes of irrelevant objects on model prediction. We provide analysis on k perturbations on context for each irrelevant object.

2.4.2 SwapMix Perturbation Results

We finetune the MCAN model and LXMERT model on GQA training split with object features extracted by pretrained Faster RCNN. After finetuning, MCAN reaches 70.55% accuracy and LXMERT reaches 83.78% accuracy on GQA validation split. These results are comparable with ones reported by original authors.

Then we perform SwapMix perturbation to extensively test models' reliance on context. The evaluation results for both the MCAN model and LXMERT model are shown in Table 2.1. For measuring robustness, context reliance and the effective accuracy are reported. Surprisingly, 45% of MCAN answers get changed after perturbation and the effective accuracy drops significantly from 70.55% to 38.77%. The significant drop suggests that the MCAN model relies heavily on the context and is not robust to context swapping. On the contrary, LXMERT is more robust. We conjecture this is because LXMERT

is pretrained on a large amount of image-text pairs from five vision-and-language datasets [2] and the large-scale pretraining equipped the model with better robustness.

Next, we study VQA models with perfect sight. We train both models with perfect sight using encodings of ground truth object names and attributes. As shown in table 2.1, both models achieve more than 90% accuracy with perfect sight. We observe a significant improvement in robustness of MCAN with perfect sight: its context reliance drops by 28.7% compared with training on Faster RCNN features (from 45.1% to 16.4%) and the effective accuracy improves from 38.77% to 75.53%. This suggests that models trained with perfect sight are more robust than its FasterRCNN counterparts when trained with the same amount of data. It is also noticeable that the LXMERT performance is in a similar range with MCAN, which suggests that LXMERT is no more robust than MCAN without seeing more pretraining data in the same domain.

In Table 2.2, we provide more detailed results of perturbations on object class and attribute separately. Interestingly, we observe that models with perfect sight are highly robust to *attribute* perturbations: only 3.7% and 1.4% of the correct answers get changed by attribute perturbation for MCAN and LXMERT respectively. This suggests that given the ground-truth class name, the model can distinguish the relevant and irrelevant objects well, thus are robust to perturbation on the attributes of context objects.

To further support our claim of generalisation of SwapMix, we tested our approach on OSCAR [5]. We see 26.3% of OSCAR answers relies on visual context. The results are consistent with our initial results that transformer

models are more robust than MCAN.

2.4.3 SwapMix Training Results

Using SwapMix as a training data augmentation strategy consistently improves the robustness of both models in all settings. For both MCAN and LXMERT, trained with both FasterRCNN features and perfect encodings, SwapMix training reduces the models' reliance on context and boosts the effective accuracy.

As shown in Table 2.1 (marked as +SwapMix), SwapMix training significantly decreases the context reliance of MCAN by 40% (from 45% to 27%) and increases its effective accuracy by 5.8% (row 3). The results are also consistent for MCAN with perfect sight and LXMERT. Table 2.2 further shows that SwapMix training improves robustness in both context class reliance and attribute reliance. The results consistently show that SwapMix as a training strategy decreases model reliance on context, encourages model robustness and improves effective accuracy.

Interestingly, we notice that there is a trade-off between model robustness and overall accuracy. While we see significant improvement in model robustness, it is noticeable that the overall model accuracy drops to some extent. For example, when applying SwapMix training to MCAN model with perfect sight, its context reliance reduces by 4.7% and effective accuracy improves by 2.3%, while the overall model accuracy drops by 2.2%. The model utilizes biases and correlations in context to achieve high performance, thus when the

context reliance is reduced by SwapMix training, the effective accuracy improves while the overall accuracy drops. Hereby we suggest that the effective accuracy is a better description of the models' true ability to understand the task without relying on context bias.

2.4.4 Ablations and Analysis

Ablating the swapping number k. In Table 2.2, we additionally provide ablation study results for the hyperparameter k, which is the perturbation number. The results for $k = 5$ or $k = 10$ are shown in the table. We do k perturbations on class names and attributes of context objects and report the percentage of questions affected. The result shows that when we increase the perturbations number of k from 5 to 10, the reported answer changes increase accordingly for both models, which is expected. Whereas it is also notable that the reliance increase is not significant, showing that most reliance on context can be revealed with a relatively small number of perturbations. By default, we use k=10 to benchmark reliance on the context of VQA models.

Random padding to k swaps. When doing object name perturbation, for cases where number of compatible classes is less than k, we select random classes from the dataset to pad the perturbation number to exactly k. To verify that this random padding does not bring extra noise in the result, we compare results with and without random padding. As shown in table 2.3, the effect of random padding is negligible.

Examples for SwapMix Perturbation. In Figure 2.3, we show examples for our proposed SwapMix perturbation. Example (a) is based on class name

MCAN			
		Random	w/o random
k=10	FRCNN	45.1	40.3
	FRCNN + SwapMix	26.9	24.3
k=5	FRCNN	38.1	35.0
	FRCNN + SwapMix	22.4	20.8

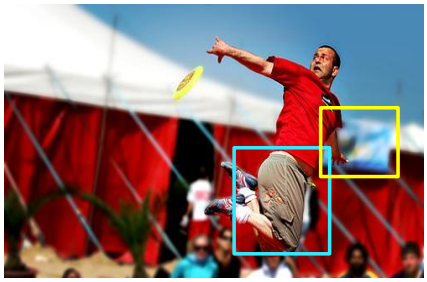
Table 2.3: Context reliance measured by SwapMix with and without random padding to generate k perturbations. This table verifies that random padding does not lead to significant difference.

swapping and example (b) is based on attribute swapping, both of which resulted in the change of model prediction. In example (a), the boot is irrelevant to the question about sweater color, while changing boots into snow boots results in a change in model prediction. In example (b), when we swap the blue signboard in the background with a green signboard, the model prediction on the short's color changed to green as well. The examples are based on the results of MCAN model with perfect sight. The examples intuitively show that VQA models rely heavily on context and by perturbing irrelevant context in the image, we can change model prediction.

Attention visualization for SwapMix training. Training using SwapMix as data augmentation reduces the models' reliance on context. In Figure 2.4, we show the visualization of attention weights for models trained without SwapMix and models trained using SwapMix as data augmentation. The visualization is based on the LXMERT model with perfect sight. For the given question, "Is the camera silver or tan", a model with vanilla training pays more attention to irrelevant context objects such as the car, the tree, etc., while model trained with SwapMix augmentation focuses highly on the

Question: Is the sweater blue?
Swap: boot → snow boot
Answer change: No → Yes

(a) Context object class swap



Question: The shorts have what color?
Swap: signboard blue → green
Answer change: Grey → Green

(a) Context attribute swap

Figure 2.3: Examples for SwapMix perturbationson GQA val split. Blue boxes show relevant objects and yellow boxes show context objects. In (a) we perform class name perturbation and change boots to snow boots. In (b) we perform attribute perturbation and change color of signboard from blue to green. Both these SwapMix perturbations change model prediction.

relevant object, camera, and pays very little attention to other objects. The visualization qualitatively shows that when applied as data augmentation strategy, SwapMix effectively suppresses the model's dependency on visual context and forces the model to focus more on relevant objects.

2.5 Conclusion

In this work, we study the reliance of VQA models on context, i.e. irrelevant objects in the image for prediction. We propose a simple yet effective perturbation technique: SwapMix. SwapMix is effective in both diagnosing model robustness on context reliance, and regularizing the context reliance of VQA models thus making them more robust. Our experiments of two representative models on GQA show the effectiveness of SwapMix. Interestingly, we

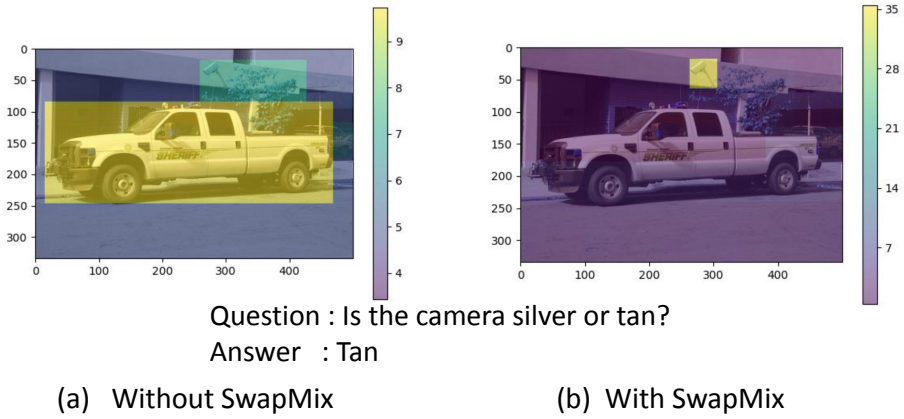


Figure 2.4: Visualization of attention weights for models (a) without SwapMix training (b) with SwapMix training. SwapMix training effectively suppresses models' reliance on visual context.

find that the robustness of VQA models highly depends on the quality of visual perception and models with perfect sight are more robust to context perturbation. Large-scale pretraining also helps improve model robustness. We hope that our initial analysis on reliance on visual context can serve as a starting point for future researchers to study VQA robustness and reliability.

Negative impact and limitations. Our work study the robustness of VQA models and find that the models are vulnerable to context perturbations. The proposed SwapMix perturbation strategy may be used maliciously to attack VQA models. To overcome this potential negative impact, we suggest that training with SwapMix can effectively regularize reliance on context and that better visual representation may improve model robustness. The limitation of our work is that we only study two representative models, using two types of visual features on the GQA dataset.

Chapter 3

Super-CLEVR: A Virtual Benchmark to Diagnose Domain Robustness in Visual Reasoning

VQA models often perform poorly on out-of-distribution data and struggle on domain generalization. Due to the multi-modal nature of this task, multiple factors of variation are intertwined, making generalization difficult to analyze. This motivates us to introduce a virtual benchmark, Super-CLEVR, where different factors in VQA domain shifts can be isolated in order that their effects can be studied independently. Four factors are considered: visual complexity, question redundancy, concept distribution and concept compositionality. With controllably generated data, Super-CLEVR enables us to test VQA methods in situations where the test data differs from the training data along each of these axes. We study four existing methods, including two neural symbolic methods NSCL[54] and NSVQA[55], and two non-symbolic methods FiLM [56] and mDETR[57]; and our proposed method, probabilistic NSVQA (P-NSVQA), which extends NSVQA with uncertainty reasoning. P-NSVQA

outperforms other methods on three of the four domain shift factors. Our results suggest that disentangling reasoning and perception, combined with probabilistic uncertainty, form a strong VQA model that is more robust to domain shifts. The dataset and code are released at https://github.com/Lizw14/Super-CLEVR.

3.1 Introduction

Visual question answering is a challenging task that assesses the reasoning ability of models to answer questions based on both visual and linguistic inputs. Current VQA methods are typically developed on standard benchmarks like VQAv2 [58] or GQA [59], with the implicit assumption that testing data comes from the same underlying distribution as training data. However, as has been widely studied in computer vision [60, 61, 62], algorithms trained on one domain often fail to generalize to other domains. Moreover, having learned the distributional prior of training data, models often struggle on out-of-distribution tests. This has been studied in VQA from the perspective of domain transfer [63, 64, 65], dataset bias [66, 8, 37], counter-factual diagnosis [7, 30], and out-of-distribution benchmarking [11].

The multi-modal nature of VQA gives rise to multiple intertwined factors of variation, making domain shift an especially difficult problem to study. For example, [63] suggests that VQA domain shifts are a combination of differences in images, questions or answers; and [67] reveals a gap between synthetic and real VQA datasets by differences in the over-specification of questions and the underlying distribution of concepts. However, despite a

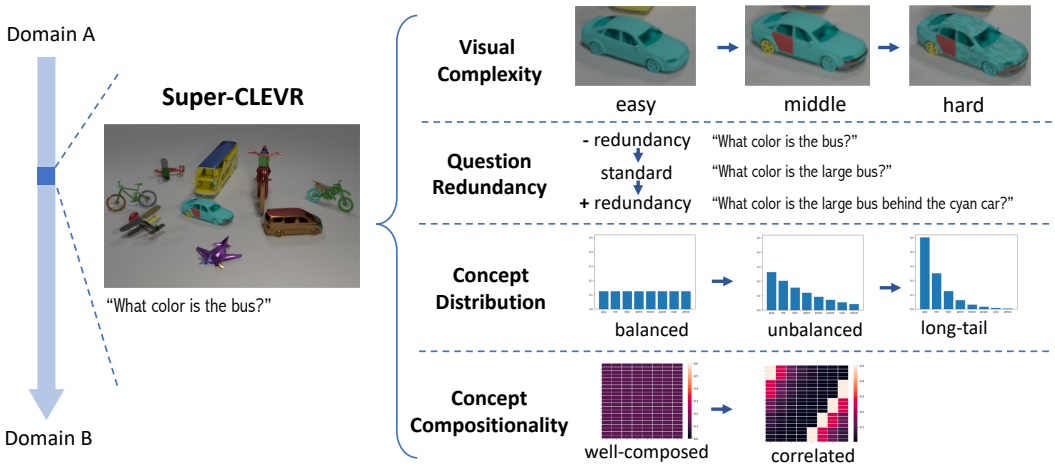


Figure 3.1: We decompose VQA domain shifts into four contributing factors: visual complexity, question redundancy, concept distribution and concept compositionality. The domain shifts along each factor can be independently studied with the proposed Super-CLEVR dataset.

wealth of research on domain generalization in VQA [68, 69, 65, 64], there is no systematic analysis of the contributing factors in domain shifts.

To this end, we introduce a virtual benchmark, Super-CLEVR, which enables us to test VQA algorithms in situations where the test data differs from the training data. We decompose the domain shift into a set of isolated contributing factors, so that their effects can be diagnosed independently. We study four factors: visual complexity, question redundancy, concept distribution, and concept compositionality. These are illustrated in Fig. 3.1 and described in Sec. 3.3.1. With controllable data generation using our Super-CLEVR virtual benchmark, we are able to isolate the different factors in VQA domain shifts so that their effects can be studied independently. Compared with the original CLEVR dataset [70], Super-CLEVR contains more complicated visual components and has better controllability over the domain shift

factors. As shown in Fig. 8.1, the Super-CLEVR dataset contains images rendered from 3D graphical vehicle models in the UDA-Part dataset [71], paired with questions and answers automatically generated from templates. The objects and questions are sampled based on the specified underlying probability distribution, which can be controlled to produce distribution shifts in different factors.

With Super-CLEVR, we diagnose the domain robustness of current VQA models. Four representative models are studied: for the classic two-stream feature fusing architecture, we choose FiLM [56]; for a large-scale pretrained model we take mDETR [57]; we use NSCL [54] and NSVQA [55] as representative neuro-symbolic methods. We observe that all these models suffer from domain shifts to varying degrees of sensitivity. We analyze each factor separately to examine the influence of different model designs. Specifically, we find that the step-by-step design of neural modular methods enhances their robustness to changes in question redundancy compared with non-modular ones; however, the non-modular models are more robust to visual complexity. Furthermore, thanks to its decomposed reasoning and perception, NSVQA is more robust to concept distribution shifts.

While existing models suffer from domain shifts with different characteristics, we make a technical improvement over NSVQA which enables it to significantly outperform existing models on three of the four factors. In particular, we inject probabilities into the deterministic symbolic executor of NSVQA, empowering it to take into account the uncertainty of scene understanding. We name our model *probabilistic NSVQA* (P-NSVQA), and show

that its performance improvement in both the in-domain and out-of-domain settings. With superior results of P-NSVQA, we suggest that disentangling reasoning from vision and language understanding, together with probabilistic uncertainty, gives a strong model that is robust to domain shifts.

Our contributions are as follows. (1) We introduce the Super-CLEVR benchmark to diagnose VQA robustness along four different factors independently. This benchmark can also be used for part-based reasoning. (2) We enhance a neural-symbolic method by taking the uncertainty of visual understanding into account in reasoning. (3) We conduct detailed analysis of four existing methods, as well as our novel approach to study the influence of model designs on distinct robustness factors. We conclude that disentangled reasoning and perception plus explicit modeling of uncertainty leads to a more robust VQA model.

3.2 Related work

Visual question answering. Popular VQA methods fall into three categories. Two-stream methods extract features for image and questions using CNN and LSTM respectively, then enable interaction between the two modalities with different feature fusing methods [50, 18, 23, 72, 73, 56, 74]. Neural symbolic methods, on the other hand, use a parse-then-execute pipeline where the question is parsed into a functional program, which is then executed on the image using neural modules [55, 54]. Recently, transformers-based models have achieved impressive performance on various vision-and-language tasks by pretraining on large scale dataset then finetuning for downstream tasks

[2, 75, 5, 76, 57]. We choose FiLM [56], NSCL[54] and mDETR[57] as category representatives.

VQA datasets. Datasets containing real images and human-written questions have been widely used to benchmark VQA models, *e.g*. VQA [14], VQAv2 [58], Visual 7w [77], VizWiz [78], Visual Genome [79], COCO QA [80], etc. However, subsequent work has revealed the strong prior and bias in those datasets which might be exploited by models to correctly predict the answers without reasoning [7, 81, 8, 9, 58, 11, 12]. Attempts to address this problem include better balancing datasets [66] and creating counterfactual examples [37, 30]. To assess a model's true reasoning ability, the CLEVR dataset [70] proposes to generate complex multi-step questions on synthetic images, which is then extended to various vision-and-language tasks [82, 83, 84, 85, 36, 86, 87]. The GQA dataset [59] extends CLEVR-style questions to real images. Our benchmark is distinct from existing ones because we introduce more complex visual scenes into CLEVR and provide controllability to study domain robustness on isolated factors.

Domain shift in VQA. Domain shift is a long-standing challenge in computer vision, explore in prior works in domain adaptation [88, 89, 60, 90] and domain generalization [61, 62]. Recent works have focused on domain shifts in VQA. [63, 65] improves model adaptation between datasets by feature learning. [64] analyze domain shifts between nine popular VQA datasets and proposes an unsupervised method to bridge the gaps. [67] generalize symbolic reasoning from synthetic to real dataset. [68] introduce a question-answer generation module that simulates the domain shifts. [69] propose a

training scheme X-GGM to improve out-of-distribution generalization. [36] assess models generalization on the CLOSURE of linguistic components. In contrast to prior works we study each of the different domain shift factors independently with our virtual benchmark.

3.3 Super-CLEVR

3.3.1 Motivation: domain shift factors

Visual complexity. A major difference between different VQA datasets is visual complexity. For example, in the CLEVR dataset, objects are simple, atomic shapes while in real-world data, objects are more complex and have hierarchical parts. While hard to quantify, visual complexity is related to various factors, such as object variety, object size, background, texture, lighting, occlusion, view point, etc. In our work, we control visual complexity by introducing more challenging objects that can have distinct attributes associated with their parts, and by optionally pasting various textures onto objects. Examples of generated images with different complexity levels are shown in Fig. 3.1.

Question redundancy. Question redundancy refers to the amount of over-specified or redundant information in the question, which can be in the form of either *attributes* or *relationships*. For example, in Fig. 3.1, "what color is the large bus behind the cyan car", `large` (*attributes*) and `behind the cyan car` (*relationship*) are redundant because there is only one bus in the image. As observed in linguistics and cognitive science [91, 92, 93, 94], human speakers may include over-specified information when identifying a target

object, which has also been studied in referring expression generation [95]. For VQA, as analyzed in [67], a significant difference between synthetic and real datasets is that real questions contain some redundant information, which sometimes is a distraction leading to model prediction errors. Therefore, in this work, we generate questions with different redundancy levels and study the effect of question redundancy on model behaviors.

Concept distribution. The distributions of *concept*, *i.e*. *objects* (*e.g*. car) and *attributes* (*e.g*. large), are distinct across different VQA datasets. For example, while colors are well-balanced in CLEVR dataset, in the GQA dataset, the color distribution is long-tailed where "white" appears > 50 times more frequently than "gold". Long-tailed distributions have been a challenge in many computer vision tasks [96, 97, 98, 99, 100]. In VQA, the long-tailed concept distribution not only hinders the learning of infrequent concepts due to few training samples, but also introduces strong biases and priors in the dataset that may mislead the models. For example, "tennis" is the correct answer to most questions with "what sport is ..." [66]. With strong priors in data, it is hard to assess the true reasoning capacity of current models. While previous works address this problem by carefully re-balancing datasets [66], in our work, we controllably vary the concept distribution in our dataset and study model robustness to concept distribution shifts.

Concept compositionality. Concept compositionality refers to how different concepts (shapes, attributes) compose and co-occur with each other, *e.g*. roses are usually red while violets are usually blue [11]. Concept compositionality can be viewed as a conditional concept distribution in the context

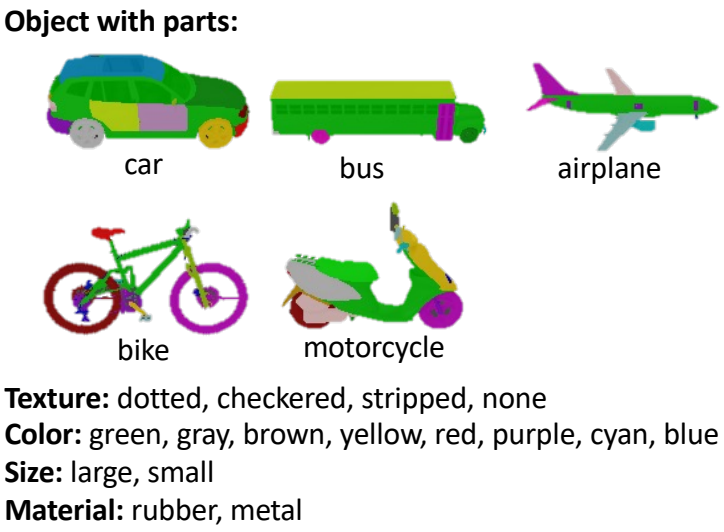


Figure 3.2: Super-CLEVR contains 21 vehicle models belonging to 5 categories, with controllable attributes.

of other concepts. Shifts in concept compositionality impede the generalization of VQA models. For example, the model may fail to recognize a *green* banana because most bananas are *yellow* in the training data [101]. Previous works evaluate the out-of-distribution performance by collecting counterfactual testing examples [7]. In our work, we control the compositionality of *shapes* and *colors* with an intuitive motivation: if, for example, in the training data, bicycles are red and cars are blue, will the models be able to recognize blue bicycles and red cars in testing?

3.3.2 Dataset generation

Super-CLEVR follows a similar data generation pipeline as CLEVR, but with more complex visual components and better control of domain gap factors. We describe the generation procedure below.

Objects with parts. To improve the visual complexity of CLEVR scenes, we replace the simple shapes (*e.g.*, *cube*, *sphere*) in CLEVR dataset with vehicles from UDA-Part dataset [71]. There are 21 vehicle models, belonging to 5 categories: *car*, *motorbike*, *aeroplane*, *bus*, and *bicycle*. Each 3D model comes with part annotations, *e.g.*, *left front wheel* or *left right door* for *car*. Examples for the vehicle models are shown in Fig. 3.2. We remove or merge small parts from the original annotations to avoid severe difficulty in visual understanding. The full object and parts list is in the supplementary material.

Attributes. Besides the attributes in the original CLEVR dataset, *i.e. color*, *material*, *size*, we optionally add *texture* as an additional attribute to increase visual complexity. Note that in order to enable part-based questions, the attributes (color or material) of object parts can be different from that of the object. For example, a blue car can have a red wheel or a green door. In this case, the attribute of the holistic object refers to the attribute of its main body (*e.g.* the blue car has blue frame).

Scene rendering. Following CLEVR, each scene contains 3 to 10 objects. The objects are placed onto the ground plane with random position and orientation. When placing the objects, we ensure that the objects do not overlap with each other and we avoid severe occlusion by thresholding the number of visible pixels for each object. Random jitters are added to lamp and camera positions. When rendering, we also save the ground-truth bounding boxes and segmentation masks for each of the objects and their parts, which are required when training some of the models.

Question generation. Super-CLEVR follows similar question generation

pipeline in CLEVR, which instantiates question templates using the underlying reasoning program that can be operated on the scene graph. For example, the program `select_shape(truck)` $\rightarrow$ `query_color(·)` can be instantiated as question "what is the color of the truck". Therefore, redundancy level of questions can be controlled by removing or adding redundant reasoning steps in the underlying reasoning program.

3.3.3 Controlling the dataset

To study domain generalization, we generate several variants of the dataset for each of the domain shift factors. The variants of the datasets serve as different data domains to test the model robustness. Here we describe the method for controllably generating the dataset variants.

Visual complexity. We generate three variants of the dataset with different levels of visual complexity: *easy*, *mid* (middle) and *hard*. The only difference between the 3 versions is visual complexity: for the easy version, objects with different sizes, colors and materials are placed into the scene; for the middle version, we choose 3 parts on each object that are visible and randomly change their attributes; for the hard version, we further add random textures to the objects and parts. An example of the 3 dataset versions can be found in Fig. 3.1. Note that the scene layout and the questions are shared, so that the influence of visual complexity can be isolated and studied independently.

Question redundancy. Three variants of the dataset with different redundancy levels are generated: *rd-*, *rd* (default), *rd+*. By default (*rd*), as in

original CLEVR dataset, the questions contain some redundant *attributes* resulting from random sampling, while all redundant *relationships* are removed. In *rd-*, we also remove all redundant attributes from the questions, leading to no redundancy in the questions. In *rd+*, we add all possible attributes and relationships into the question, so that questions contain a high level of redundancy. For all the variants, the questions are ensured to be valid.

Concept distribution. We generate three dataset variants with different concept distributions: *bal* (balanced), *slt* (slightly unbalanced) and *long* (long-tail distributed). More specifically, we change the distribution of *shapes*, *colors* and *materials* while the distribution of *size* is kept fixed in order to keep visual complexity consistent, since objects with smaller sizes are visually harder to recognize. By default (*bal*), the shapes and attributes are randomly sampled, leading to a balanced distribution. For *slt* and *long*, the concept distribution $\mathbf{d}$ is generated by $d_i = a^{-i}$, where i is the index of the concept. a is a hyper-parameter controlling the length of the tail. A larger a leads to more imbalanced distribution and $a = 1$ leads to flat distribution (cf. Fig. 3.1). For *slt*, $a = 1.3$; for *long*, $a = 2.0$. In addition, to better analyze the performance on the frequent and rare concepts, we generate three variants for testing purpose only: *head* (frequent concepts in the long-tail distribution), *tail* (infrequent/rare concepts), and *oppo* (opposite to the long-tail distribution). We test each model on those three variants to analyze the performance on concepts with different degrees of frequency.

Concept compositionality. We generate 3 versions of the dataset, *co-0*, *co-1* and *co-2*, with different compositions of the 21 *shapes* (from 5 categories)

and the 8 *colors*. The compositionality of the dataset is controlled with the co-distribution matrix $M \in \mathbb{R}^{21\times 8}$, where each entry M_{ij} is the probability an object of the i-th shape has the j-th color. Entries in each row of M sum up to 1. In the version *co-0*, M is a flat matrix so that the shapes and colors are randomly composed. In *co-1*, each shape in one category has a different color distribution, *e.g*. truck and sedan, while shapes from different categories may share the same color distribution *e.g*. sedan and airliner. Oppositely, in *co-2*, we make the shapes in same category have the same color distribution, whiles shapes from different categories have different distributions. The motivation is that since shapes from the same category are visually similar, the difference in *co-1* and *co-2* will help analyze the difference in model predictions on visually similar objects and dissimilar objects when composed with different color distributions.

Dataset Statistics. Every dataset variant contains 30k images, including 20k for training, 5k for validation and 5k for testing. Each image is paired with 10 object-based and 10 part-based questions. By *default*, the dataset refers to the version with *mid* visual complexity level (*i.e*. objects are untextured and has up to 3 parts with distinct attributes), *rd* redundancy level, balanced (*bal*) concept distribution and random (*co-0*) compositionality. More dataset statistics are in supplementary materials.

3.4 Evaluated methods

3.4.1 Simple baselines

Random, Majority. These simple baselines pick a random or the most frequent answer for each question type in the training set as the predicted answer.

LSTM. This question-only baseline encodes the question word embeddings with LSTM [102] and predicts answer with an MLP on top of the final hidden states of the LSTM.

CNN+LSTM. The image is represented with features extracted by CNN and question is encoded by LSTM. An MLP predicts answer scores based on the concatenation of image and question features.

3.4.2 Existing models

FiLM. We choose *Feature-wise Linear Modulation* [56] as a representative of classic two-stream feature merging methods. The question features extracted with GRU [103] and image features extracted with CNN are fused with the proposed FiLM module.

mDETR. mDETR [57] is a transformer-based detector trained to detects objects in an image conditioned on a text query. The model is pretrained with 1.3M image and text pairs and can be finetuned for various downstream tasks like referring expression understanding or VQA.

NSCL. The *Neuro-Symbolic Concept Learner* [54] is a representative neural symbolic method. NSCL executes neural modules on the scene representation based on the reasoning program, during which the modules learns

embeddings of each concept with the answer supervision.

NSVQA. *Neural-Symbolic VQA* [55] is a neural symbolic method composed of three components: A scene parser (Mask-RCNN [104]) that segments an input image and recovers a structural scene representation, a question parser that converts a question from natural language into a program and a program executor that runs the program on the structural scene representation to obtain the answer. Notably, compared to NSCL, the individual components of NSVQA can be learned separately, hence, for example the scene parser can be learned from data that does not necessarily have Visual-Question annotations.

3.4.3 Probabilistic NSVQA (P-NSVQA)

Since the program executor in NSVQA is a collection of deterministic, generic functional modules, it can be augmented with a probabilistic reasoning process that takes into account the confidence of the predictions of the scene parser. This allows the model to execute the program that has the largest joint likelihood, instead of only taking the maximal likelihood execution at each step of the program. The experiment results demonstrate a significant performance improvement of this probabilistic approach over the deterministic NSVQA model proposed in [55].

In particular, we interpret the confidence of the Mask-RCNN output as a likelihood function for all detected object classes p_{object} and their attributes p_{att}. Moreover, we define a likelihood $p_{spatial}$ for the spatial relations between objects (behind, in front, left, right) that is proportional to the distance between the centers of two bounding boxes. Given a reasoning program containing

multiple reasoning steps, we execute each step based on the scene parsing likelihood and produce an step-wise output with confidence. Finally, we use a factorized model, multiplying the output for all the steps to get the final answer prediction. We refer readers to the Appendix for more details.

3.4.4 Implementation details

Training mDETR requires ground-truth grounding of question tokens to image regions, which is available in Super-CLEVR. NSCL requires bounding box of objects, which can predicted using a trained Faster RCNN, and the reasoning program, which can be parsed using a trained parser. Similarly, ground-truth programs are used for training NSVQA and P-NSVQA. Note that we empirically find that the question-to-program parsing is a relatively easy task ($>$ 99% accuracy using a simple LSTM), so we focus more on models' reasoning ability in our analysis.

Unless specified, the models are trained with default setting as in the official implementation. FiLM is trained for 100k iterations with batch size 256. mDETR is trained for 30 epochs with batch size 64 using 2 GPUs for both the grounding stage and the answer classification stage. NSCL is trained for 80 epochs with batch size 32. For NSVQA and P-NSVQA, we first train the object parser (Mask RCNN [104]) for 30k iterations with batch size 16, then train the attribute extraction model (using the Res50 backbone) for 100 epochs with batch size 64. For P-NSVQA, when counting the objects or determining whether objects exist in the scene, we use a threshold (0.7) to obtain the final selected objects. Early stopping is used based on validation accuracy. All the

models are trained with 200k questions. We repeat experiments on *default* split for 3 times with different random seeds and get *std* of ± 0.10 (P-NSVQA) and ± 0.40 (NSVQA), showing the statistical significance of our results, then we only run other experiments once.

3.5 Results and analysis

In this section, we first show evaluation results for in-domain setting, then provide results and analysis on out-of-domain evaluation. Finally, we describe additional studies and future works.

3.5.1 In-domain results

In-domain evaluation refers to the setting where the training and testing data come from the same domain (the *default* dataset variant in this case). We compare the in-domain results on Super-CLEVR and CLEVR. The results are shown in Fig. 3.3.

For all the models, the performance is lower on Super-CLEVR than CLEVR, suggesting that Super-CLEVR is a more challenging benchmark. The scenes with vehicles are much harder visually for the models to understand compared with the simpler shapes from CLEVR. Note that the performance gap on two datasets for simple baselines (Random, Majority, LSTM, CNN+LSTM) is smaller than for the other better models. This is because Super-CLEVR contains more object types, and therefore the performance of simply guessing is lower than on CLEVR.

When comparing the performance of different models, we find that the

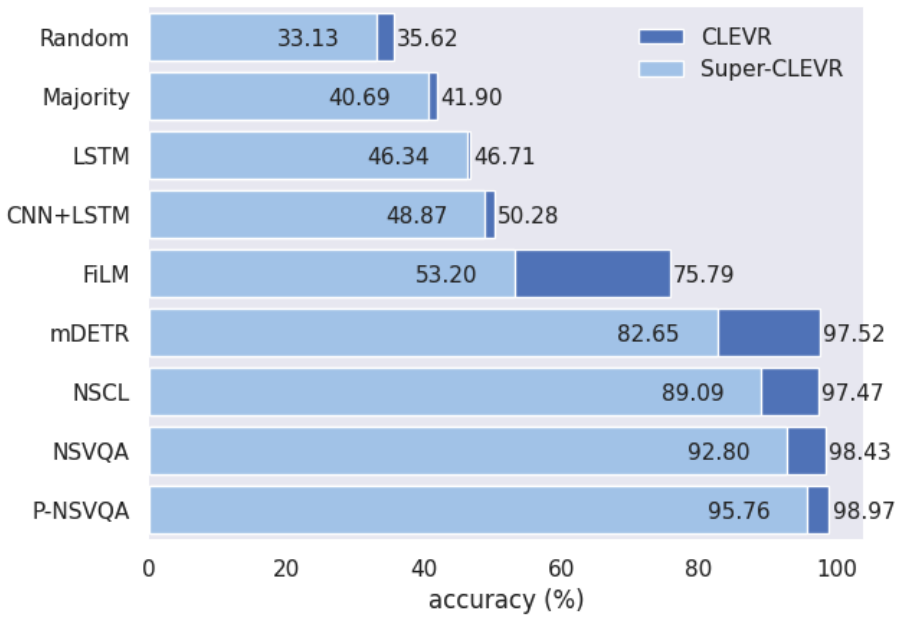


Figure 3.3: Comparison of models' accuracy on Super-CLEVR and the original CLEVR dataset.

neural modular methods, *i.e.* NSCL, NSVQA, P-NSVQA, perform much better than non-modular ones. This is not surprising given their nearly perfect performance on the original CLEVR dataset, which shows its strong ability to model synthetic images. The large-scale pretrained grounding model mDETR, which is a leading model on both real and synthetic images, also achieves good performance (82.7%) on Super-CLEVR. The two-stream method FiLM does not achieve very strong performance (53.2%), but is still much better than the other simple baselines.

Our proposed P-NSVQA outperforms all the other models. In particular, on Super-CLEVR, it outperforms its deterministic counterpart, NSVQA, by 2.96%. This shows the advantage of taking into account the probabilities when the scenes are challenging thus the model's uncertainty of predictions can be

utilized.

3.5.2 Out-of-domain results

	FiLM			mDETR			NSCL			NSVQA			Prob NSVQA		
Visual Complexity															
	easy	mid	hard	easy	mid	hard	easy	mid	hard	easy	mid	hard	easy	mid	hard
easy	**59.96**	53.95	50.66	**93.36**	84.30	82.97	**95.13**	92.31	90.81	**95.19**	94.19	94.09	**96.76**	95.98	96.37
mid	**57.41**	53.28	50.18	**83.34**	82.36	81.27	84.5	**89.10**	86.33	81.99	92.80	**93.78**	86.25	**95.76**	95.11
hard	**55.95**	53.11	50.47	79.71	79.94	**80.71**	76.85	78.66	**85.08**	73.11	79.71	**92.65**	79.81	86.47	**95.36**
Question Redundancy															
	rd-	rd	rd+	rd-	rd	rd+	rd-	rd	rd+	rd-	rd	rd+	rd-	rd	rd+
rd-	51.42	52.54	**53.51**	**83.94**	80.37	66.28	88.64	88.82	**90.33**	**92.95**	92.94	92.67	95.66	**95.72**	95.43
rd	50.39	53.28	**54.78**	**82.77**	82.36	70.36	88.45	89.10	**91.45**	91.19	**92.78**	92.14	94.87	**95.72**	95.43
rd+	46.14	52.30	**71.47**	78.48	84.05	**90.42**	87.94	88.34	**91.16**	91.38	91.96	**92.80**	94.88	95.47	**95.72**
Concept Distribution															
	bal	slt	long	bal	slt	long	bal	slt	long	bal	slt	long	bal	slt	long
bal	50.47	53.04	**54.35**	**80.71**	75.79	74.54	**85.08**	83.79	75.10	**92.65**	90.82	83.74	**95.36**	94.89	89.88
long	49.43	54.75	**62.96**	79.06	80.29	**90.66**	85.33	89.42	**91.10**	92.73	**93.38**	92.53	96.31	**96.32**	95.25
head	48.60	58.06	**61.60**	80.75	79.60	**87.46**	84.58	88.39	**90.19**	93.87	**94.82**	92.48	96.42	**96.80**	95.92
tail	**51.80**	48.70	50.08	**81.50**	70.88	60.94	**86.10**	80.27	60.55	**90.26**	89.20	75.32	**94.08**	93.20	82.68
oppo	**49.06**	48.93	46.68	**79.13**	68.37	56.98	**85.07**	77.86	55.14	**91.22**	88.65	71.32	**95.76**	94.09	79.74
Concept Compositionality															
	co-0	co-1	co-2	co-0	co-1	co-2	co-0	co-1	co-2	co-0	co-1	co-2	co-0	co-1	co-2
co-0	53.28	**57.00**	56.1	**83.36**	77.03	82.43	**89.1**	82.52	83.77	**92.80**	90.11	91.59	**95.76**	94.02	95.12
co-1	52.41	**60.57**	56.67	79.46	82.45	**83.93**	78.89	**87.18**	84.2	78.74	89.99	**90.67**	87.12	94.53	**94.78**
co-2	52.96	57.37	**60.53**	80.03	77.41	**87.24**	78.40	81.55	**88.84**	77.85	89.28	**92.23**	87.19	93.49	**95.61**

Table 3.1: Accuracy of models trained and tested on different domains. Column headings indicate *training* settings, while rows indicate the dataset variant for *testing*. The best performance in each row (*i.e.* the best training setting) is marked in **bold** and best performance in each column (*i.e.* the best testing setting) is underlined. Description for different splits is in Sec. 3.3.3 and analysis is in Sec. 3.5.2.

In this section, we train and test the five models (FiLM, mDETR, NSCL, NSVQA and P-NSVQA) on different dataset variants, and diagnose their domain robustness on each of the four domain shift factors. Please refer to Sec. 3.3.3 for a description of different variants. The validation accuracy is used for analysis here and the results are shown in Tab. 3.1.

All the methods suffer from domain shifts. The results show that the best performance mostly occurs in situations where the model is tested on the same dataset variant as it is trained on, *i.e.* the bold or underlined numbers fall mostly on the diagonals in Tab. 3.1.

We compare the domain robustness of the five models by measuring the relative performance decrease when the testing data differs from the training data, *i.e.* smaller performance drop on different testing domains means better robustness. Based on this intuition, for easier understanding of Tab. 3.1, we propose a measurement metric for domain robustness named ***Relative Degrade (RD)*** to better analyze the results. We define *Relative Degrade* as the the percentage of accuracy decrease when the model is tested under a domain shift, *i.e.* the accuracy drop divided by the in-domain accuracy. Specifically, if a model gets accuracy a under in-domain testing (*i.e.* testing with the same dataset variant as training) and accuracy b under out-of-domain testing (*i.e.* testing with a different dataset variant from training), then $RD = (a - b)/a$. Since we train each model on three data variants, the RD's for the three models are averaged to measure its domain robustness.[1]

Tab. 3.2 shows the *Relative Degrade* of the five models on the four factors. We see that P-NSVQA outperforms other models by a significant margin on three of the four factors, indicating that it has better overall domain robustness. In the following, we take a closer look at the results on each of the factors separately, to diagnose the influence of different model designs.

[1]For concept distributions, we compute relative degrade with a slight change: we compute the accuracy drop from *head* to *tail* and the drop from *long* to *oppo*, take their average, and divide by the accuracy on *bal*.

	Visual	Redund.	Dist.	Comp.
FiLM	**4.03**	21.33	28.46	9.04
mDETR	9.81	19.05	36.34	9.45
NSCL	15.57	0.92	37.44	15.40
NSVQA	17.48	1.72	20.92	11.44
Prob NSVQA	12.88	**0.84**	**13.72**	**7.00**

Table 3.2: *Relative Degrade* under domain shifts, *i.e.* the percentage of accuracy decrease when the model is tested with domain that differs with training. Lower *RD* means better robustness.

Question redundancy. Neural modular methods are much more robust to question redundancy shifts than non-modular ones. The relative degrades for modular methods are less than 2%, while one-modular ones degrade for around 20%. Due to the step-by-step design of the reasoning in modular methods, each reasoning step is independent of the others so that the models are less likely to learn the spurious correlation between question and answers. Therefore the modular methods are less vulnerable to change in question/program length.

Visual complexity. Different from our findings on question redundancy, for domain shifts in visual complexity, non-modular methods are more robust compared to modular ones. As shown in Tab. 3.2, while FiLM and mDETR gets less than 10% degrade, NSCL and (P-)NSVQA degrade for more than 12%. The reason might be that the simple reasoning modules in modular methods can not process the visual signals as well as the dense non-modular models.

Comparing P-NSVQA with NSVQA, we find that injecting probability into

deterministic symbolic reasoning greatly improves the robustness on visual complexity (4.04% decrease in *RD*). This suggests that some errors in visual understanding can be corrected and recovered by taking into account the uncertainty of visual parsing and combining the results of each reasoning step with probability.

Concept distribution. While all the four existing models suffer a lot (larger than 20% *RD*) on domain shifts in concept distribution, we see that the symbolic method NSVQA is better than the other three (by more than 7.5%). With the disentangled reasoning and visual understanding components in NSVQA, the distribution priors in the images and the programs/answers cannot intertwine with each other, which prevent the model heavily relying on the priors. With uncertainty, we can further boost the robustness of NSVQA with a large margin (from 21% to 14% *RD*).

Moreover, the head-tail results suggests that the overall accuracy, which is commonly used to measure VQA performance, should be taken with cautious. When the testing split is imbalanced, the seemingly high accuracy is misleading because the head concepts dominates the testing while the tail ones are not well-reflected. For example, for NSCL, although it gets high accuracy (91%) on the long-tailed data, its performance is only 60.6% on the tail concepts. In real-world datasets, the data are usually not well-balanced, which suggests the value of synthetic testing.

Concept compositionality. Comparing the existing methods, we find that the non-modular methods seems to be more robust than modular methods NSCL or NSVQA. However, with uncertainty, P-NSVQA improves the result

of NSVQA, which even outperforms the non-modular methods. This suggest the large potential of better robustness of modular methods by improving current models.

In summary, while non-modular methods are more robust to visual complexity shifts, the modular symbolic methods (improved with uncertainty) are more robust on the other three factors. By disentangling reasoning with visual understanding, separately executing every each reasoning step then merging the results of the steps using probabilities based on uncertainty, our P-NSVQA outperforms all the existing models in question redundancy, concept distribution and compositionality. Therefore, we suggest that symbolic reasoning with uncertainty leads to strong VQA models that are robust to domain shifts.

3.5.3 More analysis and future work

Synthetic-to-real transfer. We provide an additional proof-of-concept study to show that the findings drawn from Super-CLEVR dataset can transfer to real datasets. In the following experiments, we show our finding that neuro-symbolic methods (NSCL, NSVQA, P-NSVQA) are more robust than mDETR on question redundancy also holds true on the real GQA dataset [59]. More precisely, we progressively removed the redundant operations from the reasoning program in GQA testdev split, and then regenerated questions using a program-to-question generator. Using the change of models' testing accuracy as the redundant operations are removed, we can evaluate the models' robustness towards question redundancy. The results are show

in Tab. 3.3.[2] We observe that the performance drop of mDETR is much larger than neuro-symbolic methods as the redundant information is progressively removed, which indicates that symoblic methods have better question redundancy than mDETR on GQA dataset. This is consistent with our findings on Super-CLEVR.

	0%	14%	32%	70%	91%	100%
mDETR	0	-4.82	-8.46	-13.16	-13.88	-14.56
NSCL	0	-0.14	-0.34	-1.09	-1.71	-2.59
NSVQA	0	-3.47	-4.80	-7.01	-7.02	-7.02
P-NSVQA	0	-1.93	-3.15	-5.73	-5.91	-5.78

Table 3.3: Accuracy drop on the GQA dataset when redundant information is progressively removed.

Reasoning with part-object hierarchies. In addition to evaluating domain generalization, Super-CLEVR can be extended for broader purposes, *e.g.* part-based reasoning. We can ask questions like "what is the color of the front wheel of the bike?", "what is the color of the vehicle that has a yellow wheel", etc. Those questions require the model to correctly understanding the part-object hierarchy, which is an ability that current VQA models lack.

Limitations. The main limitations of our work lie in the synthetic nature of our dataset. Future efforts can be made in collecting better controlled and balanced real datasets for model diagnosis. We emphasize that the purpose of the dataset is for model diagnosis and that models should also be tested on real data.

[2]For implementation of NSCL on a real-word dataset, we use the model in [67] (the version without calibration). The model accuracies on the original not-perturbed GQA testdev split are as following: mDETR (61.67%), NSCL (56.13%), NSVQA (39.58%), P-NSVQA (39.66%).

3.6 Conclusion

We diagnose domain shifts in visual reasoning using a proposed virtual benchmark, Super-CLEVR, where distinct factors can be independently studied with controlled data generation. We evaluate four existing methods and show that all of them struggle with domain shifts, highlighting the importance of out-of-domain testing. Among the evaluated methods, neural modular methods are more robust towards question redundancy. In particular, NSVQA with disentangled perception and reasoning shows better robustness towards distribution and compositionality shifts. We further propose P-NSVQA, which improves NSVQA with uncertainty in the reasoning modules. We show that P-NSVQA outperforms all the existing methods in both in-domain testing and out-of-domain testing. With detailed analysis, our study suggests that disentangling reasoning and perception, combined with probabilistic uncertainty, form a strong VQA model that is more robust to domain shifts. We hope our analysis may facilitate better understanding of strengths and weaknesses of VQA models and, more broadly, future work might explore using the Super-CLEVR benchmark for other tasks like part-based reasoning.

Chapter 4

3D-Aware Visual Question Answering about Parts, Poses and Occlusions

Despite rapid progress in Visual question answering, existing datasets and models mainly focus on testing reasoning in 2D. However, it is important that VQA models also understand the 3D structure of visual scenes, for example to support tasks like navigation or manipulation. This includes an understanding of the 3D object pose, their parts and occlusions. In this chapter, we introduce the task of 3D-aware VQA, which focuses on challenging questions that require a compositional reasoning over the 3D structure of visual scenes. We address 3D-aware VQA from both the dataset and the model perspective. First, we introduce Super-CLEVR-3D, a compositional reasoning dataset that contains questions about object parts, their 3D poses, and occlusions. Second, we propose PO3D-VQA, a 3D-aware VQA model that marries two powerful ideas: probabilistic neural symbolic program execution for reasoning and deep neural networks with 3D generative representations of objects for robust

visual recognition. Our experimental results show our model PO3D-VQA outperforms existing methods significantly, but we still observe a significant performance gap compared to 2D VQA benchmarks, indicating that 3D-aware VQA remains an important open research area. The code is available at https://github.com/XingruiWang/3D-Aware-VQA.

4.1 Introduction

Visual question answering is a challenging task that requires an in-depth understanding of vision and language, as well as multi-modal reasoning. Various benchmarks and models have been proposed to tackle this challenging task, but they mainly focus on 2D questions about objects, attributes, or 2D spatial relationships. However, it is important that VQA models understand the 3D structure of scenes, in order to support tasks like autonomous navigation and manipulation.

An inherent property of human vision is that we can naturally answer questions that require a comprehensive understanding of the 3D structure in images. For example, humans can answer the questions shown in Fig. 4.1, which ask about the object parts, their 3D poses, and occlusions. However, current VQA models, which often rely on 2D bounding boxes to encode a visual scene [50, 76, 57] struggle to answer such questions reliably (as can be seen from our experiments). We hypothesize this is caused by the lack of understanding of the 3D structure images.

In this chapter, we introduce the task of 3D-aware VQA, where answering the questions requires compositional reasoning over the 3D structure of the

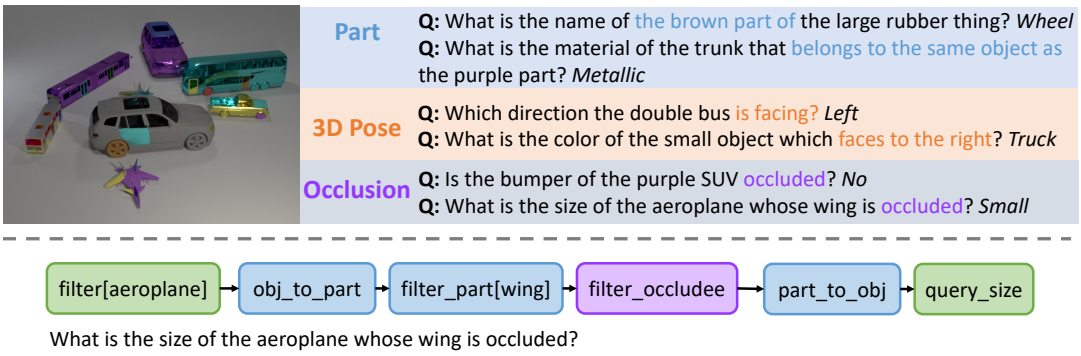


Figure 4.1: Examples from Super-CLEVR-3D. We introduce the task of 3D-aware VQA, which requires 3D understanding of the image, including the parts, 3D poses, and occlusions.

visual scenes. More specifically, we focus on challenging questions that require multi-step reasoning about the object-part hierarchy, the 3D poses of the objects, and the occlusion relationships between objects or parts.

We address the challenging 3D-aware VQA task from both the dataset and the model perspective. From the dataset perspective, we introduce Super-CLEVR-3D, which extends the Super-CLEVR dataset [105] with 3D-aware questions. Given the visual scenes from Super-CLEVR that contain randomly placed vehicles of various categories, we define a set of 3D-aware reasoning operations and automatically generate 3D questions based on these operations. Fig. 4.1 shows examples of the images, questions and the underlying 3D operations for the questions. From the model perspective, we introduce PO3D-VQA, a VQA model that marries two powerful ideas: probabilistic neural symbolic program execution for reasoning and a deep neural network with 3D generative representations of objects for robust visual scene parsing. Our model first recovers a 3D scene representation from the image and a program from the question, and subsequently executes the program on the

3D scene representation to obtain an answer using a probabilistic reasoning process that takes into account the confidence of predictions from the neural network. We refer to our system as PO3D-VQA, which stands for **P**arts, Poses, and **O**cclusions in **3D** **V**isual **Q**uestion **A**nswering.

On Super-CLEVR-3D, we experiment with existing representative models, their variants, and our model PO3D-VQA. The results show that our model outperforms existing methods significantly, leading to an improvement in accuracy of more than 11%, which shows the advantage of the generative 3D scene parser and the probabilistic neural symbolic reasoning process. Moreover, further analysis on questions with different difficulty levels reveals that the improvements of our model are even greater on harder questions with heavy occlusions and small part sizes. Our results indicate that a reliable 3D understanding, together with the modular reasoning procedure, produces a desirable 3D-aware VQA model.

In summary, our contributions are as follows. (1) We introduce the challenging task of 3D-aware VQA and propose the Super-CLEVR-3D dataset, where 3D visual understanding about parts, 3D poses, and occlusions are required. (2) We propose a 3D-aware neural modular model PO3D-VQA that conducts probabilistic reasoning in a step-wise modular procedure based on robust 3D scene parsing. (3) With experiments, we show that 3D-aware knowledge and modular reasoning are crucial for 3D-aware VQA, and suggest future VQA methods take 3D understanding into account.

4.2 Related Work

Visual Question Answering. Rapid progress has been made in VQA [14] in both the datasets and the models. To solve the challenging VQA datasets [58, 77, 78, 80] with real images, multiple models are developed including two-stream feature fusion [50, 18, 23, 72, 73, 56, 74] or transformer-based pretraining [2, 75, 5, 76, 57]. However, the real datasets are shown to suffer from spurious correlations and biases [7, 81, 8, 9, 58, 11, 12]. Alternatively, synthetic datasets like CLEVR [70] and Super-CLEVR [105], are developed to study the compositional reasoning ability of VQA systems, which are also extended to study other vision-and-language tasks [82, 83, 84, 85, 36, 86, 87]. The synthetic datasets promote the development of neural modular methods [106, 55, 54, 107], where the reasoning is done in a modular step-by-step manner. It is shown that the modular methods have nice properties including interpretability, data efficiency [55, 54], better robustness [105] and strong performance on synthetic images [55]. However, most existing methods rely on region features [50, 76] extracted using 2D object detectors [108] for image encoding, which is not 3D-aware. We follow the works on the synthetic dataset and enhance the modular methods with 3D understanding.

VQA in 3D. Multiple existing works study VQA under the 3D setting, such as SimVQA [101], SQA3D [109], 3DMV-VQA [110], CLEVR-3D [111], ScanQA [112], 3DQA [112], and EmbodiedQA [113], which focus on question answering on the 3D visual scenes like real 3D scans [109, 111, 114, 112], simulated 3D environments [25, 113], or multi-view images [110]. PTR [87] is a synthetic VQA dataset that requires part-based reasoning about physics,

analogy and geometry. Our setting differs from these works because we focus on 3D in the *questions* instead of 3D in the *visual scenes*, since our 3D-aware questions explicitly query the 3D information that can be inferred from the 2D input images.

3D scene understanding. One popular approach for scene understanding is to use the CLIP features pretrained on large-scale text-image pairs and segment the 2D scene into semantic regions [115, 116]. However, these methods lack a 3D understanding of the scene and cannot be used to answer 3D-related questions. Another approach is to adopt category-level 6D pose estimation methods that can locate objects in the image and estimate their 3D formulations. Previous approaches include classification-based methods that extend a Faster R-CNN model for 6D pose estimation [117, 118] and compositional models that predicts 6D poses with analysis-by-synthesis [118]. We also notice the huge progress of 3D vision language foundation models, which excel in multiple 3D vision-language understanding tasks [110, 119, 120]. Still, we focus on the reasoning with compositional reasoning that brings more interpretability and robustness [105].

4.3 Super-CLEVR-3D Dataset

To study 3D-aware VQA, we propose the Super-CLEVR-3D dataset, which contains questions explicitly asking about the 3D object configurations of the image. The images are rendered using scenes from the Super-CLEVR dataset [105], which is a VQA dataset containing synthetic scenes of randomly placed vehicles from 5 categories (car, plane, bicycle, motorbike, bus) with various of

sub-types (*e.g*. different types of cars) and attributes (color, material, size). The questions are generated by instantiating the question templates based on the image scenes, using a pipeline similar to Super-CLEVR. In Super-CLEVR-3D, three types of 3D-aware questions are introduced: part questions, 3D pose questions, and occlusion questions. In the following, we will describe these three types of questions, and show the new operations we introduced for our 3D-aware questions about object parts, 3D poses, and occlusions. Examples of the dataset are shown in Fig. 4.1.

Part questions. While in the original Super-CLEVR dataset refers to objects using their holistic names or attributes, objects are complex and have hierarchical parts, as studied in recent works [71, 121, 87]. Therefore, we introduce part-based questions, which use parts to identify objects (*e.g*. "which vehicle has red door") or query about object parts (*e.g*. "what color is the door of the car"). To enable the generation of part-based questions, we introduce two new operations into the reasoning programs: `part_to_object(·)`, which find the objects containing the given part, and `object_to_part(·)`, which select all the parts of the given object. We also modify some existing operations (*i.e*. `filter`, `query` and `unique`), enabling them to operate on both object-level and part-level. With those reasoning operations, we collect 9 part-based templates and instantiate them with the image scene graph to generate questions.

3D pose questions. Super-CLEVR-3D asks questions about the 3D poses of objects (*e.g*. "which direction is the car facing in"), or the pair-wise pose relationships between objects (*e.g*. "which object has vertical direction with the red car"). The pose for an individual object (*e.g*. "facing left") can be processed

in a similar way as attributes like colors, so we extend the existing attribute-related operations like `filter` and `query` to have them include pose as well. For pair-wise pose relationship between objects, we add three operations, *i.e.* `same_pose`, `opposite_pose` and `vertical_pose`, to deal with the three types of pose relationships between objects. For example, `opposite_pose(·)` returns the objects that are in the opposite pose direction with the given object. 17 templates are collected to generate 3D pose questions.

Occlusion questions. Occlusion questions ask about the occlusion between entities (*i.e.* objects or parts). Similar to 3D poses, occlusion can also be regarded as either an attributes for an entity (*e.g.* "which object is occluded"), or as a relationship between entities (*e.g.* "which object occludes the car door"). We extend the attribute-related operations, and introduce new operations to handle the pair-wise occlusion relationships: `filter_occludee` which filters the entities that are being occluded, `relate_occluding` which finds the entities that are occluded by the given entity, and `relate_occluded` which finds the entities that are occluding the given entity. Using these operations, 35 templates are collected to generate the occlusion questions.

4.4 Method

In this section, we introduce PO3D-VQA, which is a parse-then-execute modular model for 3D-aware VQA. The overview of our system is shown in Fig. 4.2. We first parse the image into a scene graph representation that is aware of 3D information like object parts, 3D poses and occlusion relations, then we parse the question into a reasoning program and execute the program on

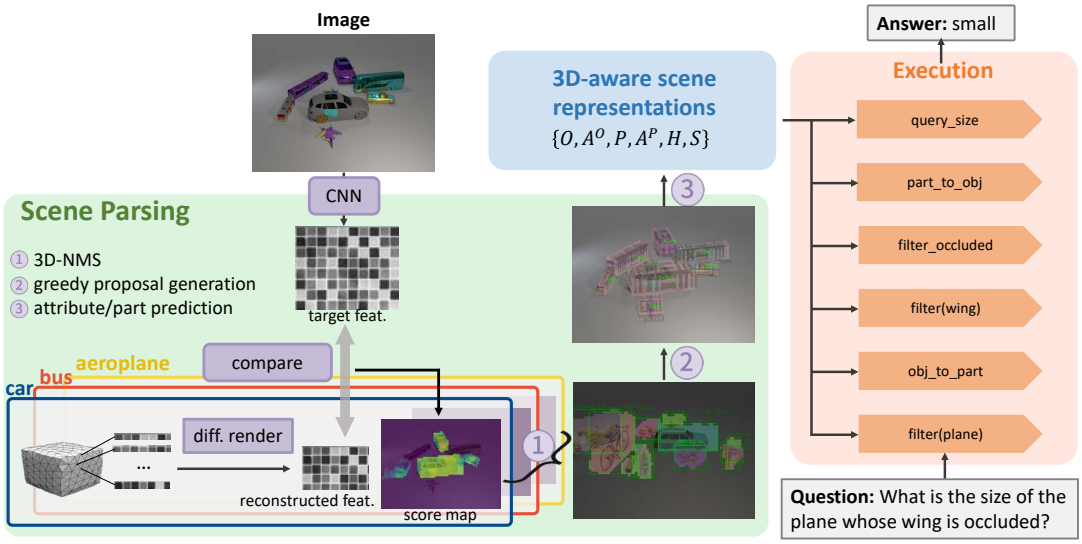


Figure 4.2: An overview of our model PO3D-VQA. The image is parsed into 3D-aware scene representations (blue box) using our proposed scene parser based on the idea of render-and-compare (green box). The question is parsed into a program composed of reasoning operations (orange box). Then the operations are executed on the 3D-aware scene representations to predict the answer.

the derived scene representations in a probabilistic manner. In Sec. 4.4.1, we define the scene representation required; in Sec. 4.4.2, we describe how we parse the image into the scene representation based on a multi-class 6D pose estimation model with non-trivial extensions; in Sec. 4.4.3, we describe how the question is executed on the derived scene representation to predict the answer.

4.4.1 3D-aware scene representation

Given an input image I, we parse it into a 3D-aware scene representation R that contains the **objects** (O) with attributes (A^o), the **parts** (P) with attributes (A^p), the **hierarchical relationships** between objects and parts (H), and the **occlusion relationships** between them (S). The attributes include the 3D

poses and locations of objects or parts, as well as their colors, materials, and sizes. The scene representation $R = \{O, P, A^o, A^p, H, S\}$ is comprehensive and therefore we can directly execute the symbolic reasoning module on this representation without taking into account the image any further.

In more detail, **objects** are represented as a matrix $O \in \mathbb{R}^{n \times N_{obj}}$ containing the probability scores of each object being a certain instance, where n is the number of objects in the given image and N_{obj} is the number of all possible object categories in the dataset (*i.e.* vocabulary size of the objects). Similarly, **parts** are represented as $P \in \mathbb{R}^{p \times N_{prt}}$, where p is the number of parts in the image and N_{prt} is the vocabulary size of the object parts. The **object-part hierarchy** is represented by a binary matrix $H \in \mathbb{R}^{n \times p}$, where $H_{ij} = 1$ if the object i contains the part j or $H_{ij} = 0$ otherwise. The attributes $A^o \in \mathbb{R}^{n \times N_{att}}$ and $A^p \in \mathbb{R}^{p \times N_{att}}$ containing probability scores of each object or part having a certain attribute or the value of bounding box. Here N_{att} is the number of attributes including the 3D poses, location coordinates, colors, materials and sizes. **Occlusion relationships** are represented by $S \in \mathbb{R}^{(n+p) \times n}$, where each element S_{ij} represents the score of object (or part) i being occluded by object j.

4.4.2 Multi-class 6D Scene Parsing

While most existing VQA methods [50, 76] encode the image using pretrained object detectors like Faster-RCNN [108], we build our 6D-aware scene parser in a different way, based on the idea of analysis-by-synthesis through inverse rendering [122] which has the following advantages: first, the model prediction is more robust [122] as the render-and-compare process can naturally

integrate a robust reconstruction loss to avoid distortion through occlusion; second, while the object parts are usually very challenging for Faster-RCNN to detect due to their small size, they can be much easier located using the 3D object shape, by first finding the object and estimating its 3D pose, and subsequently locating the parts using the 3D object shape (as shown in our experimental evaluation).

However, we observe two open challenges for applying existing 6D pose estimators that follow a render-and-compare approach [118, 122]: (a) these pose estimators assume that the object class is known, but in Super-CLEVR-3D the scene parser must learn to estimate the object class jointly with the pose; and (b) the scenes in Super-CLEVR-3D are very dense, containing multiple close-by objects that occlude each other. In order to address these two challenges, we introduce several improvements over [118] that enable it to be integrated into a 3D-aware VQA model.

In the following, we first describe neural meshes [122, 118], which were proposed in prior work for pose estimation of *single objects* following an analysis-by-synthesis approach. Subsequently, we extend this method to complex scenes with densely located and possibly occluded objects to obtain a coherent scene representation, including object parts and attributes.

Preliminaries. Our work builds on and significantly extends Neural Meshes [118] that were introduced for 6D pose estimation through inverse rendering. The task is to jointly estimate the 6D pose (2D location, distance to the camera and 3D pose) of objects in an image. An object category is represented with a category-level mesh [122] $M_y = \{v_n \in \mathbb{R}^3\}_{n=1}^{N}$ and a neural

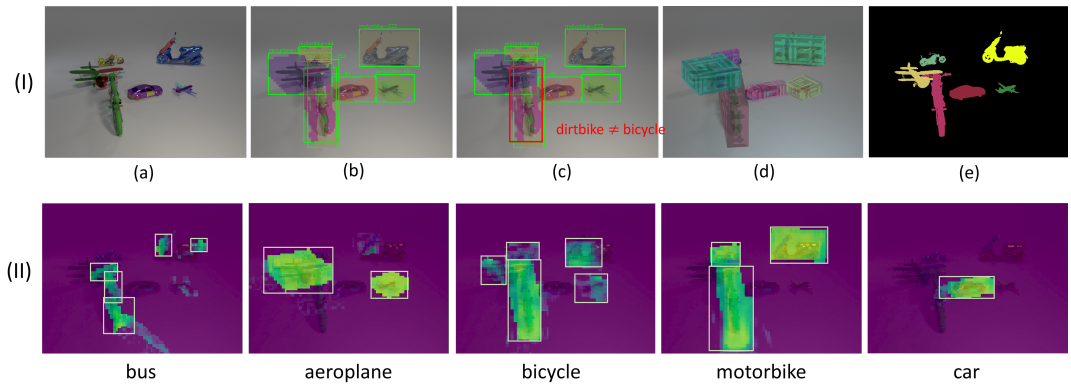


Figure 4.3: Visualization of intermediate steps in our scene parser. Given an image (a), per-category feature activation maps (shown in II) are computed through render-and-compare. Then the category-wise competition (3D-NMS) is performed (results shown in b) and a post-filtering step is taken to remove mis-detected objects (c). Based on the pose estimation results (d), we project the 3D object mesh back onto the image to locate parts and occlusions(e).

texture $T_y \in \mathbb{R}^{N \times c}$ on the surface of the mesh M_y, where c is the dimension of the feature and y is the object category. Given the object 3D pose in camera view α, we can render the neural mesh model $O_y = \{M_y, T_y\}$ into a feature map with soft rasterization [123]: $F_y(\alpha) = \mathfrak{R}(O_y, \alpha)$. Following prior work in pose estimation [122] we formulate the render-and-compare process as an optimization of the likelihood model:

$$p(F \mid O_y, \alpha_y, B) = \prod_{i \in \mathcal{FG}} p(f_i \mid O_y, \alpha_y) \prod_{i \in \mathcal{BG}} p(f_i' \mid B) \tag{4.1}$$

where $\mathcal{FG}$ and $\mathcal{BG}$ are the set of foreground and background locations on the 2D feature map and f_i is the feature vector of F at location i. Here the foreground and background likelihoods are modeled as Gaussian distributions.

To train the feature extractor Φ, the neural texture $\{T_y\}$ and the background model B jointly, we utilize the EM-type learning strategy as originally

introduced for keypoint detection in CoKe[124]. Specifically, the feature extractor is trained using stochastic gradient descent while the parameters of the generative model $\{T_y\}$ and B are trained using momentum update after every gradient step in the feature extractor, which was found to stabilize training convergence.

At inference time, the object poses α can be inferred by minimizing the negative log-likelihood w.r.t. the 3D pose α using gradient descent [118].

Multi-object competition with 3D-NMS. We extend Neural Meshes to predict the 6D object pose and class label in complex multi-object scenes. In particular, we introduce 3D-Non-Maximum-Suppression (3D-NMS) into the maximum likelihood inference process. This introduces a competition between Neural Meshes of different categories in explaining the feature map. In contrast to classical 2D-NMS, our 3D-NMS also takes into account the distance of each object to the camera and hence naturally enables reasoning about occlusions of objects in the scene.

We denote the 6D pose as $\gamma = \{x, l\}$, where $x = \{\alpha, \beta\}$ represents the 3D object pose α and object distance to the camera β, and l is the 2D object location in the feature map. We first detect the 6D poses of each object category independently and apply 2D-NMS such that for each 2D location l' in a neighborhood defined by radius r, the predicted 6D pose $\{x, l\}$ yields the largest activation:

$$\max_x \; p(F \mid x, l) \;\; s.t. \;\; p(F \mid x, l) > p(F \mid x, l'), \;\; \forall l' \in \{l' \mid 0 < |l' - l| < r\} \tag{4.2}$$

We enable multi-category 6D pose estimation by extending this formulation to

a 3D non-maximum suppression (3D-NMS). Using $\mathcal{Y}$ to represent the set of all object categories, we model the category label y from a generative perspective:

$$\max_{x} \; p(F \mid x, l, y) \;\; s.t. \;\; p(F \mid x, l, y) > p(F \mid x, l', y), \;\; \forall l' \in \{l' \mid 0 < |l' - l| < r\} \tag{4.3}$$

$$and \;\; p(F \mid x, l, y) > p(F \mid x, l, y'), \;\; \forall y' \neq y \in \mathcal{Y} \tag{4.4}$$

Dense scene parsing with greedy proposal generation. Typically, object detection in complex scenes requires well chosen thresholds and detection hyperparameters. Our render-and-compare approach enables us to avoid tedious hyperparameter tuning by adopting a greedy approach to maximize the model likelihood (Eq. (4.1)) using a greedy proposal strategy. In particular, we optimize the likelihood greedily by starting from the object proposal that explains away the most parts of the image with highest likelihood, and subsequently update the likelihood of the overlapping proposals taking into account, that at every pixel in the feature map only one object can be visible [125]. Formally, given a list of objects proposals $\{o_i = (O_{y,i}, \alpha_{y,i})\}_{i=1}^{k}$ (with predicted category label y and 6D pose α), we first order the object proposals based on their likelihood score $s = p(F|o_i, B)$ such that $s_i \leq s_j$ for $i < j$. Based on the ordering, we greedily update the 6D pose α_j and the corresponding proposal likelihood for object o_j by masking out the foreground regions of previous objects o_i with $1 \leq i \leq j-1$. In this way, we can largely avoid missing close-by objects or duplicated detection.

Part and attribute prediction. Given the predicted location and pose of each object, we project the object mesh back onto the image to get the locations

for each part. To predict the attributes for the objects and parts, we crop the region containing the object or part from the RGB image, and train an additional CNN classifier using the cropped patches to predict the attributes (color, size, material) and the fine-grained classes (*i.e.* different sub-types of cars) of each patch using a cross-entropy loss. The reason why this additional CNN classifier is needed instead of re-using the features from the 6D pose estimator is that the pose estimation features are learned to be invariant to scale and texture changes, which makes it unsuitable for attribute prediction.

Post-filtering. Finally, we post-process the located objects using the fine-grained CNN classifier. We compare the category labels predicted by the 6D pose estimator with the ones predicted by the CNN classifier, and remove the objects for which these two predictions do not agree. This post-filtering step helps with the duplicated detections that cannot be fully resolved with the 3D-NMS.

Summary. Fig. 4.2 provides an overview of our scene parser and Fig. 4.3 visualize the intermediate results. With the idea of render-and-compare (shown in the green box of Fig. 4.2), the model first computes an activation map for each possible object category (Fig. 4.3II). Next, to infer the category for each object, the category-wise competition 3D-NMS is performed (Fig. 4.3b) and a post-filtering step is taken to remove mis-detected objects (Fig. 4.3c). Fig. 4.3d shows the 6D pose estimation results. To predict parts, we project the 3D object mesh back onto the image to locate parts based on projected objects (Fig. 4.3e). In this way, the input image can be parsed into a 3D-aware representation, which is ready for the question reasoning with program execution.

4.4.3 Program execution

After the 3D-aware scene representations are predicted for the given image, the question is parsed into a reasoning program, which is then executed on the scene representation to predict the answer. The question parsing follows previous work [55], where a LSTM sequence-to-sequence model is trained to parse the question into its corresponding program. Like P-NSVQA [105], each operation in the program is executed on the scene representation in a probabilistic way. In the following, we describe the execution of the new operations we introduced.

The part-related operators are implemented by querying the object-part hierarchy matrix H, so that the object containing a given part (`part_to_object`) and the parts belonging to the given object (`object_to_part`) can be determined. The pose-related operators are based on the estimated 3D pose in the object attributes A^o. For the `filter` and `query` operations regarding pose, the 3D poses are quantified into four direction (left, right, front, back). For the pair-wise pose relationships, the azimuth angle between two objects is used to determine the same/opposite/vertical directions. The occlusion-related operations are implemented by querying the occlusion matrix S. Based on the occlusion scores S_{ij} representing whether entity i being occluded by entity j, we can compute the score of one entity being occluded $\sum_j S_{ij}$ (`filter_occludee`), find the entities that occlude a given entity (`relate_occluded`), or find the entities that are occluded by a given entity (`relate_occluded`).

4.5 Experiments

4.5.1 Evaluated methods

We compare our model with three representative VQA models: FiLM [56], mDETR [57], and PNSVQA [105]. Additionally, we introduce a variant of PNSVQA, PNSVQA+Projection, to analyze the benefit of our generative 6D pose estimation approach.

FiLM [56] *Feature-wise Linear Modulation* is a representative two-stream feature fusion method. The FiLM model merges the question features extracted with GRU [103] and image features extracted with CNN and predicts answers based on the merged features.

mDETR [57] mDETR is a pretrained text-guided object detector based on transformers. The model is pretrained with 1.3M image and text pairs and shows strong performance when finetuned on downstream tasks like referring expression understanding or VQA.

PNSVQA [105] PNSVQA is a SoTA neural symbolic VQA model. It parses the scene using MaskRCNN [104] and an attribute extraction network, then executes the reasoning program on the parsed visual scenes with taking into account the uncertainty of the scene parser. To extend PNSVQA to the 3D questions in Super-CLEVR-3D, we add a regression head in the attribute extraction network to predict the 3D posefor each object; parts are detected in a similar way as objects by predicting 2D bounding boxes; the part-object associations and occlusions are computed using intersection-over-union: a part belongs to an intersected object if the part label matches the object label,

otherwise it is occluded by this object.

PNSVQA+Projection Similar with NSVQA, this model predicts the 6D poses, categories and attributes using MaskRCNN and the attribute extraction network. The difference is that the parts and occlusions are predicted by projecting the 3D object models onto the image using the predicted 6D pose and category (same with how we find parts and occlusions in our model). This model helps us ablate the influence of the two components in our model, *i.e*. 6D pose prediction by render-and-compare, and part/occlusion detection with mesh projection.

4.5.2 Experiment setup

Dataset. Our Super-CLEVR-3D dataset shares the same visual scenes with Super-CLEVR dataset. We re-render the images with more annotations recorded (camera parameters, parts annotations, occlusion maps). The dataset splits follow the Super-CLEVR dataset, where we have 20k images for training, 5k for validation, and 5k for testing. For question generation, we create 9 templates for part questions, 17 templates for pose questions, 35 templates for occlusion questions (with and without parts). For each of the three types, 8 to 10 questions are generated for each image by randomly sampling the templates. We ensure that the questions are not ill-posed and cannot be answered by taking shortcuts, *i.e*. the questions contain no redundant reasoning steps, following the no-redundancy setting in [105]. More details including the list of question templates can be found in the Appendix.

Table 4.1: Model accuracies on the Super-CLEVR-3D testing split, reported for each question type, *i.e.* questions about parts, 3D poses, occlusions between objects, occlusions between objects and parts.

	Mean	Part	Pose	Occ.	Part+Occ.
FiLM [56]	50.53	38.24	67.82	51.41	44.66
mDETR [57]	55.72	41.52	71.76	64.99	50.47
PNSVQA [105]	64.39	50.61	**87.78**	65.80	53.35
PNSVQA+Projection	68.15	56.30	86.70	70.70	58.90
PO3D-VQA (Ours)	**75.64**	**71.85**	86.40	**76.90**	**67.40**

Implementation details. We train the 6D pose estimator and CNN attribute classifier separately. We train the 6D pose estimator (including the contrastive feature backbone and the nerual mesh models for each of the 5 classes) for 15k iterations with batch size 15, which takes around 2 hours on NVIDIA RTX A5000 for each class. The attribute classifier, which is a ResNet50, is shared for objects and parts. It is trained for 100 epochs with batch size 64. During inference, it takes 22s for 6D pose estimation and 10s for object mesh projection for all the objects in one image. During inference of the 6D pose estimator, we assume the theta is 0. During 3D NMS filtering, we choose the radius r as 2, and we also filter the object proposals with a threshold of 15 on the score map.

4.5.3 Quantitative Results

We trained our model and baselines on Super-CLEVR-3D's training split, reporting answer accuracies on the test split in Tab. 4.1. Accuracies for each question type are detailed separately.

Comparison with baselines. First, among all the baseline methods, the

neural symbolic method PNSVQA performs the best (64.4% accuracy), outperforming the end-to-end methods mDETR and FiLM by a large margin ($> 8\%$). This shows the advantage of the step-wise modular reasoning procedure, which agrees with the findings in prior works that the modular methods excel on the simulated benchmarks that require long-trace reasoning. Second, our model achieves 75.6% average accuracy, which significantly outperforms all the evaluated models. Especially, comparing our PO3D-VQA with its 2D counterpart NSVQA, we see that the injection of 3D knowledge brings a large performance boost of 11%, suggesting the importance of the 3D understanding.

Comparison with PNSVQA variants. By analyzing the results of PNSVQA variants (*PNSVQA*, *PNSVQA+Projection*, and our *PO3D-VQA*), we show (a) the benefit of estimating object 3D poses using our analysis-by-synthesis method over regression and (b) the benefit of object-part structure knowledge. First, by detecting part using 3D model projection, *PNSVQA+Projection* improves the *PNSVQA* results by 4%, which indicates that locating parts based on objects using the object-part structure knowledge is beneficial. Second, by estimating object 6D poses with our generative render-and-compare method, our *PO3D-VQA* outperforms *PNSVQA+Projection* by 7% (from 68.2% to 75.6%), showing the advantage of our render-and-compare model. Moreover, looking at the per-type results, we find that the improvement of our PO3D-VQA is most significant on the part-related questions (21% improvement over PNSVQA) and part-with-occlusion questions (14%), while the accuracy on pose-related questions does not improve. The reason is that part

and occlusion predictions require precise pose predictions for accurate mesh projection, while the pose questions only require a rough pose to determine the facing direction.

4.5.4 Analysis and discussions

To further analyze the advantage of PO3D-VQA over other PNSVQA variants, we compare the models on questions of different difficulty levels. It is shown that the benefit our model is the most significant on hard questions. In Fig. 4.4, we plot the relative accuracy drop [1] of each model on questions with different occlusion ratios and questions with different part sizes.

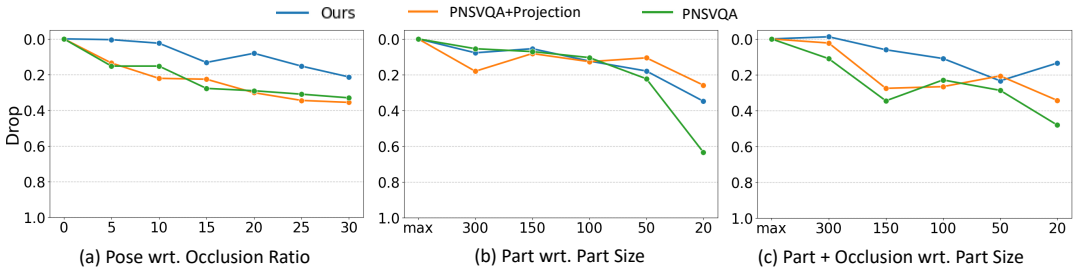


Figure 4.4: Analysis on questions of different difficulty levels. The plots show the relative accuracy drop of models, on pose questions w.r.t. different occlusion ratios (a), on part questions w.r.t. different part sizes (b), and on part+occlusion questions w.r.t. different part sizes (c).

Questions with different occlusion ratios. We sort pose-related questions into different sub-groups based on their *occlusion ratios* and evaluate the models on each of the sub-groups. The *occlusion ratio r* of a question is the *minimum* of occlusion ratios for all the objects in its reasoning trace. We choose r from 0% to 30%, in increment of 5%. The results are shown is Fig. 4.4 (a).

[1] Relative accuracy drop means the ratio of absolute accuracy drop and the original accuracy. For example, if a model's accuracy drops from 50% to 45%, its relative accuracy drop is 10%.

Our PO3D-VQA is much more robust to occlusions compared to the other two methods: while the performances of all the three models decrease as the occlusion ratio increases, the relative drop of ours is much smaller than others. The results show that our render-and-compare scene parser is more robust to heavy occlusions compared with the discriminative methods.

Questions with different part sizes. Questions about small parts are harder than the ones about larger parts. We sort the questions into different part size intervals (s, t), where the *largest* part that the question refers to has an area (number of pixels occupied) larger than s and smaller than t. We compare the models on the part questions and the part+occlusion questions with different part sizes in Fig. 4.4 (b) and (c). In (b), the accuracy drop of PO3D-VQA is smaller than PNSVQA+Projection and PNSVQA when parts get smaller. In (c), PNSVQA+Projection is slightly better than our model and they are both better than the original PNSVQA.

In summary, by sorting questions into different difficulty levels based on occlusion ratios and part sizes, we show the advantage of our PO3D-VQA on harder questions, indicating that our model is robust to occlusions and small part sizes.

Qualitative results. Fig. 4.5 shows examples of predictions for our model and PNSVQA variants. In (a), the question asks about occlusion, but with a slight error in the pose prediction, PNSVQA+Projection misses the occluded bus and predicts the wrong answer, while our model is correct with **accurate pose**. In (b), the question refers to the heavily occluded minivan that is difficult to detect, but our model gets the correct prediction thanks to its **robustness to**

occlusions.

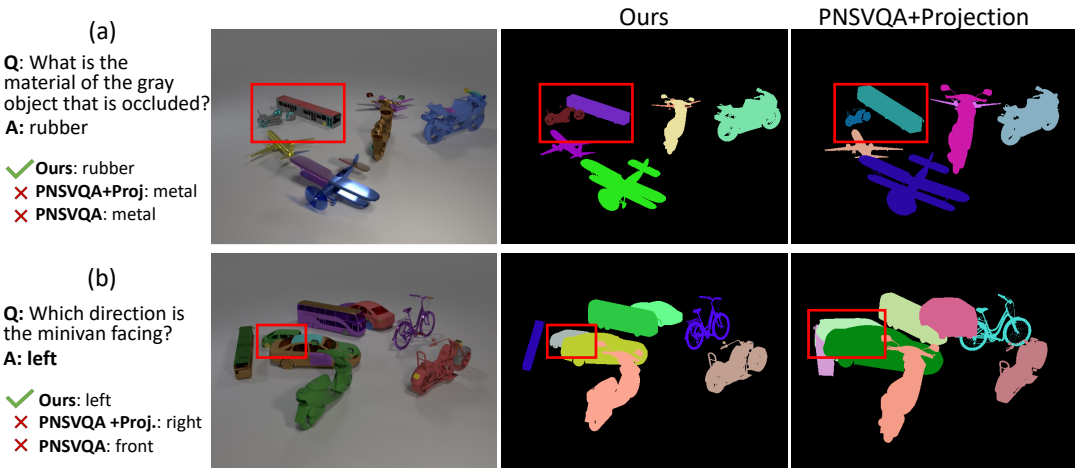


Figure 4.5: Examples of models' predictions. Our model (a) predicts the object pose accurately and (b) is robust to heavy occlusions. Red boxes are for visualization only.

Limitations and failure cases. Due to the difficulties of collecting real images with compositional scenes and 3D annotations, our work is currently limited by its synthetic nature. For PO3D-VQA, it sometimes fails to detect multiple objects if they are from the same category and heavily overlap (see Appendix D for more visualizations). 3D NMS can effectively improve the dense scene parsing results when objects are from different categories, but conceptually it is limited when objects are from the same category. However, 6D pose estimation in dense scenes is a challenging problem, whereas many current works on 6D pose estimation are still focusing on simple scenes with single objects [118, 126, 127].

4.6 Further Discussion

In this section, we discuss two meaningful extensions of our work: the incorporation of z-direction questions and the application of our model to real-world images.

Z-direction questions. While the proposed Super-CLEVR-3D dataset has been designed with 3D-aware questions, all objects within it are placed on the same surface. Introducing variability in the z direction can further enrich our dataset with more comprehensive 3D spatial relationships.

We consider the scenario where aeroplane category, is in different elevations, introducing the z dimension into the spatial relationships (see Fig. 4.6). This allowed us to formulate questions that probe the model's understanding of height relationships and depth perception. We create a subset containing 100 images and 379 questions and test our PO3D-VQA model directly on it without retraining the 6D parser. On this dataset, our PO3D model achieves 90.33% accuracy on height relationship questions and 78.89% on depth-related questions, suggesting that our model can successfully handle questions about height. As the baseline models only use the bounding box to determine the spatial relationship between objects, they are not able to determine the height relationships.

Extension to real-world images While our PO3D-VQA model has demonstrated impressive performance on the synthetic Super-CLEVR-3D dataset, an essential research direction is extending it to real images or other 3D VQA datasets (such as GQA and FE-3DGQA). However, it's not trivial to truly evaluate it on these real-world problems, and a primary challenge is the lack

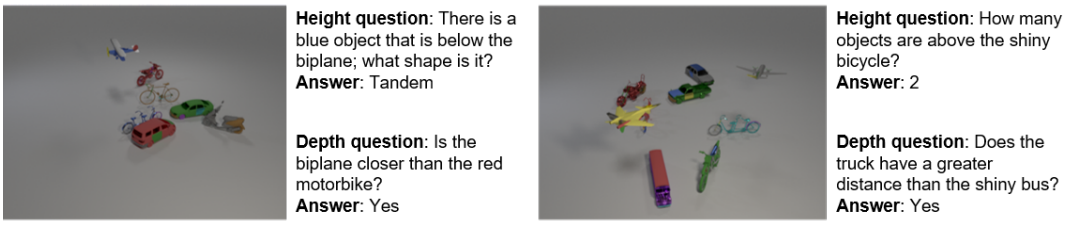


Figure 4.6: Example images and questions of objects with different elevations.

of 3D annotations and the highly articulated categories (like the human body) in these datasets.

However, we show that our PO3D-VQA model can, in principle, work on realistic images. We generate several realistic image samples manually using the vehicle objects (e.g. car, bus, bicycle) from ImageNet with 3D annotation (see Fig. 4.7) and real-image background. In this experiment, the pose estimator is trained on the PASCAL3D+ dataset, and is used to predict the poses of objects from the image before pasting, as shown in (b). The attribute (color) prediction module is trained on Super-CLEVR-3D and the object shapes are predicted by a ResNet trained on ImageNet. Our model can correctly predict answers to questions about the object pose, parts, and occlusions, e.g. "Which object is occluded by the mountain bike".

Figure 4.7: Examples of results on realistic images. Given a realistic image (a1, a2), our model can successfully estimate the 6D poses of objects (b1, b2) and answer the 3D-aware questions (c1, c2).

4.7 Conclusion

In this chapter, we study the task of 3D-aware VQA. We propose the Super-CLEVR-3D dataset containing questions explicitly querying 3D understanding including object parts, 3D poses, and occlusions. To address the task, a 3D-aware neural symbolic model PO3D-VQA is proposed, which enhances the probabilistic symbolic model with a robust 3D scene parser based on analysis-by-synthesis. With the merits of accurate 3D scene parsing and symbolic execution, our model outperforms existing methods by a large margin. Further analysis shows that the improvements are even larger on harder questions. With the dataset, the model, and the experiments, we highlight the benefit of symbolic execution and the importance of 3D understanding for 3D-aware VQA.

Part II

Compositional Models for Visual Question Answering

Chapter 5

Calibrating Concepts and Operations: Towards Symbolic Reasoning on Real Images

While neural symbolic methods demonstrate impressive performance in visual question answering on synthetic images, their performance suffers on real images. We identify that the long-tail distribution of visual concepts and unequal importance of reasoning steps in real data are the two key obstacles that limit the models' real-world potentials. To address these challenges, we propose a new paradigm, **Calibrating Concepts and Operations (CCO)**, which enables neural symbolic models to capture underlying data characteristics and to reason with hierarchical importance. Specifically, we introduce an executor with learnable concept embedding magnitudes for handling distribution imbalance, and an operation calibrator for highlighting important operations and suppressing redundant ones.

Our experiments show CCO substantially boosts the performance of neural symbolic methods on real images. By evaluating models on the real world

dataset GQA, CCO helps the neural symbolic method NSCL outperforms its vanilla counterpart by 9.1% (from 47.0% to 56.1%); this result also largely reduces the performance gap between symbolic and non-symbolic methods. Additionally, we create a perturbed test set for better understanding and analyzing model performance on real images. Code is available at https://github.com/Lizw14/CaliCO.git.

5.1 Introduction

Visual question answering aims to develop a model that can answer open-ended questions from images. Currently, end-to-end methods, which directly make predictions over dense visual and textual features [17, 23], represent the most effective class of models for VQA. Nonetheless, such methods have been criticized for exploiting shortcuts (*e.g.*, statistical dataset bias [9, 10], question prior [128] or isolated text and image elements [27]) to answer questions; these shortcuts often make them unable to generalize well on out-of-domain data.

In contrast, neural symbolic methods [106, 129, 130, 131] are equipped with strong reasoning ability, enabling them to answer multi-hop and complex questions in a compositional and transparent manner—they first parse each question into a program with a series of reasoning steps, and then compose neural modules on the fly to execute the program on the image. While symbolic methods achieve nearly perfect performance on synthetic dataset, they perform poorly on real-world datasets. For instance, neural symbolic concept learner (NSCL) [131] achieves 98.9% accuracy on the synthetic CLEVR dataset [70], but only 47.0% accuracy on the real-world GQA dataset [59]. Note

the original NSCL cannot be directly applied to GQA; this 47.0% accuracy is obtained from our own re-implementation, where minimal but necessary modifications are made (*e.g.*, adding in *same* and *common* modules), for making models runnable on GQA.

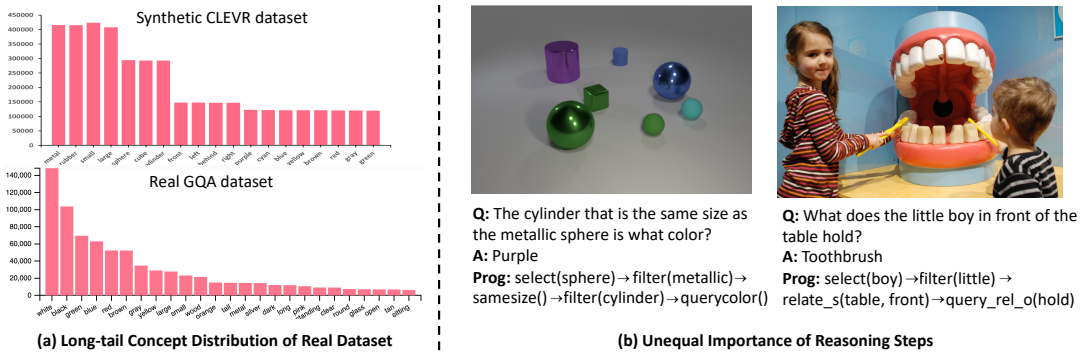


Figure 5.1: Statistics and examples from the synthetic CLEVR dataset and the real GQA dataset. Compared to the synthetic dataset, VQA on real data needs to deal with a long-tail concept distribution and the uneven importance of reasoning steps.

As summarized in Figure 5.1, we note there are two major differences between synthetic datasets and real-world datasets. First, while visual concepts are well-balanced in the synthetic datasets, they follow a long-tail distribution in real-world datasets. For example, as shown in Figure 5.1(a), in GQA, common concepts like “man”, “window”, “black”, “white” are far more frequent than uncommon ones like “pink” and “eraser”, in both questions and answers. Second, unlike in synthetic data, the reasoning steps on real data have varying importance, mainly because of redundancy/over-specification in question description. For example, as shown in Figure 5.1(b), in the question "What is the little boy doing?", the noun (*i.e.*, boy) itself is enough to select the person being asked about while the adjective (*i.e.*, little) only serves as a nuisance factor.

We identify that this mismatch of dataset characteristics is the main obstacle for adapting neural symbolic methods from synthetic datasets to real-world datasets. More concretely, we find that the original architecture designs of neural symbolic methods (which were designed/verified mainly on synthetic datasets) are no longer suitable for the real-world setting. For examples, as shown in Section 5.3, even simple operations like removing the normalization on concept embeddings or manually assigning larger weights to less discriminative modules are effective to improve the performance of neural symbolic methods on real images.

To better cope with real images, we propose *Calibrating Concepts and Operations (CCO)*, which enables neural symbolic methods to explicitly learn weights for concept embedding and reason with contextual module importance. Specifically, CCO learns different concept embedding magnitudes for each execution module, and learns an operation weight predictor to contextually predict weights for each operation in the reasoning program. In this way, the model will be able to handle unbalanced concept distributions and to reason with varying operation importance.

Our empirical results show that CCO substantially boosts the applicability of neural symbolic methods on real images. For example, on the real-world GQA dataset, CCO outperforms the baseline NSCL by a large margin of 9.1% (from 47.0% to 56.1%). Moreover, the proposed CCO method largely reduces the performance gap between the symbolic method and the state-of-the-art non-symbolic methods [2, 132] on real-world GQA dataset.

Additionally, based on the proposed operation weight calibrator, we create a perturbed test set by progressively removing the operations with low weights from testing questions. Our purpose is to verify whether the learned operation weights are able to highlight important operations and suppress redundant ones, and simultaneously to access the robustness of different models regarding this operation information erasing. Our analysis reveals 1) GQA questions contain superfluous information by way of over-specification and 2) the ability to effectively handle this extraneous information is crucial for models to improve performance. We hope this perturbed test set will allow researchers to better understand the compositionality of VQA questions and to further improve symbolic reasoning over real images.

5.2 Related Work

Visual Question Answering [14] requires an understanding of both visual and textual information. Pure deep learning methods that based on convolution, LSTM and attention have achieved good performance. For example, Fukui *et al*. [18] used multimodal compact bilinear pooling to combine visual and language features into a joint representation. Yang *et al*. [17] used stacked attention to refine the attended image region relevant to the question. Kim *et al*. [23] proposed bilinear attention network to learn attention between the two modalities with residual connections between multiple attention maps. Yang *et al*. [133] proposed a tiered relational reasoning method that dynamically attends to visual objects based on textual instruction. **Visual reasoning.** Prior work has suggested that above mentioned VQA models

may rely on dataset shortcuts and priors to predict answer [9, 10, 128, 28, 101, 134]. Therefore, recent efforts have focused more on visual reasoning with complex compositional questions that requires multi-step reasoning and true visual scene understanding. Johnson *et al*. [70] propose CLEVR that requires reasoning over synthetic scenes with compositional questions automatically generated using question templates. Hudson *et al*. [59] further constructed GQA, a dataset with real images and procedurally generated multi-step questions, for visual reasoning.

Attention is widely used in vision and language tasks, including image captioning [135, 136, 137, 138], visual question answering [17, 23, 133], referring expressions [139, 140, 141]. It is shown effective in learning distinct importance of images in an image group, of sub-regions over an image or of words in a sentence. Our work calibrates different concepts and operations, thus enabling the model to reason with weighted concepts and contextual operation importance.

Neural symbolic methods. [142, 143, 144] show impressive reasoning ability on abstract reasoning tasks like [85, 145]. For VQA, Andreas *et al*. [106] propose neural modular networks, which decompose a question into a functional program (reasoning steps) that can be executed by neural modules over the image. This method is further improved by executing the functional program explicitly [129, 146, 73, 132, 147] or implicitly [148, 107], manipulating visual and textual features using convolution or dual attention. Specifically, [130, 131, 149] propose a pure symbolic executor given pre-parsed or learned

explicit programs, and achieve state-of-the-art performance on CLEVR. Relatedly, Amizadeh *et al*. [150] propose a symbolic reasoner based on first order logic to diagnose reasoning behavior of different models. While symbolic methods provide interpretable programs, their reasoning capacity on real data is still limited [59]. Our work aims to reduce the performance gap between symbolic and non-symbolic models on real data.

5.3 Motivation

In this section, we provide simple examples to demonstrate how the dataset differences (between the synthetic CLEVR and the real GQA) affect the performance of neural symbolic methods. Interestingly, we find that the traditional design principles in neural symbolic methods, which are usually obtained from synthetic datasets, may not be optimal for the real-world datasets.

5.3.1 Normalized Concept Embedding?

For neural symbolic methods, at each step of execution, a similarity score between each object embedding and the learned concept semantic embedding is computed to select the target object that is being asked about (*i.e.*, selecting the object that is closest to the query concept) and to predict answers (*i.e.*, selecting the concept that is closest to the target object). By default, normalization is applied to both object embedding and concept embedding.

Interestingly, on the real-world GQA, we find this default strategy is not optimal; simply removing the normalization on concept embedding yields substantially better performance (+3.4%). This phenomenon indicates that

in addition to the angle alignment between object embedding and concept embedding, the magnitude of concept embedding is also informative for symbolic reasoning on real images.

We conjecture this is because the magnitude can represent the concept distribution, which is drastically different between synthetic datasets and real datasets. For example, while CLEVR contains only a relatively small and perfectly-balanced set of concepts (*i.e.*, 19 concepts including shapes, materials), real datasets deal with thousands of concepts which are far more complex and follows a long-tail distribution. We validate this hypothesis in Section 5.6—with a learnable magnitude for each concept embedding, we find its value is strongly correlated with concept frequency, *i.e.*, more frequent concepts tend to have larger magnitudes.

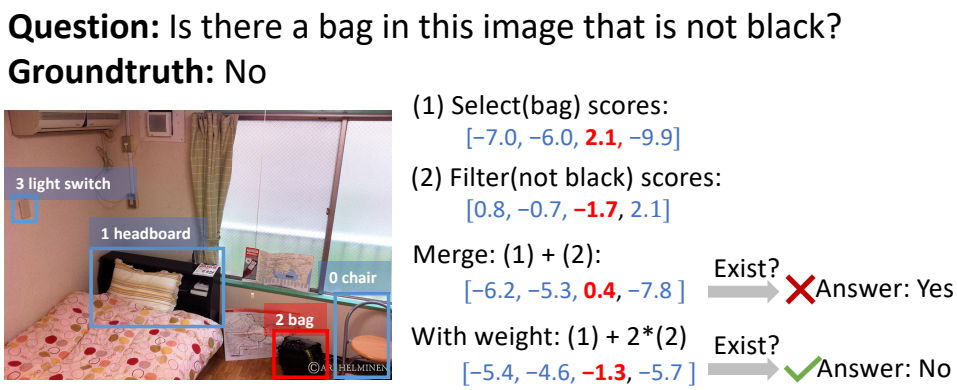


Figure 5.2: A failure case that can be corrected by re-weighting the operations. The $select(bag)$ operation overrides $filter(not\ black)$, thus lead to incorrect answer. This can be corrected by scaling up the result of $filter$ operation.

5.3.2 Module Re-weighting

In addition to this long-tailed distribution, the reasoning steps on real data are of varying importance during execution. For example, in most cases, the $select(noun)$ module are more discriminative than the $filter(attribute)$ or the $relate(relationship)$ operations, due to implicit entailment in natural language and over-specification of the question (*e.g.*, "little boy", "trees below the sky"). Therefore directly adapting symbolic methods to GQA will bias the model towards putting more focus on learning discriminative operations while neglecting the rest, resulting in errors on questions where all operations are important. For example, in Figure 5.2, the question asks for a bag that is not black; but $select(bag)$ operation produces large values, overriding the $filter(not\ black)$ step, leading to a "yes" answer, even though the bag is *not* in the required color.

Surprisingly, in this example, if we simply magnify the output of $filter(not\ black)$ operation by a factor of 2, the $filter$ operation then can successfully rule out the black bag, thus correctly answering the question. This result suggests that, while many questions contain redundant operations that the model tends to overlook, correctly re-weighting the operations is crucial for symbolic reasoning on real images.

5.4 Calibrating Concepts and Operations

Given the observations in Section 5.3, we next explore designing more sophisticated algorithms for automatically and effectively dealing with the complex

characteristics of real data (*e.g.*, long-tailed distribution and unequal reasoning steps), for the purpose of increasing neural symbolic methods' real-world applicability.

5.4.1 Formulation

In symbolic reasoning, a parser first parses a question $Q =< \hat{w}_1, ..., \hat{w}_l >$ into a tree-structured functional program P. The program P consists a set of modules $< p_1, ..., p_m >$ with dependency relationships between each other. As the functional program is either a chain or a binary tree, it can be linearized into sequence by pre-order traversal. Each operation p has its type p^t (e.g., *select*, $filter$), attribute p^a (e.g., *color*, *material*) and concept p^c (e.g., *red*, *plastic*). We denote the total number of module types, attributes and concepts as n_t, n_a, n_c, respectively. Then execution modules are composed on the fly, based on this generated program P. The module outputs are merged based on their dependency relationship and fed into the final module to get the answer **a**.

For scene representation, we first obtain a set of feature vectors $\mathbf{v}_i \in \mathbb{R}^d$ from the image I, with n objects detected in the image. Specifically, the feature vector **v** can be either visual features obtained from Faster RCNN [108], or the symbolic representation for each object (which can be obtained by concatenating distributions over N_c object categories and N_a attributes).

5.4.2 Basic Executor Architecture

Given the program P, the executor then executes it over input scene representations **v** to get a reasoning answer **a**. The basic executor principle follows the

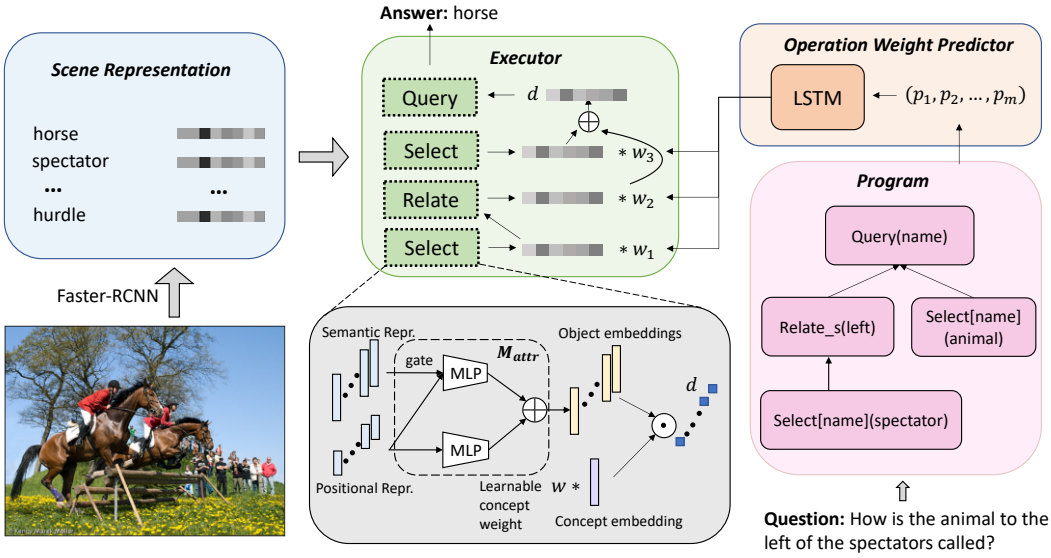


Figure 5.3: Overview of our method. We first parse the image into a symbolic scene representation in the form of objects and attributes, then parse the question into a program. In each reasoning step, a reasoning module takes in the scene representation and the instruction from the program, and outputs a distribution over objects. The Operation Weight Predictor predicts a weight for each reasoning module, which will be used to merge module outputs based on the program dependency. The final distribution is fed into the output module to predict answers.

design in [131].

As shown in Figure 5.3, each module (except for the output module) produces a distribution $\mathbf{d}$ over N objects in the image ($\mathbf{d} \in \mathbb{R}^N$), which are then merged based on their dependencies. In contrast to the default setup in the synthetic dataset, we use the mean operation (rather than minimum as in NSCL) to merge the module results due to its more stable training behavior. Finally, the output module takes in the object distribution produced by intermediate modules and queries/verifies the specified attribute of the selected object.

For module design, a semantic embedding $\mathbf{c}$ is learned for each concept

(*e.g.*, man, red, round, etc). Without loss of generality, we illustrate the computation for *select* and *query* modules. The architecture of the other module types can be found in supplementary materials.

We take the module $select[name](spectator)$ as an example. First, a small network $\mathcal{M}_{name}$ maps each object representation $\mathbf{v}_i$ into the concept embedding space, and then the similarity $\mathbf{s}_i$ between the embedded object representation $\mathbf{e}_i$ and the embedding of concept "spectator" ($\mathbf{c}_{spectator}$) is computed. This similarity $\mathbf{s}_i$ can be interpreted as the likelihood of each object being "spectator". The computation of *select* module can be summarized as the following:

$$\mathbf{e}_i = \mathcal{M}_{attr}(\mathbf{v}_i) \tag{5.1}$$

$$s_i = \text{sim}(\mathbf{e}_i, \mathbf{c}_{cept}) \tag{5.2}$$

$$\mathbf{d}_{select} = [s_1, s_2, ..., s_N] \tag{5.3}$$

where cosine similarity, *i.e.*, dot product of normalized $\mathbf{e}$ and $\mathbf{c}$, is used for similarity computation.

The detailed network architecture of the representation mapping network $\mathcal{M}_{attr}$ is shown in Figure 5.3. It gates the input object representation and passes it through a MLP to get the corresponding semantic embedding. The semantic embedding is then added with spatial embedding to get the final object embedding. The mapping networks $\mathcal{M}$ corresponding to different attributes share the same network architecture but with different weights.

We also briefly summarize the computation of *query* module below, as

another example:

$$\mathbf{e}_i = \mathcal{M}_{attr}(\mathbf{v}_i) \tag{5.4}$$

$$\mathbf{e} = \mathbf{d} \cdot [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_N] \tag{5.5}$$

$$\mathbf{a}_j = \text{sim}(\mathbf{e}, \mathbf{c}_j) \tag{5.6}$$

where the operation $\cdot$ refers to element-wise multiplication between two vectors, and $\mathbf{c}_j$ refers to the concept embedding of possible answers.

5.4.3 Calibrating Concepts and Operations

We hereby formally propose *Calibrating Concepts and Operations (CCO)*, which includes a concept calibration module and an operation calibration module, to help neural symbolic methods improving their applicability on real images. The overall design is illustrated in Figure 5.3.

Calibrating concepts. As diagnosed in Section 5.3, the magnitude of concept embedding $\mathbf{c}$ is informative for measuring the similarity between the object embedding and concept embedding. This motivates us to design an extra architectural element for explicitly capturing such information in magnitudes. Moreover, this designed architectural element is expected to be adaptive for different concepts, as each distinct type of operation is dealing with varying concept frequency distributions. For example, general concepts like "person" are common in *select* module, but not in *query* as the answers usually expect more specific concepts.

In light of these intuitions, we offer a simple solution—explicitly learning different embedding magnitudes for each module type. We expect the learned norm sizes can encode the concept distribution, thus more frequent concepts have larger norms sizes, leading to larger similarity values. Concretely, we calibrate concept embeddings by:

$$\mathbf{c}_{concept} = w^{type}_{concept}\mathbf{c}_{concept} \tag{5.7}$$

where w is different for each module type and each concept. This is applied whenever concept embeddings are used for similarity computation (*e.g.* in Equation 5.2). To this end, distinct types of modules share the same concept embedding direction, but varying magnitudes, corresponding to different concept distributions.

Calibrating operations. As shown in Section 5.3, on real images, it is important to enable the model to reason with different operation importance. To this end, we propose to customize the weight of each operation in the program. Specifically, a bi-directional LSTM weight predictor is used here to predict operation weights based on the whole program. For each operation p_i in the program, its weight w_i is computed as following:

$$\mathbf{e}_i = [\mathbf{e}^t_i; \mathbf{e}^a_i; \mathbf{e}^c_i] \tag{5.8}$$

$$\mathbf{h}_1, ..., \mathbf{h}_m = \text{LSTM}(\mathbf{e}_1, ..., \mathbf{e}_m) \tag{5.9}$$

$$w_i = \text{sigmoid}(W\mathbf{h}_i) \tag{5.10}$$

where m is the program length. The inputs $\mathbf{e}$ to LSTM is the concatenation of

the operation type embedding $\mathbf{e}^t$, the attribute embedding $\mathbf{e}^a$ and the concept embedding $\mathbf{e}^c$. The predicted operation weights are then used to merge outputs of the operations with a weighted-sum operation:

$$\mathbf{d}_i = \sum_{j \in \mathcal{D}(p_i)} w_j \mathbf{d}_j \tag{5.11}$$

where $\mathcal{D}(p_i)$ is the set of dependency operations of operation p_i. In this way, operations with higher weights play a more important role in the merging step.

Summary. With the proposed CCO, neural symbolic executor now is able to capture the underlying data characteristics and reason with learnable operation importance. As we will next show, CCO substantially boosts model performance on GQA, meanwhile largely reduces the performance gap between symbolic and non-symbolic methods.

5.5 Experiments

5.5.1 Dataset and Experiment Setup

Dataset. Our experiments are on GQA [59], which is a dataset focusing on reasoning and compositional question answering over real images. Building on top of Visual Genome dataset [79], it contains more than 110K images and 22M questions. Each image is annotated with a scene graph cleaned from Visual Genome that contains the information of objects, attributes and relationships. Each question comes with a corresponding functional program that specifies reasoning steps. By default, we use its balanced version with

943k, 132k, 13k and 95k questions in train, val, testdev and test split for training and evaluating.

Scene representation. We train a Faster RCNN with an additional attribute head using cross entropy loss following [50]. We train with 1313 object classes (lemmatized and with plurals removed) and 622 attributes. The model gets 24.9 mAP for object detection and 17.1 groundtruth attribute average rank.[1] The 1935-d concatenation of class and attribute scores are used as symbolic scene representation.

Implementation details. The inner dimension of our model is 300. The concept embedding is initialized using GloVe embedding [52]. We train our reasoning model using the Adam optimizer with an initial learning rate of 0.0005 and a batch size of 256. Linear learning rate is used with 2000 warm-up steps. We train the model for a total of 30 epochs, with early stopping (based on accuracy on the balanced testdev split) to prevent overfitting. To avoid confounding caused by parsing errors, we use *gold programs* to analyze the execution performance by default.

5.5.2 Execution Results

We choose NSCL [131] as our baseline model. By default, concept embeddings are normalized before similarity computation (cosine similarity) and operation results are merged by taking the average. After appplying minimal but necessary changes to NSCL for making it runnable on GQA, it achieves 47.01%

[1]Attribute prediction is evaluated by the average rank of groundtruth attribute in all the 622 attributes. We only consider the correctly detected objects (IOU>0.5) for attribute evaluation.

	Concept	Operation	Acc.
1 (Baseline)	Normalized	Average	47.01
2	Normalized	Calibrated	51.30
3	Unnormalized	Calibrated	54.65
4 (Ours)	**Calibrated**	**Calibrated**	**56.13**

Table 5.1: Accuracy comparison on the balanced GQA testdev split. Compared to the baseline, both concept calibration and operation calibration substantially improve model performance. The best performance is achieved by calibrating both concept and operation.

accuracy. We then integrate the proposed concept and operation calibration strategies on top of this baseline, while keeping other settings unchanged. As shown in the fourth row of Table 5.1, CCO helps the baseline gain a substantial improvement, *i.e.*, the accuracy is increased from 47.01% to 56.13%. This 9.12% improvement margin in accuracy demonstrates the effectiveness of our proposed method.

To further analyze the improvement brought by each individual component, we progressively add in our proposed concept calibration and operation calibration into the NSCL baseline. As shown in the second row of Table 5.1 where the operation calibration is added, it outperforms the baseline by 4.29%, demonstrating the effectiveness of operation calibration. We then remove the normalization of concept embeddings and keep the embedding magnitudes when computing similarity. As shown in the third row of Table 5.1, such strategy successfully leads to an additional 3.35% improvement. This result suggests that the embedding magnitudes are informative, which is consistent with our analysis in Section 5.3.1. In summary, these results support that both concept weighting and operation weighting are useful for improving the

NSCL baseline.

5.5.3 Ablations

Scene representations. Regarding scene representations, besides using symbolic representations, we also test model performance with other alternatives. To validate the correctness of our model design, we feed the operation modules with gold scene representation. Our CCO achieves 89.61% accuracy, which is similar to human performance (89.30%). This high upper bound indicates that model performance can be further improved by better visual perception.

We also examine the model performance by using visual features (Faster-RCNN feature after mean-pooling) as scene representation. Our CCO achieves 53.00% accuracy, where the 3.13% performance gap (*i.e.*, 53.00% *vs*. 56.13%) shows the advantage of the abstract symbolic scene representation over the dense visual features.

Program parsing. In all previous experiments, we apply gold program for facilitating performance analysis. While in this part, we now examine the model performance in the wild, *i.e.*, gold program is no longer available. In order to parse the question into functional program, we apply MISO, a popular sequence-to-graph parser used for parsing in a number of graph-based formalisms [151, 152, 153]. Different from simple sequence-to-sequence parser as in [129] that can only handle program with one argument, or the two-stage parser as in [147] that handles multiple arguments by hard constraints, MISO can automatically handle multiple arguments by treating the program

as a graph. The inputs to the MISO parser are word embedding sequences and output is a pre-order traversal of a program trees.

We present the parsing results in Table 5.2. We use exact match score, which is calculated by the percentage of predicted programs that exactly match the gold program, for measuring the quality of the predicted program. Our parser outperforms the parser in MMN [147] by a large margin of 6.05% in terms of exact match score. Nonetheless, interestingly, we find final model accuracy is less impacted by the quality of program—by executing either ours or MMN's predicted program, the difference in the final model accuracy is only 0.1%. This seemly "frustrating" result may suggest the performance of other components in current neural symbolic methods are severely lagged behind therefore are not able to cope with the advances brought by our strong parser.

Model	Exact match	Acc.
MMN [147]	85.13	54.01
Ours	**91.18**	**54.11**

Table 5.2: Parsing performance on `testdev_balanced` split, measured by exact match score and execution accuracy.

	Method	**Acc**	**Binary**	**Open**	**Const.**	**Plaus.**	**Valid.**	**Dist.**
Non-Symbolic	LXMERT [2]	60.33	77.16	45.47	89.59	84.53	96.35	5.49
	NSM [132]	63.17	78.94	49.25	93.25	84.28	96.41	3.71
	MMN [147]	60.83	78.90	44.89	92.49	84.55	96.19	5.54
Symbolic	∇-FOL [150]	54.76	71.99	41.22	84.48	-	-	-
	CCO (ours)	56.38	74.83	40.09	91.71	83.76	95.43	6.32

Table 5.3: Comparison with state-of-the-art symbolic and non-symbolic methods on the official testing split.

Comparing to the state-of-the-arts. To fairly compare different methods on GQA, we follow the training setups in [150, 147] where we first train the model on unbalanced training split then finetuned on balanced training split. Gold programs are used for training while parser predicted programs are used for evaluation. Performance is reported using the official evaluation metrics, including overall accuracy, accuracy on binary questions, accuracy on open questions, consistency, plausibility, validity and distribution.

We consider three non-symbolic methods (*i.e.*, LXMERT [2], NSM [132], MMN [147]) and one symbolic method (*i.e.*, ∇-FOL [150]) for performance comparison. In short, LXMERT is a representative multi-modal pretraining method; NSM is a graph-based model that achieves state-of-the-art performance on GQA; MMN is a modular method but is still based on dense features manipulation; ∇-FOL[2] is a symbolic method based on first order logic and contextual calibration. We summarize the model performance on the held-out test split in Table 5.3.

Compared to the previous state-of-the-art symbolic method ∇-FOL, our proposed CCO surpasses it by 1.58% in terms of accuracy. Moreover, as shown in Table 5.3, we note the performance gain over ∇-FOL is mainly on the binary questions (+2.84) and on predicting consistent answers for different questions (+7.2%).

We next compare with the state-of-the-art non-symbolic methods. Though our model still has lower accuracy than these non-symbolic methods, we note their performance on consistency, plausibility and validity is on a par

[2] ∇-FOL does not report full result on the official test split, therefore results on balanced testdev split is shown for comparison.

with each other. We conjecture this is due to the symbolic nature of our model, *i.e.*, the proposed CCO execute strictly according to the program, thus answers are plausible and valid, and questions with same underlying program get consistent answer. These results suggest that the proposed CCO largely reduces the performance gap between symbolic and non-symbolic methods on the real-world GQA dataset.

5.6 Analysis

5.6.1 Learned Embedding Magnitudes

To verify our motivation that the learned concept embedding magnitudes are informative for representing the unbalanced concept distribution in real dataset, we visualize the correlation between concept counts and their magnitude after calibration (in *query* module), *i.e.*, $\|\mathbf{c}_{concept}\|_2$ after calibration in Equation 5.7. In the plot, X-axis is the count of concepts in *query* module (taking log), and Y-axis is the learned magnitude of concept embeddings.

As verified in Figure 5.4, more frequent concepts consistently learn larger magnitudes, while less frequent concepts generally have smaller magnitudes. With larger magnitudes, the frequent concepts will produce values with higher confidence when computing similarity in the output of each module. Another interesting observation is that the magnitudes for few-shot concepts are not very consistent (*i.e.*, have larger variance), which is possibly caused by the insufficient number of training examples.

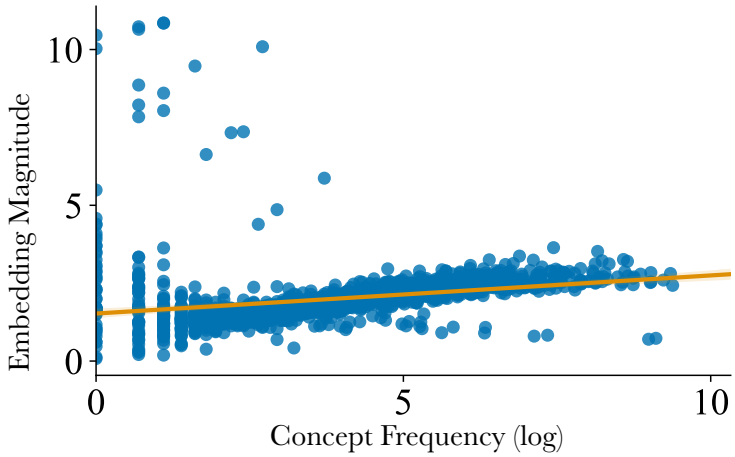


Figure 5.4: A positive correlation between learned embedding magnitude and concept frequency confirms our motivating intuition: more frequent concepts have larger magnitudes.

5.6.2 Perturbed Test Set

We create a perturbed testing data splits for the following purposes: a) we want to validate that proposed operation weighting strategy predicts larger weights for more important operations and smaller weights for unimportant ones; b) we need a test set for better studying the question over-specification in GQA dataset; and c) we aim to benchmark behavior of symbolic and non-symbolic methods in terms of how much information in the over-specified operations can be effectively utilized.

Specifically, this perturbed test set is created using the operation weights predicted by the learned LSTM operation weight predictor. We perturb the functional programs in balanced testdev splits by progressively removing the removable operations with smaller predicted weights[3]. Note that removable

[3]We set the weight thresholds to be $-\infty$, -2, -1, -0.5, 0, $+\infty$; resulting in removing 0%, 14%,31%,70%, 90%, 100% of removable operations, respectively

operations here refer to the intermediate operations that can be removed without syntactically breaking the programs, *i.e.*, *filter*, *relate* and their dependent operations. Then, we train a simple sequence-to-sequence generator to recover questions from the perturbed programs.

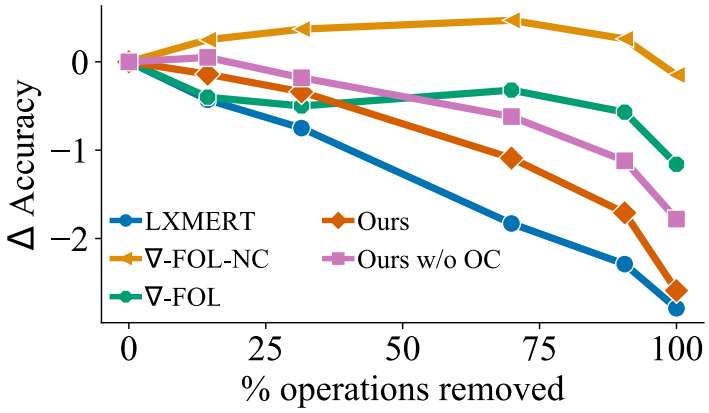


Figure 5.5: Accuracy drop of different models when the testing questions are progressively perturbed by removing reasoning operations with low weights.

The results are shown in Figure 5.5. We test five methods, including non-symbolic method LXMERT [2], symbolic method ∇-FOL [150], its variant ∇-FOL-NC which is a pure reasoner based on first order logic, our model, and ours without operation calibrating.[4] Our observations can be summarized as the following:

Validity of operation weights. All curves exhibit a sharper decrease at the end when more operations with higher weights are removed. In other words, the removal of operations with larger predicted weights will result in bigger negative influence on model accuracy. This validates the predicted weights

[4]Original accuracy of the five models (LXMERT, ∇-FOL, ∇-FOL-NC, ours, and ours w/o OC) are 58.13, 54.02, 51.86, 56.13, 55.49, respectively.

correctly represent operation importance.

Question over-specification. From the curves, we note while 59.0% questions in the balanced testdev split contain removable operations and are perturbed, less then 3.0% questions are incorrectly answered after removing those modules. This phenomenon suggests that for most questions in GQA dataset, the $filter$ and *relate* operations are not necessary for figuring out the answer, *i.e.*, removing all the intermediate attributes and relationships from questions does not change the answer for most of the questions.

Effectiveness of operation weighting. Interestingly, the performance of the pure logic reasoner ∇-FOL-NC and our model without operation weighting sees a slight increase when removing a small amount of operations. This phenomenon indicates that those operations are hard for models to learn thus can even derail the model predictions. This verifies our motivation for designing operation calibration as it helps the learning of $filter$ and *relate* modules.

Comparison of symbolic and non-symbolic methods. Compared to symbolic methods, non-symbolic methods have larger accuracy drops, therefore indicating they can more effectively utilize the information in adjectives and relationships. Moreover, methods with higher performance tend to have larger decrease when questions are perturbed. This suggests enhancing the model's ability to understand filtering adjectives and relationships is crucial for improving symbolic methods on real images.

5.6.3 Hard and Easy Subset

We additionally perturb the visual-hard and the visual-easy testing splits [150] and evaluate our CCO model on them. Specifically, the easy split contains questions that visually easy thus can be answered correctly by their differentiable first-order logic formula, while the hard split are harder in perception. In other words, the easy split contains questions that can be answered by a perception system alone, while the hard split contains images requiring more reasoning. With perturbed versions, we can investigate to what degree low-weight operations are implicated in multi-step reasoning for visually hard questions.

We summarize the model performance in Table 5.4. With more operations get removed, the accuracy drop on perturbed hard split is much larger than the easy split. This indicates that the visually hard questions force the model to better utilize every piece information in the question, while easy questions contain more redundant operations that are not necessarily needed.

threshold	**All**	**Easy**	**Hard**
$-\infty$(orig)	56.13	78.03	37.42
-2	-0.14	0.4	-2
-1	-0.34	0.13	-2.17
0.5	-1.09	-0.51	-3.04
0	-1.71	-0.93	-3.86
$+\infty$	-2.59	-1.88	-4.72

Table 5.4: Model accuracy on perturbed easy/hard splits.

5.7 Conclusion

To improve symbolic reasoning for VQA on real images, we propose to calibrate concepts and operations (CCO), which helps models handle the unbalanced concept distribution and unequal importance of reasoning operations. Experimental results demonstrate the effectiveness of the proposed method, where CCO outperforms several baselines by a large margin and reduces the performance gap between symbolic and non-symbolic methods. Additionally, we propose a perturbed test set for better understanding and analyzing model performance on real images. We hope this dataset can help researchers to further study the potential of symbolic reasoning on real images in the future.

Chapter 6

ExoViP: Step-by-step Verification and Exploration with Exoskeleton Modules for Compositional Visual Reasoning

Compositional visual reasoning methods, which translate a complex query into a structured composition of feasible visual tasks, have exhibited a strong potential in complicated multimodal tasks like visual question answering, language-guided image editing, etc. Empowered by recent advances in LLMs, this multimodal challenge has been brought to a new stage by treating LLMs as few-shot/zero-shot planners, *i.e.*, visual-language programming [154]. Such methods, despite their numerous merits, suffer from challenges due to LLM planning mistakes or inaccuracy of visual execution modules, lagging behind the non-compositional models. In this work, we devise a "plug-and-play" method, ExoViP, to correct the errors at both the planning and execution stages through introspective verification. We employ verification modules as "exoskeletons" to enhance current vision-language programming schemes.

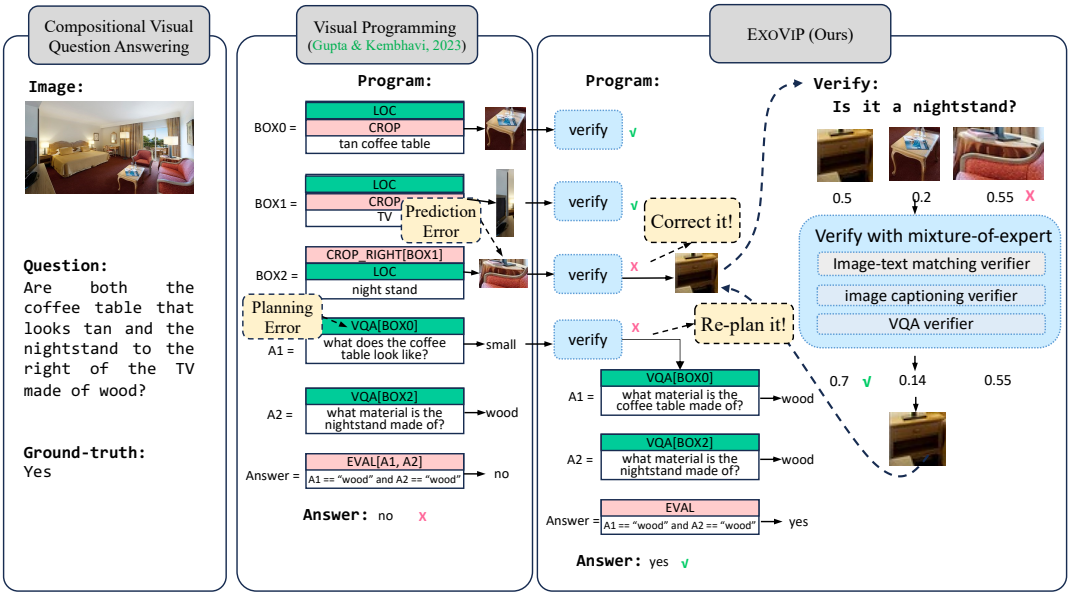


Figure 6.1: An overview of ExoViP. The prediction after each step is verified by the proposed "Exoskeleton" verification modules, which contain a mix of three sub-verifiers. The verified scores help correct the errors in the vision module predictions or refine the reasoning programs planned by LLM.

Specifically, our proposed verification module utilizes a mixture of three sub-verifiers to validate predictions after each reasoning step, subsequently calibrating the visual module predictions and refining the reasoning trace planned by LLMs. Experimental results on two representative vision-language programming methods showcase consistent improvements on five compositional reasoning tasks on standard benchmarks. In light of this, we believe ExoViP can foster better performance and generalization on open-domain multimodal challenges.

6.1 Introduction

Compositional visual reasoning tasks, such as visual question answering or image editing following language instructions, are challenging multimodal tasks that require complex multi-step visual reasoning based on the language instruction. Compositional methods like neural modular networks [155, 156, 157, 158, 159, 160], which translate the complex language instruction into feasible individual visual tasks, has been successful in this task. However, traditional compositional methods require well-designed neural modules for specific datasets, thus struggle in generalization to open domains. In addition, the intermedia embedding and attention among the neural modules can not be improved by introducing supervision signals or feedback, so the performance of these works is limited to the end-to-end training mechanism. Recently, empowered by the advances in LLMs such as in-context learning and train-of-thought reasoning [161, 162, 163, 164, 165]. recent methods like VISPROG [154] and ViperGPT [166] apply LLMs as zero-shot/few-shot planners to solve visual reasoning tasks, *i.e*. visual language programming. These visual language programming methods leverage off-the-shelf pretrained vision models and compose them step by step according to the reasoning trace planned by LLMs, yielding interpretable intermediate results and highly generalizable reasoning ability.

However, despite their merits, current visual programming methods still suffer from challenges due to the failure of the LLM planning or the visual modules, lagging behind the performance of non-compositional models. To analyze the drawbacks, we manually checked 100 randomly sampled failure

cases of VISPROG [154] on the visual question answering GQA dataset [167]. We find that most of the failures can be classified into two categories: (1) around 30% of the failures are due to planning errors: LLM can not parse the language query into a correct solvable program; (2) more than 40% of the failures are due to module error: the visual modules are not able to correctly execute the program. The others (less than 30%) are caused by synonyms (*e.g.* "woman" vs "lady") or ambiguity in the questions.

Motivated by these failure modes, in this work, we introduce EXOVIP, a "plug-and-play" method that uses "exoskeleton" verification modules to verify the reasoning results step by step, thus correcting the module errors and refining the LLM planning traces. In Fig. 6.1, we demonstrate how EXOVIP helps correct the two types of errors. Specifically, the verification module contains a mixture of three sub-verifiers, including an image-text matching verifier, an image captioning verifier, and a visual question answering verifier. The verification module validates the correctness of the predictions of the vision modules step by step and calibrates them to correct the module errors. Furthermore, to refine the planning traces, we build a reasoning trace tree based on the verification scores as well as the self-correctness score from LLMs [168], and search through the tree to find the best trace that has the highest score.

To demonstrate the effectiveness of EXOVIP, we apply our method to two recent visual programming methods: self-defined programs, *i.e.*, VISPROG [154] and Python code programs, *i.e.*, ViperGPT [166]. We run experiments on five compositional visual reasoning tasks: compositional image

question answering on GQA [167]; referring expression understanding on RefCOCO and RefCOCO+ [169, 170], natural language for visual reasoning on NLVR [171], visual abstract reasoning on KILOGRAM [172], and language-guided image editing on MagicBrush [173]. Experiment results show consistent improvements with the two models on the five tasks. In light of this, we believe ExoViP can foster better performance on open-world compositional reasoning tasks. To summarize, our main contributions are as follows:

- We introduce the "exoskeleton" verification modules for compositional visual reasoning, which verifies the correctness of vision module predictions step by step.
- We show how the verification modules are leveraged to correct the module errors by calibrating the module predictions, and to correct the planning errors by tree searching considering both verification scores and LLM self-correctness.
- We apply our method on two models and show consistent improvements over five tasks, showing the effectiveness of ExoViP.

6.2 Related Work

LLMs in multimodal tasks. LLMs brought great convenience to multimodal tasks with their generalizability and knowledgeability. Generally, there are three ways researchers use LLMs to solve multimodal tasks. Some researchers incorporate additional parameters to adjust LLMs for use in multimodal domains, then fine-tune the model with the LLMs either frozen [174, 175, 176, 177, 178, 179, 180] or unfrozen [181, 182, 183]. Others take language model as

an expert, and mixture it with experts from other modalities, such as vision, speech to collaborate on various kinds of multimodal tasks [184, 185, 186]. In this work, we mainly focus on the third way which adopts LLM's planning ability in parsing complex queries. VISPROG [154] takes LLM to compose models for queries by generating programs. The strong zero-shot performance of VISPROG on a range of vision-language tasks demonstrates its potential in multimodal tasks involving complex reasoning. ViperGPT [166] leverages LLMs to generate Python code, which composes a set of available modules. MM-REACT [187] builds a multi-round, dialogue-based system to call a set of vision experts by designing the prompt of LLMs. However, the performances of these works are hindered by both the parsed planning chain and the visual experts. Inspired by the excellent performance gain from the step-by-step verification [188], we improve this train of work with additional verification strategies.

Compositional multimodal methods. Compositional methods have long been explored to improve neural models' interpretability and reasoning ability. At an early stage, neural module networks (NMN) [155, 156, 157, 158, 159, 160] compose neural models to end-to-end differentiable networks. However, the pre-defined neural modules have limited applications on open-domain challenges, and the intermedia embedding and attention makes it difficult to construct intermedia supervision signals. Recently, the presence of LLMs has made it possible to automatically compose various kinds of finetuned neural models [184, 154, 166, 187, 186] or external tools [189, 190, 191, 192, 193, 194]. These works allow us to diagnose the intermedia rationales of the reasoning

process. However, human annotation of these intermedia results can be rather time-consuming. In this work, we make ways to correct errors in the intermedia results without any human intervention.

Self-correctness in LLMs. Although LLMs achieve great success in various tasks, there are many errors in LLM-based natural systems [168]: hallucination [195, 196], unfaithful reasoning [197, 198, 199], toxic, biased, and harmful contents [200], flawed code. One popular way to fix these errors is to use the LLMs themselves [201, 202, 203, 204] to obtain feedback, which can be adopted to correct the errors. Motivated by the self-correction capability of LLMs in addressing mistakes from LLM-powered natural language systems, some researchers introduce the self-correcting strategy to reduce the reasoning chain in multimodal frameworks. IPVR [205] additionally utilizes LLMs to generate the rationale supporting the answer, checks the generated rationale with a cross-modality classifier, and makes sure that the rationale can consistently infer the predicted output. IdeaGPT [206] takes another LLM as a reasoner to get the final answer by summarizing the intermedia results from visual experts. Additionally, the reasoner helps to improve the results iteratively through self-consistency. However, it's intuitive that LLM's self-correction ability would be limited by the LLM itself. In our work, we combine the feedback from LLM and other visual experts to verify the intermedia results and the planned reasoning chain.

6.3 Preliminaries

Task Definition. Our work focuses on a set of Visual Compositional Reasoning (VCR) tasks, such as visual question answering, referring expression understanding, visual reasoning using natural language, abstract reasoning, language-guided image editing. These VCR tasks require compositional reasoning about an image input I and a text input T, and predict the output, *e.g.* answer to a given question, edited images given a language instruction, etc.

Visual-Language Programming (VISPROG). VISPROG [154] is a zero-shot model for the VCR tasks, utilizing LLMs and pretrained vision models. VISPROG first uses LLMs to decompose the complex text description into a sequence of individual operations, then executes each operation by calling various pretrained visual operation models, including object detectors, image captioners, VQA models, image generators, *etc*. In other words, different vision models are composed in a way that is specified by the LLM to get the prediction. Given the input text T, an LLM transforms it into an executable program P containing a sequence of operations: $P = \{o^1, \ldots, o^n\}$, where n is the number of operations. Each operation o^i can be executed by some symbolic operations (*e.g.*, "crop", "and", "or"), or by calling some pretrained visual models (*e.g.* CLIP [207], BLIP [208]). The output of operation o^i is denoted as a_i. The final prediction is derived after we execute all the operations. However, this perspective highlights two key shortcomings of existing approaches: i) module error, the operation models can not predict the answer correctly; ii) planning error, the LLM might generate unfaithful reasoning.

6.4 ExoViP: Exoskeletons with Verification and Exploration

To address the aforementioned shortcomings, we propose ExoViP, a framework that adopts exoskeleton verification modules to calibrate the prediction of the execution modules and refine the reasoning path with tree searching. Fig. 6.1 depicts the overall framework.

For each operation o^i, we get a set of candidate answers $\{a_1^i, \ldots, a_k^i\}$, with confidence scores $\{p_1^i, \ldots, p_k^i\}$. Unlike VisProg, which directly takes the top answer, we use additional verification modules to verify each candidate answer, thus producing verification scores $\{s_1^i, \ldots, s_k^i\}$. Then the verification scores s are used to calibrate the original scores, so the errors made by the execution modules can be corrected. Additionally, we use the verification scores to search for a program with high verification scores, in order to refine the execution program P by tree-searching.

In this section, we will first introduce the verification modules, and then describe how the verification results are applied to correct the results of execution modules, and to search for the reasoning trace.

6.4.1 Verification Modules

The verification modules aims to verify the candidate answers $\{a_1^i, \ldots, a_k^i\}$ given an operation o^i. For example, the `LOC(nightstand)` operation returns a set of candidate bounding boxes containing a nightstand, then the verification module verifies whether each of the returned boxes contains a nightstand and produces verification scores.

Our verification module is a mixture of three sub-verifiers, including an image-text matching verifier, an image captioning verifier, and a visual question answering verifier. Each verifier is a pretrained vision-and-language model that is taken off the shelf. The outputs of the three verifiers are combined as the final verification score. Note the verification model does not introduce additional pretrained models, as these verifiers are from the execution modules of VISPROG.

Image-text matching verifier calculates the similarity between the whole images and all candidate sentences, which returns the semantic representation of the image-sentence pair. We construct the candidate sentences $\mathcal{T}_{ans}$ by filling the template "a photo of" with candidate answers. In this work, we select CLIP [207] to calculate the similarity between images and sentences.

$$s_{ans}^{itm} = \text{ITM}(\mathcal{T}_{ans}, img) \tag{6.1}$$

Image captioning verifier leverages natural language to describe the visual details of the image. We first get the caption of the image $\mathcal{C}_{img}$ by BLIP [208]. We then construct the descriptions of candidate answers $\mathcal{C}_{ans}$ with the template "the image describe". Specifically, for candidate question-answer pairs, we initially transform the pair into a sentence before inserting it into the template. After that, we calculate the sentence semantic similarity [209] between the captions and the constructed descriptions as the verification score.

$$s_{ans}^{cap} = Sim(\mathcal{C}_{ans}, \mathcal{C}_{img}) \tag{6.2}$$

VQA verifier is more flexible than others, which offers us more opportunities to evaluate the advanced relationships between image and language, such as entailment and factual consistency. Slightly different from the other two types of models, for VQA verifier, we design templates w.r.t. the neural modules. For example, we use “Is there any object in the image ?" for the object detection model, and use “Does this part looks like object ?" for the classification model used in the abstract reasoning task. We determine the verification score by BLIP [208] by calculating the difference in answer probabilities $\mathcal{Q}_{ans}$ between "yes" and "no".

$$s_{ans}^{vqa} = \text{VQA}(\mathcal{Q}_{ans}, True) - \text{VQA}(\mathcal{Q}_{ans}, False) \tag{6.3}$$

Verification score Having the scores from each individual verification module, we compute their average to get the verification score for each given answer.

$$s_{ans} = \text{avg}(s_{ans}^{itm}, s_{ans}^{cap}, s_{ans}^{vqa}) \tag{6.4}$$

Negative sampling. Empirically, we find that directly applying this verification score does not work well, because the score scales for different kinds of candidates are not well-calibrated. Motivated by recent works in truthfulness [210], commonsense [211], and bias [212], we propose to take the difference of a candidate answer a_j with its antonym n_j as the final verification score. More specifically, the antonym n_j is selected based on the text embeddings from CLIP [207], *i.e.* the word of lowest embedding similarity is selected. For example, the antonym of “nightstand” is “stocking”. We then compute the difference of the verification scores of the candidate answer and its antonym,

and get the final verification score. Mathematically, given a candidate answer a^j, the final verification score is

$$s_j = s_{a_j} - s_{n_j} \tag{6.5}$$

Calibration using verification scores After obtaining the verification scores of all candidate answers $S = \{s_1, \ldots, s_k\}$, we normalize them as weights and calibrate the candidate predictions.

$$p'_j = w_j * p_j, \tag{6.6}$$

where w_j is the normalized verification score. More specifically, the verification score s_j is re-scaled to $w_j = \frac{s_j - s_{min}}{s_{max} - s_{min}} \cdot (\tau - \frac{1}{\tau}) + \frac{1}{\tau}$, where τ is a hyper-parameter controlling the scaling factor (s_{min}, s_{max} are the minimum or maximum of all the candidate scores.

6.4.2 Exploration with Reasoning Trace

To correct the second type of reasoning errors, *i.e.* planning errors, we further apply the verification scores to refine the reasoning trace predicted by LLMs. Motivated by the recent works showing that searching through a combinatorial problem space can greatly improve the performance of LLMs for complex tasks [213, 214, 215], we introduce our dynamic reasoning trace searching procedure, which takes advantage of both the LLM self-correctness potential and our verification modules.

The reasoning trace searching procedure is represented as a tree structure, where each node of the tree is a reasoning operation. To get the best reasoning

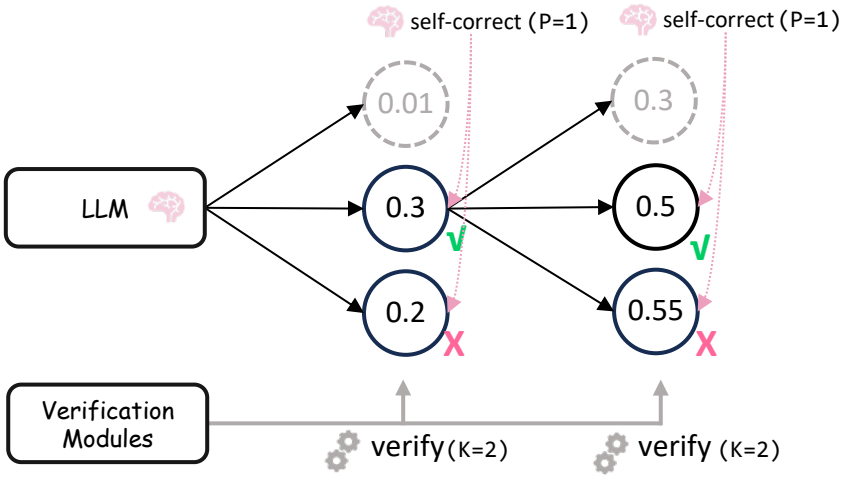


Figure 6.2: Search of the reasoning trace. We beam search through the program tree, based on the verification scores as well as the LLM self-correctness.

trace, we search from the tree using the beam search algorithm [216, 217, 218], which has long been proven to be effective in sequence-to-sequence problems. In each step of searching, we consider both the verification scores and the LLM self-correctness scores.

More specifically, our trace searching procedure contains three steps. First, in order to generate more diverse reasoning traces to search from, we randomly perturb the in-context examples (*i.e.* change the order or remove some samples of examples) in the prompt for LLM. Second, after we get the result of candidate neural modules, we sort them according to the verification scores and select the top K candidate reasoning traces. Third, because the verification scores can be very close for the selected K traces, we further use the self-correctness ability of LLMs to reorder the K traces and select the top P from them ($P < K$). If the verification score is zero at some step, we re-plan the search trace.

6.5 Experiments

6.5.1 Setup

We set up experiments on the following five tasks.

Compositional image question answering on GQA. GQA [167] is a large-scale dataset containing complex reasoning questions about real-world images in MSCOCO-style [219]. Considering the large size of the dataset, in order to balance the cost of LLM API and the diversity of evaluation dataset, we follow the setting of VISPROG [154] and sample a subset from GQA for evaluation. We randomly sample 5 samples from the balanced val set and 20 samples from testdev set of each question type. *e.g.* "weatherVerify" for judging the weather, "twoCmomon" for judging common attributions of two objects. In summary, there are 102 question types and 2327 questions in our test set.

Referring expression understanding on RefCOCO and RefCOCO+. Given a natural language query describing a region in a given image, the referring expression understanding task requires identifying the bounding box of the object in the image being referred to. RefCOCO and RefCOCO+ [169, 170] are two standard datasets for this task. We randomly sample 2 samples per type from the test set from RefCOCO dataset and RefCOCO+ dataset. In summary, our test set includes 66 types, *e.g.* "bicycle", "backpack", and 261 queries.

Natural language for visual reasoning on NLVR2. In NLVR2 [171], given a description of a collection of images, the model needs to justify whether the description is correct or not (binary classification). The task requires dealing

with various kinds of linguistic phenomena, like numerical expressions, quantifiers, coreference, negation, etc. In this work, we use the NLVR2 balanced test set for evaluation, which includes 2316 questions and corresponding image pairs.

Visual abstract reasoning on KILOGRAM. KILOGRAM [172] contains richly annotated tangram puzzles and requires the model to understand the abstract tangram shapes (*e.g*. dog, bird) and classify them. Specifically, given a textual description and a set of images, the task is to select the image corresponding to the description. This task evaluates the ability to generalize through abstraction, using visually ambiguous stimuli. We conduct experiments using the test set, where the textual descriptions solely contain the whole-shape description, and the images include parts with different colors. The test set contains 1,251 descriptions, with each one paired with 10 images.

Language-guided image editing on MagicBrush. This task requires editing an image according to a natural language instruction, keeping the other area of the image unrelated to the instruction unchanged. The MagicBrush dataset [173] supports various editing scenarios including single-/multi-turn. Considering the accuracy of automatic evaluation metrics and the costs of human evaluation, in our experiments, we only choose the samples involving single-turn image editing to evaluate our method. In total, there are 100 examples in the test set. Following [173], we select the CLIP-I and DINO, which measure the image quality with the cosine similarity between the generated image and reference ground truth image using their CLIP [207] and DINO [220] embeddings.

6.5.2 Main Results

We first apply EXOVIP to VISPROG and show results on the five tasks. Then we apply it to the python-code-based compositional reasoning method ViperGPT to demonstrate its generalizability.

6.5.2.1 Compositional Visual Question Answering

Baseline Model We set up the experiments following the settings in the official VISPROG implementation.[1] Moreover, we select BLIP-flant5-xxl[176] and InstructBLIP-flan-t5-xl[179] as additional baselines, which are strong vision-language models incorporating LLMs and pretrained on large vision-language datasets. These baselines have shown strong zero-shot ability on various tasks like image caption and visual question answering.

Table 6.1: Results for compositional visual question answering on GQA.

	Methods	**Accuracy**
	BLIP2-xxl [176]	49.20
	InstructBLIP-flant5-xl [179]	55.39
0	VISPROG [154]	57.41
1	EXOVIP w/o self-correctness & negative sampling & search	57.11
2	EXOVIP w/o self-correctness & search	58.53
3	EXOVIP w/o self-correctness (whole search)	59.17
4	EXOVIP w/o self-correctness (beam search)	60.57
5	EXOVIP w/o verification (beam search)	60.16
6	EXOVIP	61.49

Analysis We apply our method to VISPROG and report the results on GQA

[1]Becasue VISPROG doesn't release their sampled evaluation subset, we do sampling following the VISPROG paper and evaluate all the methods on our sampled evaluation set.

Figure 6.3: Distribution of verification scores w. and w/o trace searching.

Table 6.2: Analysis on the sub-verifiers.

Methods	**Accuracy**
Base	58.14
ITM	59.26
Caption	59.22
VQA	59.35
All	60.03

in Tab. 6.1. While VISPROG has already demonstrated good performance (57.41) compared with BLIP2 and InstructBLIP, our method further improves its performance to 61.49, showing a significant performance boost. Note that our method does not introduce extra modules or knowledge compared with VISPROG, since the verification modules come from VISPROG itself.

To verify the effectiveness of each component in our method, we run a series of analysis experiments on our method (also in Tab. 6.1). We have the

following observations:

1. *Negative sampling is key to verification modules.* Naively adding the verification modules (line-1) does not work, even making the performance worse. But when we introduce the negative sampling strategy using antonyms to the verification modules (line-2), the performance boost becomes significant.

2. *Exploration with reasoning trace matters.* In line-3, “whole search" means we use LLMs to obtain a set of complete planning traces, then execute all the traces to get the final verification scores, and select the best trace with the highest verification score. The “beam search" strategy (line-4) means we select next step according to current verification scores. While “whole search” helps, “beam search” can further improve the accuracy to 60.57 from 59.17, which indicates the effectiveness of our tree-like step-by-step searching strategy.

3. *Self-correctness does help but is less significant than verification mechanism.* In line-5, We only use LLM self-correctness during trace searching, without using the verification scores. While the result shows an accuracy gain of 2.75 over the original VISPROG, applying both leads to further better performance.

4. EXOVIP *achieves the best performance with the collaboration of all the introduced components.* In line-6, we combine all the introduced components and get the final performance of 61.49, showing a significant boost ((4.08). We believe our method successfully incorporates the verification modules with

the LLM self-correctness ability.

Analysis on the sub-verifers. We evaluate the effects of different types of verification modules with the setting of the best demonstration setting. As is illustrated in Tab. 6.2, Different verification modules share similar boost gain, but a mixture of these modules can benefit more.

Analysis on the trace-searching strategy. We calculate the verification scores among different samples and plot the distribution of the verification scores in Fig. 6.3. We find two advances brought by the searching strategy. First, the average of the verification scores significantly improved after we applied our search strategy. Secondly, the variance gets larger after applying the search strategy, which indicates our method can potentially make use of the verification scores to prompt the effectiveness of the reasoning traces.

Analysis on invalid programs. We calculate the percentage of failure cases that can not be correctly executed by the program interpreter. We are delighted to find out that our method reduces the error rate from 5.84% to 3.82%, which indicates our method can predict more executable plan routines compared to the baseline VISPROG.

Table 6.3: Results on RefCOCO and RefCOCO+.

Methods	**IoU**
Qwen-vl-chat-7b [221]	32.54
VISPROG [154]	27.28
EXOVIP	31.50

Table 6.4: Visual reasoning on NLVR.

Methods	Accuracy
OFA-large [222]	58.38
VISPROG [154]	67.66
EXOVIP	67.96

Table 6.5: Abstract reasoning on KILOGRAM.

Methods	Accuracy
CLIP-large [207]	27.26
VISPROG [154]	24.46
EXOVIP	26.22

6.5.2.2 Visual Language Grounding

Baseline Model We adopt the Qwen-vl-chat-7b [221] as the baseline. Qwen-vl-chat-7b is a pre-trained large vision-language model that uses Qwen-7B with further training with aligned techniques. Qwen-VL outperforms current SOTA generalist models on multiple VL tasks and has a more comprehensive coverage in terms of capability range.

Analysis As demonstrated in Tab. 6.3, although our method can't achieve SOTA (Qwen-VL) on the RefCOCO dataset, it helps bridge the gap between VISPROG and the large vison-language model. While Qwen-VL is built on a LLM with 7 billion parameters, which is trained on trillions of tokens from the corpus, our method assembles a team of experts whose collective parameters total less than 1 billion. We believe our method can be improved with more

Table 6.6: Image editing on MagicBrush.

Methods	CLIP-I	DINO
InstructPix2Pix [223]	84.19	69.60
VISPROG [154]	90.82	82.70
EXOVIP	91.27	83.40

advanced experts.

6.5.2.3 Natural Language Visual Reasoning

Baseline Model We take the OFA-large [222] as baseline. OFA unifies a diverse set of cross-modal and unimodal tasks in a simple sequence-to-sequence learning framework.

Analysis Tab. 6.4 shows the results. Although VISPROG exhibits strong complex reasoning ability over the end-to-end model, our method can hardly further improve its performance. We believe this is because we only take VQA modules to solve NLVR problems. One on hand, the performance of decomposed VQA steps is hindered by the performance of VQA model, especially when there is error accumulation among a sequence of VQA steps.

6.5.2.4 Visual Abstract Reasoning

Baseline Model We use the CLIP-large [207] as a baseline to test its performance on the text-to-image retrieval task proposed by KILOGRAM.

Analysis For our method, given an object, we adopt the LLM to get its possible semantic parts. At the same time, we segment the image into several

Figure 6.4: Qualitative results of text-guided image editing on MagicBrush.

visual parts. After that, we align the semantic parts with the visual parts to enhance the matching process. In Tab. 6.5, we see the gap between VISPROG and CLIP. Although our method decreases the performance gap, the compositional method still can not achieve SOTA. Since part identification has already been demonstrated to play an important role in human abstraction [224]. We believe our method can be enhanced by introducing a better scene segmentation model.

6.5.2.5 Text-guided Image Editing

Baseline Model We take InstructPix2Pix [223] as a baseline. InstructPix2Pix is a conditional diffusion model trained on GPT3 augmented datasets.

Analysis Tab. 6.6 and Fig. 6.4 show the results on MagicBrush. While non-compositional methods are likely to change unrelated pixels, compositional methods are more controllable.

6.5.3 Generalizability of our Method

Table 6.7: Results for ViperGPT on GQA.

Methods	Accuracy
ViperGPT [166]	45.47
ViperGPT+ExoViP	46.84

To demonstrate the generalizability of our method, we apply our method to another compositional method, ViperGPT, which composes available modules by generating Python codes. We equip ViperGPT with our method and test its performance on the GQA dataset. We show the results in Tab. 6.7. We find the performance boost is less significant than which on VISPROG. We analyze this due to ViperGPT provides a few examples in the demonstration and it turns the parameter of the code-generation model to make it deterministic to generate subroutines. In other words, ViperGPT benefits little from our reasoning trace-searching strategy.

6.6 Conclusion

In this work, we identify two key types of errors in existing compositional methods: planning errors and module errors. To address these errors, we introduce an innovative verification framework EXOVIP. This framework verifies the correctness of vision module predictions. It corrects module errors by calibration and refines the planning process through tree searching. During this process, it considers both verification scores and the self-correctness of

LLM. Applying the ExoVIP to two existing models, we achieve significant performance improvements across five different tasks. The results reinforce the promise and potential of ExoVIP on various open-world compositional reasoning tasks, marking an important milestone in the realm of multimodal tasks involving complex reasoning.

Chapter 7

Localization *vs*. Semantics: Visual Representations in Unimodal and Multimodal Models

Despite the impressive advancements achieved through vision-and-language pretraining, it remains unclear whether this joint learning paradigm can help understand each individual modality. In this work, we conduct a comparative analysis of the visual representations in existing vision-and-language models and vision-only models by probing a broad range of tasks, aiming to assess the quality of the learned representations in a nuanced manner. Interestingly, our empirical observations suggest that vision-and-language models are better at label prediction tasks like object and attribute prediction, while vision-only models are stronger at dense prediction tasks that require more localized information. We hope our study sheds light on the role of language in visual learning, and serves as an empirical guide for various pretrained models.

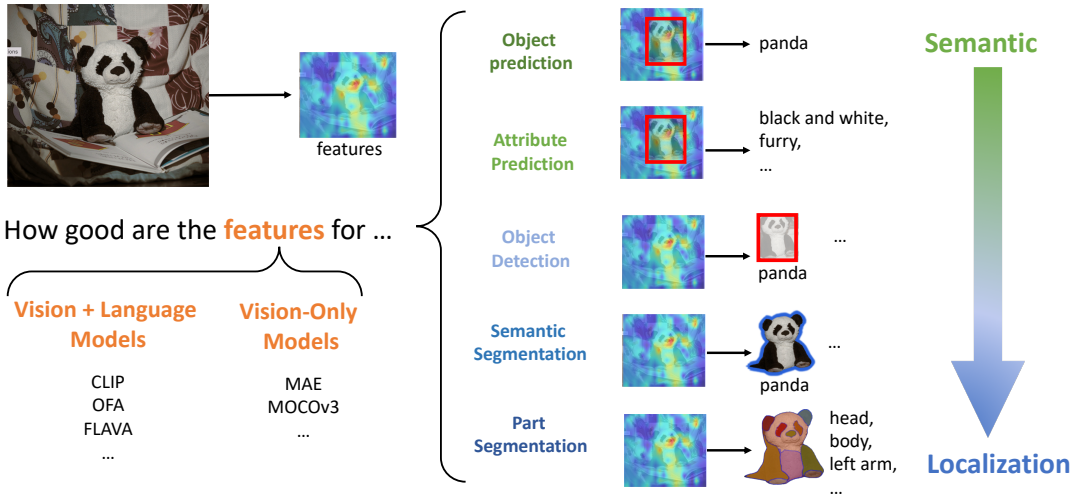


Figure 7.1: We compare the visual representations from unimodal and multimodal models on five tasks, in order to probe the semantics and localization knowledge encoded in the representations.

7.1 Introduction

The joint learning of vision and language offers mutual benefits. As evident by the recent advancements in vision-and-language pretraining (VLP) models [225, 226, 227, 228], they attain not only impressive performance on multi-modal tasks like visual question answering, but also on specialized uni-modal vision tasks like ImageNet classification [229], or language tasks GLUE language understanding [230].

Despite the superior performance, there is little understanding of *how multimodal learning can help visual representations*. Therefore, we hereby are motivated to compare the visual representations in existing vision-and-language (VL) models and vision-only (V) models from a probing perspective. Specifically, we probe the visual representations through a range of probing tasks that evaluate different properties, including semantics knowledge and localized

information, in order to gain a fine-grained understanding of the visual representations. This is inspired by recent works on multimodal feature probing [231, 232], which studies the opposite question to ours, *i.e.*, the role of vision in language models.

Fig. 7.1 illustrates our probing pipeline. We first extract image features using different pretrained models, and then train a simple prediction head to align the model's representation space with the label space of interest. We make the head as simple as possible based on the intuition that less expressive heads can more selectively reflect the quality of the representations [233]. The probing is done on various tasks and datasets: object name classification on the Visual Genome dataset [79], attribute prediction on the VAW dataset [234], object detection and instance segmentation on the MSCOCO dataset [235], and semantic object part segmentation on the PartImageNet dataset [236]. With these probing tasks, we compare vision-and-language pretrained models including OFA [227], FLAVA [228] and CLIP [225] with advanced vision-only models including MAE [237] and MOCOv3 [238].

Interestingly, our experiments suggest that VL models are much better at the label prediction tasks (*e.g.*, object class and attribute prediction), while vision-only models are stronger at dense prediction tasks like object detection and segmentation. In other words, multimodal models encode more semantic information in visual representations to better predict fine-grained labels, but fail to enrich the localization information that is required by spatial-aware tasks. This finding is further verified by a more detailed analysis of the segmentation and attribute prediction results, which reveals intriguing

properties of the unimodal and multimodal representations.

In summary, we probe the visual representations in popular VL and vision-only pretrained models on a broad spectrum of tasks and suggest that multi-modal representations encode better semantics. We hope our extensive probing results can serve as a fine-grained benchmark for the publicly released pretrained models, which provides an empirical guide to help researchers choose which model to use for different downstream tasks. Moreover, by offering these insights into the role of language in multi-modal learning, we hope to catalyze future explorations in this direction.

7.2 Related work

Vision-and-language pretraining (VLP). VLP methods perform well on multi-modal downstream tasks like visual question answering [14] and image captioning [239] and show potential on single-modal tasks. For example, dual encoders trained with a contrastive loss like CLIP [225] and ALIGN [226] achieve superior visual learning performance. While earlier VLP methods (like LXMERT [2], UNITER [240], OSCAR [5], VinVL [76]) rely on image features extracted by separately trained vision models like Faster-RCNN [104] or Resnet [241], more recent works learn the visual features jointly with language. Representative works include OFA [227], Florence [242], FLAVA [228], Unified-IO [243], CoCa [244], and SimVLM [245] etc. We refer readers to [246] for more details.

Vision and language benefit each other. Several recent works in NLP

suggest that multimodal learning can help language understanding. Vokenization [247] suggests vision improves the grounding ability of language models. [248] shows reduced reporting bias in multimodal world. Z-LaVI [249] and VIDLANKD [250] show language understanding performance can be improved by better visual imagination or knowledge distillation from videos. Recent work [232] analyzes language and multi-modal models and shows that vision can help language models learn better visual commonsense knowledge and mitigate reporting bias. However, there is little understanding of the opposite question, *i.e.* how does the visual learning differ in multimodal and unimodal models.

Probing. Probing is a widely used strategy in NLP for interpreting representations [251, 252]. Various works use probing to show that language representations encode a broad range of properties like part-of-speech [253], syntax [254], semantics [255], sentence length [256], etc., and to compare different language models in those properties [257]. Probing has also been adopted to understand multimodal representations in terms of the capacity for instance retrieval [231], inter-modality knowledge [258], understanding of verbs [259], entity and syntactic grounding [260], and visual commonsense knowledge [232], etc. However, probing has not been widely explored for visual representations, despite as a fast on-the-fly metric for model evaluation [261, 237, 238] complementary to fine-tuning. To our knowledge, we are the first to compare VL models and vision-only models using probing.

Task	Dataset	# of classes	Metric	Prediction head
object name prediction	Visual Genome [79]	151	accuracy	linear classifier on ROI features
attribute prediction	VAW [234]	620	mAP	linear classifier on ROI features
part semantic segmentation	PartImageNet [236]	40	mIOU	head from Segmenter [262]
object detection	MSCOCO [235]	80	mAP	head from VitDet [263]
instance segmentation	MSCOCO [235]	80	mAP	head from VitDet [263]

Table 7.1: The details of {dataset, number of classes, metric, prediction head} for the five probing tasks.

7.3 Method

To analyze the capacity of the learned representations of different models, we choose a set of tasks to probe the models. For each task, we first extract features using the pretrained models, then we train a simple standard head to predict the results. Mathematically, for every image $I \in \mathbb{R}^{3\times w\times h}$, we extract its features $f \in \mathbb{R}^{C\times W\times H}$ using the off-the-shelf visual encoders in the pretrained models. Here (w, h) is the size of the input image and (C, W, H) is the size of the feature. Then a prediction head P is trained to predict the task-specific results based on feature f. In the whole process, only the head P is trained while the pretrained model (*i.e.*, feature extractor) is frozen.

In this section, we will first describe the probing tasks, datasets and the prediction head for each task (Sec. 7.3.1), then we describe the evaluated models (Sec. 7.3.2), and finally how to make the comparison settings fair for every model (Sec. 7.3.3).

7.3.1 Probing tasks and datasets

We choose five probing tasks: object name prediction, attribute prediction, object detection, instance segmentation and semantic segmentation for object

parts. Among the five tasks, object name and attribute prediction focus more on predicting the semantic labels, while the others are dense prediction tasks that highly rely on spatial information.

Object name prediction. Understanding object names is critical in various multi-modal downstream tasks like VQA and image captioning, in which text descriptions refer to objects by their names. Given an image and a bounding box, object name prediction requires predicting the name of the object in the box. We use the Visual Genome dataset [79] for training and evaluation in this task. Images in Visual Genome mostly come from MSCOCO [235] and contain multiple objects. For each object, the annotations provide its bounding box, name and attributes (color, material, etc.). The annotations cover 151 object classes for 1.3M objects in 108k images.

A simple linear classifier is used to predict object names. More specifically, for each object, we first use ROI-Pooling [108] to average pool the features according to its box, then use a linear layer on top of the pooled features to predict the name class of the object. Cross entropy loss is used to train the head. Note that the ground-truth bounding box coordinates are provided to the head for both training and testing.

Object attribute prediction. Similar to object name prediction, attribute prediction requires predicting attributes for the object in the given bounding box. As shown in [76], visual features with better-encoded attribute information can substantially improve the performance of multi-modal tasks. This motivates us to treat the attribute as an important axis for evaluating visual representation. The VAW dataset [234] is used for object attribute prediction.

VAW improves the noisy attribute annotations in Visual Genome. VAW annotates 620 attributes belonging to 8 categories, including color, shape, size, material, texture, action, state, and others. Every attribute is annotated as positive, negative, or unknown for each instance. The annotation covers 260k instances from 72k images, which is a subset of Visual Genome images. Mean average precision (mAP) is used to evaluate the prediction results following [234].

Since attribute prediction is formulated as a multi-label classification problem, the prediction head is similar to object name prediction, but has several differences. First, binary cross entropy loss is used for training instead of cross entropy. Second, since the attributes naturally come with a long-tailed distribution, to prevent the rare attributes (*e.g.*, playing) from being overriden by the frequent ones (*e.g.*, black), we assign higher weights to rare attributes and lower weights to frequent ones. Third, for the attributes labeled as unknown, we treat them as negative labels with a small (0.001) weight. Those strategies are borrowed from [234].

Object detection and instance segmentation. While object name/attribute prediction tests the ability to predict class labels when the object bounding box is given, we are also interested in tasks that focus more on locating the objects. We choose object detection and instance segmentation on MSCOCO [235] for this purpose. MSCOCO contains 330K images with 1.5 million object instances in 80 categories. The bounding box and segmentation mask are annotated for each instance. mAP, *i.e.*, mean of average precision for each category, is adopted as the evaluation metric.

Because detection and segmentation cannot be completed using a simple head like a linear layer, we adopt the prediction head in VitDet [263] as our probing head. While the widely used Mask-RCNN is based on convolutional neural network (CNN) features, [263] propose a variant that is more suitable for non-hierarchical transformer features. Considering the fact that most of our evaluated models are transformer-based, we adopt this VitDet head for probing in our work. Unless specified, all the experiment settings are kept the same as [263].

Part semantic segmentation. While image classification accuracy on ImageNet dataset [229] is the most commonly used metric for evaluating visual representations, the recent PartImageNet dataset [236] provides additional annotations for the ImageNet images, thus enables finer-grained evaluation. PartImageNet annotates segmentation masks of 40 object parts (*e.g.*, head, body, tail) for 11 categories of objects on 24k images. Using this dataset, we perform semantic segmentation of object parts as an additional probing task that requires localization information.

For the segmentation head, we use the mask transformer decoder in Segmenter [262] due to its simplicity and impressive performance on standard datasets. [262] adapts transformers for semantic segmentation with the proposed "mask transformer decoder" on top of the embeddings produced by the transformer encoder (standard ViT). In our probing, we replace their transformer encoder with the pretrained models to be evaluated and train the mask transformer decoder to output the semantic segmentation map. Because our

goal is to fairly compare different models instead of achieving high performance, we reduce the input image size (from 1024×1024 to 224×224). A linear layer is used to match the feature's dimensions and bilinear upsampling is used to match feature's spatial sizes. All the other training settings are kept the same.

7.3.2 Evaluated models

We evaluate five models: three representative VL models including CLIP, OFA and FLAVA, and two vision-only models including MAE and MOCOv3. Among the five models, CLIP and MOCOv3 are trained using contrastive loss, while the others are trained with sequence modeling losses. We choose these models because they are representative and highly popular, and their pretrained weights and code are publicly available. In the following, we describe the models, especially their visual components, and how we extract features from them.

CLIP [225]. CLIP is a dual encoder model trained with contrastive loss using 400M image-text pairs. The image embeddings produced by the image encoder, which can be either a ResNet or a transformer, and the text embeddings produced by the text encoder are trained to be closer with each other in the embedding space when the image and text pair matches. The learned image embeddings are shown to have superior transferability on various downstream tasks. In our study, image features are extracted using the pretrained image encoder.

OFA [227]. OFA is a unified model that targets both uni-modal and multi-modal tasks. The vision tasks (image classification and object detection), language tasks, and multi-modal tasks (VQA, region/image captioning, visual grounding) are all formulated into a sequence-to-sequence generation problem. In particular, special visual tokens from discrete-VAE [264, 265] are used for image infilling and the object bounding box coordinates are also discretized into special tokens. The OFA model first uses a ResNet (Res101 for OFA_{base}) to encode images, then use the transformer encoder and decoder to generate the target sequence from image and text features. Cross entropy loss is used as supervision. OFA is pretrained using 20M image-text pairs with additional uni-modal data. To obtain visual representations, we feed the model with only the image (*i.e.*, empty text), send it through the ResNet, and take the output of the transformer encoder.

FLAVA [228]. FLAVA is a fully transformer-based unified model. Similar to OFA, the model solves both uni-modal and multi-modal tasks. However, the differences lie in (a) tasks, (b) model architecture, and (c) training loss. (a) FLAVA does not have bounding boxes in the vocabulary, and thus does not support box-related tasks like object detection, visual grounding or region captioning. (b) FLAVA is fully based on transformers; it uses two separate transformer encoders to encode images and texts, then uses several more transformer layers for multi-modal fusion. (c) FLAVA takes multiple losses including CLIP-like contrastive loss, masked image/text/multi-modal modeling losses, and image-text matching loss. FLAVA is pretrained on 70M image and text pairs. We take the output of the visual transformer encoder as image

representations.

MAE [237]. Masked Auto-Encoder (MAE) is a self-supervised vision model trained with a masked image modeling task. MAE encodes masked image patches with a transformer encoder and reconstructs the missing pixels with a lightweight decoder trained with MSE loss. Unlike OFA and FLAVA, the reconstruction for MAE happens in the continuous pixel space, which does not require dVAE to generate discretized image tokens. MAE is trained only with ImageNet-1k data and shows promising transfer performance to downstream tasks.

MOCOv3 [238]. We choose MOCOv3 to represent self-supervised vision transformers trained with contrastive loss. During training, two crops for each image under random data augmentation are encoded by two encoders, a key encoder and a query encoder, into two vectors named "key" and "query" respectively. During training, the goal is to retrieve the corresponding "key" by the "query". Similar to MAE, MOCOv3 is trained using ImageNet-1k.

7.3.3 Comparison settings

To make the comparison fair, we carefully choose the model size and input size, and ensure different methods are comparable. As probing tasks are highly sensitive to image size and feature's spatial size, for all the models on all the tasks, we fix the input image resolution to be 224*224. We choose this size because 224*224 is the input size for pretraining for all the models except OFA (OFA is pretrained with size 384 for *base* version and 480 for *large*). For dense tasks, although the original detection and segmentation models (*i.e.*,

VitDet and Segmenter) use larger input image sizes for better performance, we unify the input size because our goal is to fairly compare models, rather than achieving the best performance.

We find the probing results sensitive to the models' input patch size, because different patch sizes produces features with different spatial sizes.[1] Therefore, considering the availability of pretrained checkpoints with different model sizes and input patch sizes, we try our best to align the feature size and evaluate with the ViT-B/16 backbone by default. Because OFA is not purely transformer-based, we evaluate on the *base* size, which has a ResNet + transformer encoder with 120M parameters (comparable to the 86M ViT-B/16). More details of the evaluated models are shown in **??**.

7.4 Experiments

7.4.1 Implementation details

For object name and attribute prediction, the models are trained with a learning rate of 0.001 and batch size of 64 for 200 epochs. We adopt early stopping based on validation performance, then report performance on the test split using the best model. For object detection and segmentation on the COCO dataset, the model is trained for 120k iterations with batch size 20. The learning rate is first set to 8e-5, then decay twice at step 100k and 115k with a factor of 0.1. For part segmentation, we train the model with a learning rate of 0.01 and batch size of 128 for 200 epochs. The validation performance for the final

[1] *E.g.* for input images of 224*224, ViT-B/16 produces visual representations with size 768*14*14, while ViT-B/14 gives feature size 768*16*16, which will affect probing.

	Task	VG Obj.	VAW Attr.	COCO Det.	COCO Seg.	Part Seg.	IN1k ft.	IN1k probe
V+L	**OFA**	**57.13**	**61.67**	25.04	19.38	33.11	82.2	-
	FLAVA	54.29	61.51	21.06	17.20	34.77	-	75.5
	CLIP	51.54	61.15	19.55	15.56	40.61	-	**80.2**
V	**MAE**	49.52	52.59	**25.29**	**22.05**	**42.30**	**83.6**	68.0
	MOCOv3	47.81	54.44	20.31	16.96	40.11	83.2	76.7

Table 7.2: Probing results on the five tasks. VL models perform better on label prediction tasks, while vision-only models perform better on dense prediction tasks. Finetuning and linear probing results on ImageNet for each model (cited from original papers) are also shown for reference.

		MSCOCO			PartImageNet		
		mAP	Semantic	Localization	mIOU	Semantic	Localization
V+L	**OFA**	19.38	60.02	17.41	33.11	71.71	84.15
	FLAVA	17.20	61.48	14.67	34.77	75.28	83.76
	CLIP	15.56	**68.24**	13.25	40.61	**80.21**	86.80
V	**MAE**	**22.05**	46.85	**20.69**	**42.30**	75.03	**89.50**
	MOCOv3	16.96	49.80	15.08	40.11	76.18	86.08

Table 7.3: Detailed analysis of instance segmentation and part segmentation results. We evaluate the segmentation results (standard metric mAP, mIOU) from two additional perspectives: semantics (F1 score for semantic class prediction) and localization (mAP/mIOU for foreground/background segmentation). While V models are better on the standard metrics, VL models are better when evaluated with semantics metrics.

checkpoint is reported.

7.4.2 Probing results

We probe the five models on each of the five probing tasks. We make sure that the experiment settings, including model size, input size, training protocol and data splits, are well aligned for every model in order to make fair comparisons. The probing results are shown in Tab. 7.2. We also include the ImageNet finetuning accuracy and linear probing accuracy of each model for reference, because they are widely-used metrics for model evaluation. On each task, we compare the VL models and V models. Note that the evaluation metric for

each task is different (as in Tab. 7.1), performance on different tasks cannot be compared and we only compare numbers in each column separately.

For object name prediction and attribute prediction, VL models consistently perform better than V models. For object name prediction on Visual Genome, VL models all achieve more than 51% accuracy while V models get accuracy less than 50%; for attribute prediction on VAW, mAP for VL models are higher than 61% while lower than 55% for V models. This suggests that representations from VL models capture richer semantic information about the objects in each image, which can be decoded using a simple linear layer. In contrast, in V models the name and attribute information are not explicit enough.

For the dense prediction tasks, MAE performs the best on all three tasks. For part semantic segmentation on PartImageNet, MOCOv3 and CLIP also get decent performance ($> 40\%$) that is close to MAE (42%), while the other two VL models are lower by a large margin ($< 35\%$). For object detection on MSCOCO, OFA gets close mAP (25.0) to MAE (25.3) while the performance of the other three models are much lower; however, when it comes to instance segmentation, the advantage of MAE is more clear, surpassing all the other models with a margin larger than 2.7%.

Interestingly, comparing the object detection and instance segmentation results on COCO, we find that the performance drops of V models are consistently smaller than VL models, which indicates that V models learn better localized representations. For example, for OFA, the mIOU for segmentation is 5.7% (25.04-19.38) lower than that for detection; while the drop MAE and

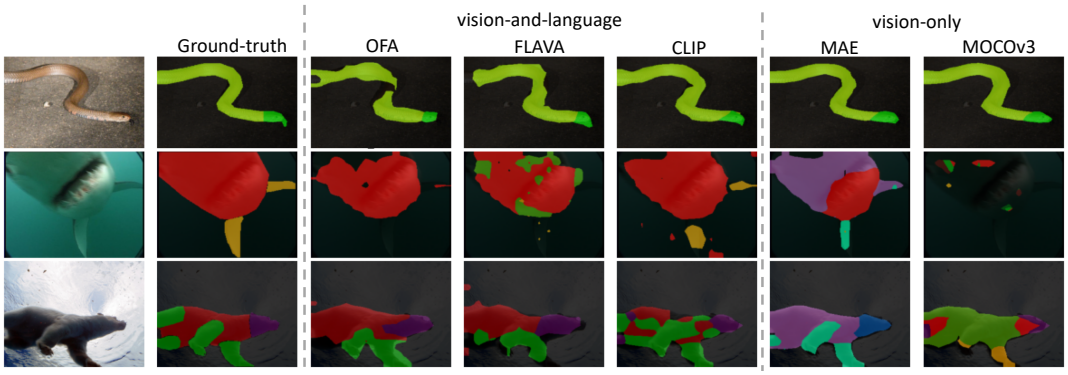


Figure 7.2: Compared to vision-and-language models, vision-only models more accurately predict the boundary of segmentation masks, but make mistakes in labeling the regions.

MOCOv3 are smaller (3.2%, 3.3%). Because segmentation requires more localized features than detection to find the boundary of objects, the performance gap between detection and segmentation can be an indicator of the localized information in the representations, considering those two tasks are based on the same dataset. With the more-localized representations, the model can better predict the mask boundary. Therefore, the smaller gap of vision-only models suggests they learn more localized representations.

To further verify this finding, we next take a closer look into segmentation results, which more clearly compare the semantics and localization information in different models.

A closer look at the segmentation results. We evaluate the instance segmentation results on COCO and semantic segmentation results on PartImageNet using two more metrics: (a) the label prediction metric, and (b) the foreground-background segmentation metric, where (a) is an indicator for semantics and (b) for localization. The motivation is that the segmentation

metrics (mAP for instance segmentation, mIOU for semantic segmentation) require correctly predicting both the class label and the boundary, so the quality of both determines the score. Therefore, we propose two additional metrics to measure the two factors separately. For (a), for each image, we transform its predicted segmentation map into label predictions, and evaluate the quality using the multi-label prediction metric. In particular, we treat the appeared classes in the segmentation map as positive labels and the others as negative; then the label predictions are evaluated using the F1 score. F1 score is defined as $\frac{2*\text{precision}*\text{recall}}{\text{precision}+\text{recall}}$, where precision and recall are averaged over label classes. For (b), we merge all the different object categories and process the segmentation map into binary labels, *i.e.*, foreground and background, then report the mIOU (for instance segmentation) or mAP (for semantic segmentation) of the binary segmentation maps.

Tab. 7.3 shows the segmentation results on COCO and PartImageNet evaluated using the above two metrics. Although MAE achieves the best performance on both datasets, when looking at the semantic and localization results, we find that its advantage mainly comes from better localization, rather than semantics. In terms of semantics, VL models perform much better than MAE. For example, on the MSCOCO dataset, VL models achieve F1 scores higher than 60, while MAE and MOCOv3 are lower than 50. The results suggest that while MAE is better at finding the object boundaries when predicting segmentation masks, VL models are better at predicting labels for the objects.

In Fig. 7.2, we show several examples of the part segmentation results

on PartImageNet. In the examples, MAE captures the object's shape more accurately, like the curly snake body, the shark's small fin, and the quadruped contour. However, MAE and MOCOv3 make more mistakes in labeling the regions compared to VL models. For example, MAE wrongly predicts the shark fin as a reptile foot, and the quadruped as a reptile; MOCOv3 confuses the quadruped head and foot as the fish head and fins. Those examples more explicitly compare the semantics and localization knowledge learned by VL and V models.

Analysis on different attribute groups. We further decompose the attribute prediction results into different attribute groups. In the VAW dataset, attributes are categorized into 8 groups: action, texture, shape, size, color, material, state, and others. The results are shown in Fig. 7.3. Interestingly, despite the overall better results of VL models, we find that their advantages differ in different groups. For example, the gap between VL and V models in the "action" category is more significant than in the "texture" category. Intuitively, "action" is less visually grounded then "texture" requires more context and semantic information, on which VL models is better at, suggesting that while vision-only ones are better at predicting highly visually grounded local attributes (*e.g.*, texture), VL models are better at more abstract ones.

7.4.3 More analysis

Findings of contrastive training. The results also show that contrastive models perform relatively better on localization for single-object images than

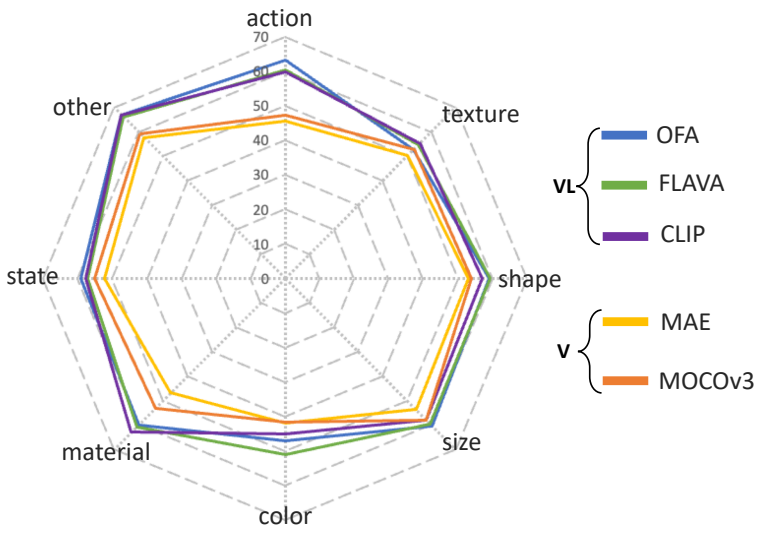


Figure 7.3: A closer look at the attribute prediction results by separately evaluating different types of attributes. The advantage of VL models is more significant in the more abstract categories (*e.g.*, *action*) than visually grounded categories (*e.g.*, *texture*).

multi-object images. Among the five tasks, part segmentation on PartImageNet dataset are based on single-object images from ImageNet, while the other four tasks are based on COCO-style multi-object images. In Tab. 7.3, comparing the contrastively trained models (CLIP, MOCOv3) and the models trained with sequence modeling objectives (OFA, FLAVA, MAE), we find that contrastive models perform relatively better on PartImageNet than MSCOCO. For example, on PartImageNet, CLIP outperforms the other two VL models (*i.e.*, OFA and FLAVA) by a large margin (more than 6% mIOU); on MSCOCO, it under-performs them. The semantic and localization evaluation suggests that this difference is mainly caused by localization, *e.g.*, the localization results of CLIP is much better than OFA and FLAVA on PartImageNet. A similar observation can be obtained by comparing MOCOv3 and MAE: although MOCOv3 underperforms MAE on both datasets, the gap is much smaller on

PartImageNet than MSCOCO (2.2 vs. 5.1). Therefore, we suggest that the localization ability of contrastive models is relatively stronger on single-object images.

The effect of model size. To study the effect of model size, in Tab. 7.4, we show the probing results with size *base* and *large* for MAE and OFA. For MAE, a larger model size improves performance on all the probing tasks in parallel for 1% to 2%. However, note that this improvement is less significant compared to the big gaps between different model types. For OFA, except for the marginal improvement in attribute prediction, the larger model size hurts probing results on the other four tasks. The reason for the decrease is that the OFA_{large} is pretrained with a larger input image size (480*480) compared with OFA_{base} model (384*384). Because we probe all models with the same image size (224*224) for a fair comparison, the gap in image size between pretraining and probing is more significant for OFA_{large}. In summary, the effect of model size is less considerable than other factors like model type or input image size.

	obj.	**attr.**	**det.**	**seg.**	**p-seg.**
$\textbf{MAE}_{base}$	49.52	52.59	25.29	22.05	42.30
$\textbf{MAE}_{large}$	**51.91**	**53.38**	**29.67**	**25.63**	**44.85**
$\textbf{OFA}_{base}$	**57.13**	61.67	**25.04**	**19.38**	**33.11**
$\textbf{OFA}_{large}$	52.33	**62.01**	21.23	16.51	32.04

Table 7.4: The influence of model size is less considerable than other factors like model type.

The effect of downstream finetuning. Tab. 7.5 compares probing results of models with and without finetuning on downstream tasks. For MAE, the results are based on the *base* size; for OFA, the results are on *large* size, due

to the availability of publicly released model checkpoints. For both models, finetuning on image classification on ImageNet-1k and VQA on VQAv2 hurts the probing performance to varying degrees (except for attribute prediction). This indicates that while in pretraining, the model learns features that capture various fine-grained information about the image, during finetuning towards a specific task, only information useful for the task is kept and other information is dropped. Moreover, compared with ImageNet finetuning, finetuning on VQA leads to a much smaller performance decrease in probing results, suggesting that the change in probing results depends on the nature of downstream tasks. In this case, VQA requires more fine-grained information about objects, attributes, etc., resulting in a smaller drop than ImageNet finetuning.

	obj.	**attr.**	**det.**	**seg.**	**p-seg.**
MAE	**49.52**	52.59	**25.29**	**22.05**	**42.30**
MAE$_{IN1k}$	45.16	**53.82**	21.41	17.74	35.62
OFA	**52.33**	62.01	**21.23**	**16.51**	**32.04**
OFA$_{IN1k}$	50.54	60.74	18.91	14.67	27.56
OFA$_{VQA}$	51.42	**63.40**	19.01	14.22	28.34

Table 7.5: Probing results of models finetuned on downstream tasks. Finetuning hurts the probing performance in most cases.

7.5 Conclusion

This work compares the visual representations in multimodal and unimodal models by feature probing. By comparing three representative VL models and two V models on five probing tasks, we find that VL models are stronger in label prediction tasks, while vision-only models are better in dense prediction

tasks. We hope our diagnostic findings serve as an empirical guidance for future works in choosing models for different downstream tasks, as well as exploring the role of language in visual representation learning.

Part III

Image Captioning with Contrastive Reasoning

Chapter 8

Context-Aware Group Captioning via Self-Attention and Contrastive Features

While image captioning has progressed rapidly, existing works focus mainly on describing single images. In this chapter, I introduce a new task, context-aware group captioning, which aims to describe a group of target images in the context of another group of related reference images. Context-aware group captioning requires not only summarizing information from both the target and reference image group but also contrasting between them. To solve this problem, I propose a framework combining self-attention mechanism with contrastive feature construction to effectively summarize common information from each image group while capturing discriminative information between them. To build the dataset for this task, I propose to group the images and generate the group captions based on single image captions using scene graphs matching. Our datasets are constructed on top of the public Conceptual Captions dataset and the new Stock Captions dataset. Experiments on the two

datasets show the effectiveness of our method on this new task. [1]

8.1 Introduction

Figure 8.1: Context-ware group captioning. Given a group of target images (shown in orange boxes) and a group of reference images which provide the context (`woman`), the goal is to generate a language description (`woman with cowboy hat`) that best describes the target group while taking into account the context depicted by the reference group.

Generating natural language descriptions from images, the task commonly known as image captioning, has long been an important problem in computer vision research [266, 267, 268]. It requires a high level of understanding from both language and vision. Image captioning has attracted a lot of research attention in recent years thanks to the advances in joint language-vision understanding models [50, 269, 270, 135]. While image captioning has progressed rapidly, existing works mostly focus on describing individual images. There

[1] Related Datasets and code are released at https://lizw14.github.io/project/groupcap.

are practical scenarios in which captioning images in group is desirable. Examples include summarizing personal photo albums for social sharing or understanding web user intention from their viewed or clicked images. Moreover, it is often the case that the target image group to be captioned naturally belongs to a larger set that provides the context. For instance, in text-based image retrieval applications, given a group of user-interested images and other images returned by the search engine, we could predict the user hidden preferences by contrasting the two groups and suggest a new search query accordingly. Figure 8.1 shows an example of such scenario. Among all the images returned by search query `woman`, the user can indicate his/her interest in some of the images (in orange boxes). The objective is to recognize that the user wants `woman with cowboy hat` and suggest the query accordingly.

Inspired by these real-world applications, we propose the novel problem of *context-aware group captioning*: given a group of target images and a group of reference images, our goal is to generate a language description that best describes the target group in the context of the reference group. Compared to the conventional setting of single-image based captioning, our new problem poses two fundamental requirements. First, the captioning model needs to effectively summarize the common properties of the image groups. Second, the model needs to accurately describe the distinguishing content in the target images compared to the reference images.

To address those requirements, we develop a learning-based framework for context-aware image group captioning based on self-attention and contrastive feature construction. To obtain the feature that effectively summarizes

the visual information from the image group, we develop a group-wise feature aggregation module based on self-attention. To effectively leverage the contrastive information between the target image group and the reference images, we model the context information as the aggregated feature from the whole set and subtract it from each image group feature to explicitly encourage the resulting feature to capture the differentiating properties between the target image group and the reference image group.

Training our models requires a large number of image groups with text descriptions and associated reference image sets. In this chapter, we leverage large-scale image caption datasets to construct the training data. In particular, we build our annotations on top of Conceptual Captions [271], a recently introduced large-scale image captioning dataset. We parse the single-image caption into scene graphs and use the shared scene graphs of image groups to generate the groups' ground-truth captions. In addition, we apply the same procedure on a large-scale image set collected from a photograph collection. This dataset contains a large number of images with compact and precise human-generated per-image descriptions. That results in our second dataset, *Stock Captions*, which we plan to contribute to the research community to encourage future research in this new problem.

The main contributions in this chapter are three-fold. First, we introduce the problem of context-aware group captioning. This novel image captioning setting can potentially be important for many real-world applications such as automatic query suggestion in image retrieval. Second, we present a learning-based approach which learns to aggregate image group visual feature for

caption generation. Our framework combines the self-attention mechanism with contrastive feature construction to effectively encode the image group into a context-aware feature representation, which effectively summarizes relevant common information in the groups while capturing discriminative information between the target and context group. Third, we introduce two large-scale datasets specifically for the context-aware group captioning problem. Experiments on the two datasets demonstrate that our model consistently outperforms various baselines on the context-based image group captioning task.

8.2 Related Work

Image captioning has emerged as an important research topic with a rich literature in computer vision [266, 267, 268]. With the advances in deep neural networks, state-of-the-art image captioning approaches [50, 272, 273, 269, 274, 270, 239, 275] are based on the combination of convolutional neural networks [276] and recurrent neural networks [102] (CNN-RNN) architecture, where the visual features are extracted from the input image using CNNs which is then decoded by RNNs to generate the language caption describing the given image. Research in image captioning has progressed rapidly in recent years. Novel network architectures [50, 277, 137, 278], loss functions [279, 280, 281, 282, 270, 283], and advanced joint language-vision modeling techniques [284, 285, 137, 135, 286, 287] have been developed to enable more diverse and discriminative captioning results. Recent works have also proposed to leverage the contextual and contrastive information from

additional images to help generating more distinctive caption for the target image [288, 289, 290, 291, 292] or comparative descriptions between image pairs [293, 294, 295, 296]. Existing works, however, mostly focus on generating captions for a single image. Our work, on the other hand, focuses on the novel setting of context-based image group captioning which aims to describe a target image group while leveraging the context of a larger pool of reference images.

Referring expression generation [297, 298, 299] is a related problem to image captioning, which aims to generate natural language descriptions for a target object in an image. Contrastive modeling has been successfully applied in state-of-the-art referring expression generation methods to describe the target image region in contrast with other image regions. Yu *et al*. [300] use relative location and feature difference to discriminate the target object. Mao *et al*. [301] maximize the probability of generated expression describing a specific region over other regions by Maximum Mutual Information training. While referring expression generation considers the target region in contrast with each negative region respectively, our problem requires contrastive context modeling among and between image groups.

Attention mechanism has been successful in image captioning [277, 136, 137, 135, 287]. These works focused on applying visual attention to different spatial regions at each text generation time step. More recently, attention in transformer[24] and pretrained BERT[302] has been very successful in natural language processing tasks. [303, 75, 304] adapts the idea of BERT to vision and language tasks and showed improved performance on multiple

sub-tasks. [305] bridges attention and non-local operator to capture long-range dependency, which has been used in many computer vision tasks [306, 307, 308, 309]. In our work, we apply attention over a group of images and show its effectiveness for summarizing information in an image group.

Our setting is inspired by **query suggestion** [310, 311, 312, 313] in the context of document retrieval systems. Query suggestion aims to predict the expanded query given the previous query used by the users while taking into account additional context such as search history [310, 311, 312] or user interaction (e.g. clicked and skipped documents) [313]. We are inspired by this task formulation and extend it to vision domain. Earlier works on query suggestion in image search focus on forming visual descriptors to help obtain better search results [314, 315] while the suggested text query is obtained solely from the current user query without taking visual content understanding into account. Our work, on the other hand, can potentially be applied to enable query suggestion from images. In this work, we focus on the image captioning aspect without relying on modeling user information and behavior as in existing query suggestion works, thus making it applicable beyond retrieval tasks.

8.3 Dataset

To train our models, we need a large-scale dataset where each data sample contains a group of target images with an associated ground-truth description and a larger group of reference images. The reference images need to be relevant to target images while containing a larger variety of visual contents

and thus provides context for describing target images. The description should be both specific to the target group and conditioned on the reference group.

In this section, we first describe the intuition and method for dataset creation, then provide details of our proposed datasets built on the Conceptual Captions dataset and our proposed Stock Captions dataset.

8.3.1 Data Construction Method

We build our dataset on top of large-scale per-image captioning datasets by leveraging the shared scene graphs among images, motivated by [289]. The overall data generation process is shown in Figure 8.2.

Form the target and reference image groups. Images with shared scene graphs compose an image group. More specifically, images with the same *(attribute)-object-relationship-(attribute)-object* are chosen to compose the target image group, while images with partially overlapping scene graphs with the target group are chosen as the reference image group. For example, as in Figure 8.2, images with the scene graph *woman in chair* are selected to form the target group, while images containing *woman* are selected to form the reference group paired with the target group. In this way, the reference group contains a larger variety of contents (woman in any places or poses) while the target group is more specific in terms of certain attributes (in chair).

Scene graph parsing for each image. In order to get the scene graphs for each image to support our grouping process, we use a pretrained language parser (improved upon [316]) to parse each ground-truth per-image caption into a scene graph. We choose to parse the scene graph from image captions

instead of using the annotated scene graph in Visual Genome dataset [79] because our scene graph needs to focus on the most "salient" content in the image. Since Visual Genome is densely annotated without the information of which object is the main content of the image, scene graphs of small trivial objects may dominate the grouping process while the main content is ignored. This will produce very noisy data, potentially unsuitable for training our models. On the other hand, while parsing errors may introduce noise, scene graphs parsed out of image captions focus on the main objects because the caption usually describes the most important contents in an image.

Generating the groundtruth caption for the image group. After getting the target and reference groups using scene graph matching, the shared scene graph among target images is flattened into text to serve as the ground truth group description. For example, in Figure 8.2, the ground-truth group caption is *woman in chair*. Other examples of ground-truth group captions include: *colorful bag on white background, girl in red, business team holding terrestrial globe, woman with cowboy hat, etc.*

Source datasets. To construct our datasets for group captioning, the per-image captioning datasets need to be large-scale to provide enough image groups. We build our group captioning datasets on top of two datasets: Conceptual Captions dataset [271], which is the largest existing public image captioning dataset, and Stock Captions dataset, which is our own large-scale per-image captioning dataset characterized by precise and compact descriptions. Details about construction on the two datasets are provided as follows.[2]

[2]For simplicity, in this paper, we call our newly constructed group captioning datasets by the same name as their parent datasets: Conceptual Captions, and Stock Captions.

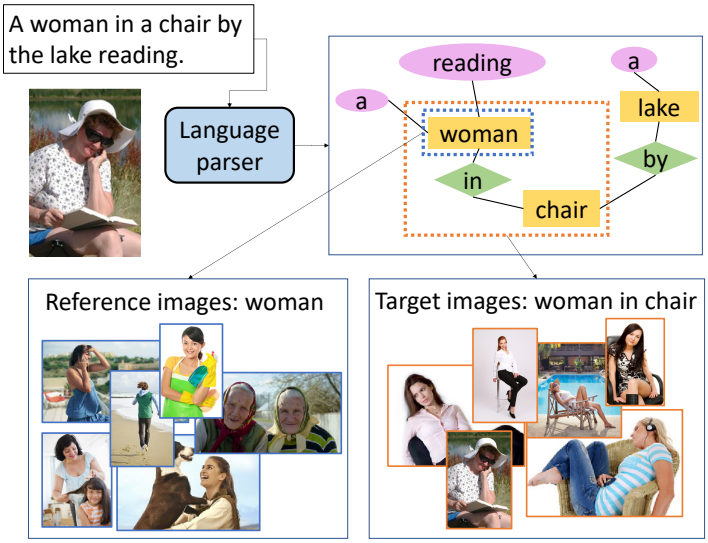


Figure 8.2: Dataset construction method. Our datasets are constructed from image collections with per-image descriptions. A pretrained language parser is used to parse each image caption into a scene graph. Then the images with shared scene graph are grouped to form the target group. Images with scene graphs that partially match the targets' form the reference group.

8.3.2 Conceptual Captions

Conceptual Captions is a large-scale image captioning dataset containing 3.3 million image-caption pairs. (By the time we download the images through the urls provided, only 2.8 million are valid.) Because the captions are automatically collected from alt-text enabled images on the web, some of the captions are noisy and not natural. However, the high diversity of image contents and large number of images makes Conceptual a suitable choice for data generation using our method.

After sampling from 2.7 million images from Conceptual Captions, we obtain around 200k samples with 1.6 million images included. Each sample

contains 5 target images and 15 reference images. The images with rare scene graphs that cannot be made into groups are not used. We manually cleaned the sampled data to remove samples that are not meaningful. For example, target group of *portrait or woman* and reference group of *woman* are not semantically different so they are removed. We also cleaned the vocabulary to remove rare words.

The 200k samples are split into test, validation and train splits, where these three splits share the same image pool. While the validation and train splits may contain samples of same group captions (because group captions are usually short), we make sure that captions in test split do not overlap with train split. More detailed statistics are provided in Table 8.1.

Original Per-Image Captioning		
	Conceptual	Stock
Size	2766614	5785034
Avg Length	9.43	4.12
Context-aware Group Captioning		
	Conceptual	Stock
Size	199442	146339
Train Split	175896	117829
Val Split	10000	10000
Test Split	13546	18510
# of images	1634523	1941370
Vocab Size	5811	2437
Avg Length	3.74	2.96

Table 8.1: Statistics of Conceptual Captions and Stock Captions, in terms of original per-image captioning dataset and our group captioning dataset constructed on top of per-image captions.

8.3.3 Stock Captions

While the Conceptual dataset excels in image diversity, we found that its captions are often long and sometime noisy. Motivated by the query suggestion application where the suggested search queries are usually short and compact, we propose to construct the dataset on a new image captioning dataset named Stock Captions. Stock Captions is a large-scale image captioning dataset collected in text-to-image retrieval setting. Stock Captions dataset is characterized by very precise, short and compact phrases. Many of the captions in this dataset are more attribute-like short image titles, e.g. "colorful bags", "happy couple on a beach", "Spaghetti with dried chilli and bacon", etc.

After grouping and filtering the 5.8 million raw images, we get 1.9 million images, grouped into 1.5 million data samples for the Stock Captions dataset. The dataset sampling and split details are similar to Conceptual.(Table 8.1).

8.3.4 User Study for Dataset Comparisons

To test the quality of our data and compare our two datasets, we conduct a user study by randomly selecting 500 data samples (250 from each dataset) and ask 25 users to give a 0-5 score for each sample, with 5 being the highest and 0 being the lowest.

To better compare the two datasets, we ask the users to give strict scores. A caption needs to be precise, discriminative and natural to be considered good. Many captions with the score of 0 and 1 are semantically good, but are unnatural. The distribution of scores is shown in Figure 8.3. As expected, in overall quality, the Stock Captions data scores significantly higher as it is

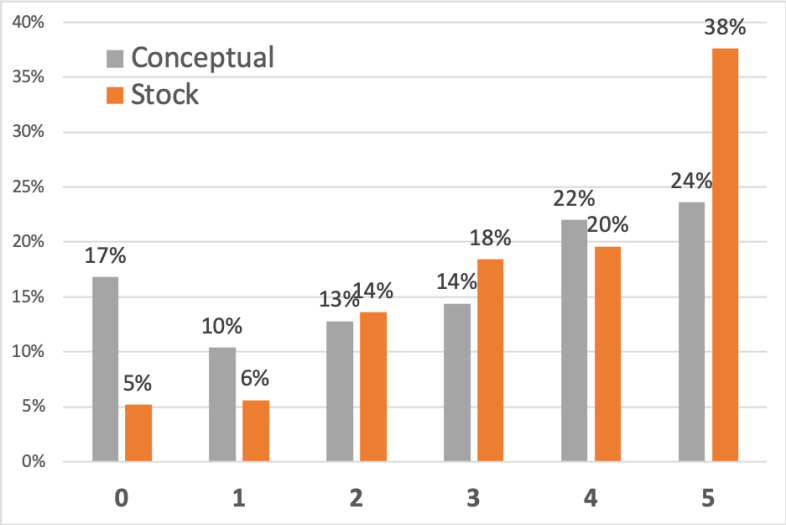


Figure 8.3: Distribution of human-given scores for our two constructed datasets. Dataset constructed on Stock Captions gets higher human scores.

based on compact and precise human-generated captions. However, several users do note that the captions in the Conceptual Captions dataset seems to be more specific, and "interesting".

8.4 Method

In this section, we explore methods to address the two main challenges in our proposed problem: a) **feature aggregation**, *i.e.* how to summarize the images within one image group, and (b) **group contrasting**, *i.e.*, how to figure out the difference between two image groups. By comparing different methods, our goal is not only finding the best performing models, but also drawing insights into the characteristics of the task, and hopefully, setting the focus for future exploration in this problem.

To begin the section, we first formalize the problem settings in Section 8.4.1.

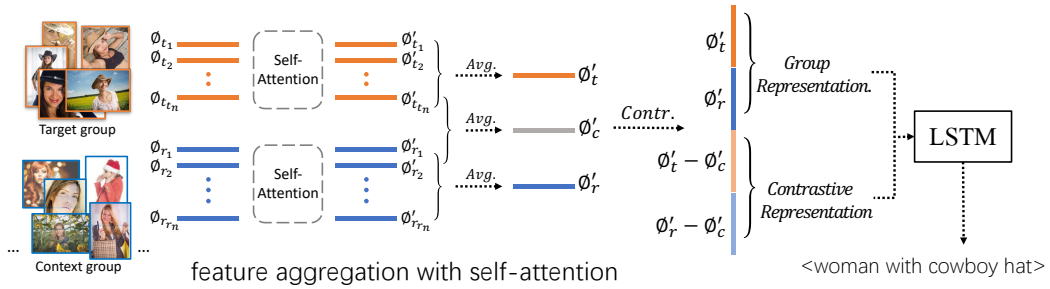


Figure 8.4: Context-aware group captioning with self-attention and contrastive features. Image features are aggregated with self-attention to get the group representation for each image group. Then the group representation is concatenated with contrastive representation to compose the input to LSTM decoder, which finally generates context-aware caption for the target image group.

In the subsequent sub-sections, we describe our method explorations path starting with a simple baseline. We then gradually introduce more computationally specialized modules. For each module, we describe our intuition and back them up with quantitative results and visual illustrations.

8.4.1 Problem Setting

Given a group of n_t target images and a group of n_r reference images, our task is to generate a description $D =< \hat{w}_1, ..., \hat{w}_l >$ to describe the target image group in context of the reference group. Here $\hat{w}_i$ denotes the word in the sentence and l is sentence length, which varies for each data sample. In our setting, $n_t = 5, n_r = 15$.

Each image is represented by a 2048-d feature extracted using the ResNet50 network [241] (after pool5 layer), pretrained on ImageNet [229]. The input of our model are the target features $\Phi_t = [\phi_t^1, ..., \phi_t^{n_t}]$ and the reference features $\Phi_r = [\phi_r^1, ..., \phi_r^{n_r}]$, where $\phi^i \in \mathbb{R}^{2048}$. We use Φ to denote a list of features,

while a single feature is denoted as ϕ.

While we believe that more detailed features (e.g. spatial features without mean-pooling, or object-level features) may improve performance, they increase the computational complexity, and by extension, the training time to an unacceptably high level in our initial testing. Thus, we simply use the mean-pooled feature vector.

8.4.2 Baseline: feature averaging and concatenation

From the problem setting above, one intuitive approach would be to summarize the target and reference features by averaging, and concatenating them to create the final feature for description generation. The process can be formalized as follows.

We compute the target group feature ϕ'_t and the reference group feature ϕ'_r by averaging the features in each group:

$$\phi'_t = \frac{1}{n_t} \sum_{i \in 1..n_t} \phi_{t_i} \qquad \phi'_r = \frac{1}{n_r} \sum_{i \in 1..n_r} \phi_{t_r}$$

Following standard captioning pipeline, we then use the concatenation of the two group features as input to LSTM to predict the context-aware descriptions. We use LSTM-RNN [102] to generate the caption in an auto-regressive manner. Denoting the output of the LSTM module at time step t as h_t, we have the equations for decoding:

$$h_1 = [\phi'_t, \phi'_r]$$

$$h_t = \text{LSTM}(h_{t-1}, \hat{w}_{t-1})$$

$$\hat{w}_t \sim \text{softmax}(h_t).$$

Finally, we follow standard beam search process to generate the captions. This decoding architecture is used in all of our subsequent model variants.

8.4.3 Feature aggregation with self attention

While the average-pooling method used for feature aggregation above is intuitive, it treats all image features equally. We note that many groups of images have prominent members that encapsulate the joint information of the whole groups (Figure 8.5). We argue that the group summarizing process could be improved if we can identify these prominent features/images. Motivated by that observation, we propose to use the transformer architecture [24] for this task. The transformer relies on a grid of attention between the elements of the set to learn a better representation. Intuitively, by learning the self-attention grid, the model can detect the prominent features as each element in the set can "vote" for the importance of the other elements through the attention mechanism. In the subsequent analysis, we show that, in our task, the self-attention gird indeed puts a lot more weights to the prominent images. The core computations of our transformer-based architecture can be summarized as follows.[3]

The first step is calculating the contextualized features using self-attention

[3]We only describe the core computation steps of the self-attention due to space constraint and to improve clarity. More details can be found in the original paper [24]. We also release our implementation if accepted.

mechanism. Given the input features Φ; three different sets of features: queries Q, keys K and values V are calculated using a linear transformation:

$$Q = W^Q \Phi + b^Q$$

$$K = W^K \Phi + b^K$$

$$V = W^V \Phi + b^V$$

Then the self-attention grid is calculated by a scaled dot product between Q and K (the scaling factor d is the dimension of the vectors in Q and K). The self-attention layer uses this attention grid and the value matrix V to compute its outputs.[4]

$$\text{Attention}\,(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) V$$

The self-attention output is then coupled with the residual signal to create the contextualized features Φ'.

$$V' = V + \text{Attention}(Q, K, V)$$

$$\Phi' = V' + \max\left(0, V'W_1 + b_1\right) W_2 + b_2$$

From this point, we denote the process of transforming from the original features set Φ to the contextualized feature set Φ' as $\Phi' = F(\Phi)$. With this formulation, we have the contextualized set of features Φ'_t and Φ'_r:

$$\Phi'_t = F_{st}(\Phi_t) \qquad \Phi'_r = F_{sr}(\Phi_r)$$

We tried both sharing and not-sharing weights of F_{st} and F_{sr}, and found that

[4]We don't use the multi-head attention in this work, as in our preliminary experiments, the multi-head attention provides no performance gain compared to a single head.

sharing weights lead to slightly better performance. This is intuitive as the task of grouping target images are not different from the task of grouping reference images, and thus, the grouping model can share the same set of weights.

In our experiments, the self-attention architecture provides a significant boost in performance compared to the average-pooling variant.

8.4.4 Group contrasting with contrastive features

The second major challenge in our proposed problem is the image group contrasting. With the aforementioned self-attention mechanism, we have good representations for the target and reference groups. The most intuitive way to learn the difference between the two features is either concatenation (which is implemented in our baseline) or feature subtraction.

We argue that, to learn the difference between two groups of images, we first need to capture their similarity. Our hypothesis is that, when we identify the similarity between all the images, we can "remove" this similarity portion from the two features to deduce more discriminative representation. This process is formalized as follows.

The first step is learning the common information ϕ'_c between all the images. We do that by applying the same self-attention mechanism described above to all the images.

$$\Phi'_c = F_a([\Phi_t; \Phi_r])$$

$$\phi'_c = \frac{1}{n_t + n_r} \sum \Phi'_c$$

Then the joint information is "removed" from the group features ϕ'_t and ϕ'_r by subtraction to generate the contrastive/residual feature ϕ^d_t and ϕ^d_r.

$$\phi^d_t = \phi'_t - \phi'_c \qquad \phi^d_r = \phi'_r - \phi'_c$$

The contrastive features ϕ^d_t and ϕ^d_r are concatenated with the group features ϕ'_t and ϕ'_r, which are then fed into LSTM-RNN to generate captions. In our subsequent analysis, we show that the contrastive features indeed focus on the difference between two image groups.

8.5 Experiments

	WordAcc	CIDER	WER	BLEU1	BLEU2	METEOR	ROUGE
Conceptual							
Per-Img. Caption	5.4638	0.4671	2.6587	0.1267	0.0272	0.0868	0.1466
Average	36.7329	1.9591	1.6859	0.4932	0.2782	0.3956	0.4964
SA	37.9916	2.1446	1.6423	0.5175	0.3103	0.4224	0.5203
Average+Contrast	37.8450	2.0315	1.6534	0.5007	0.2935	0.4057	0.5027
SA+Contrast	**39.4496**	**2.2917**	**1.5806**	**0.5380**	**0.3313**	**0.4405**	**0.5352**
Stock							
Per-Img. Caption	5.8931	0.3889	1.8021	0.1445	0.0359	0.0975	0.1620
Average	37.9428	1.9034	1.1430	0.5334	0.2429	0.4042	0.5318
SA	39.2410	2.1023	1.0829	0.5537	0.2696	0.4243	0.5515
Average+Contrast	39.1985	2.0278	1.0956	0.5397	0.2632	0.4139	0.5375
SA+Contrast	**40.6113**	**2.1561**	**1.0529**	**0.5601**	**0.2796**	**0.4332**	**0.5572**

Table 8.2: Group captioning performance on the Conceptual Captions and Stock Captions dataset.

In this section, we first describe our evaluation results on the two datasets. Then we provide quantitative analysis and visualization to expose the effectiveness of different components of our model.

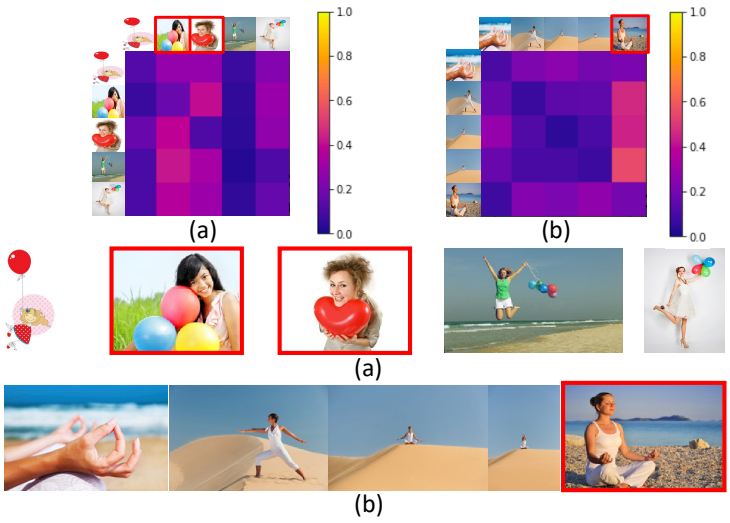


Figure 8.5: Visualization of 5×5 self-attention weight matrix for target image group. Each row sums up to 1. For group (a) `woman with balloon`, image 2 and image 3 are representative. For group (b) `yoga on beach`, image5 is representative. Images with more distinguishable features become the representative images of a group and get higher attention weights.

8.5.1 Group Captioning Performance

We evaluate our context-aware group captioning method on both Conceptual Captions and Stock Captions datasets. The same hyper-parameters are used for all experiments on each dataset. On the Stock Captions dataset, we use batch size 512 and initial learning rate 1×10^{-4}. On the Conceptual Captions dataset, we use batch size 512 and learning rate 5×10^{-5}. We train the model for 100 epochs with Adam optimizer[317] on both datasets.

We measure the captioning performance on the test splits in both datasets using a variety of captioning metrics. Specifically, we consider the standard metrics widely used in image captioning literature, including BLEU[318],

Model	WordAcc	CIDER	BLEU2	METEOR	ROUGE
Tgt0 + Ref15	24.4709	1.0399	0.0614	0.2341	0.3965
Tgt1 + Ref15	28.7479	1.3447	0.1292	0.2938	0.4415
Tgt3 + Ref15	34.6574	1.7641	0.2098	0.3698	0.5048
Tgt5 + Ref0	31.8061	1.6767	0.2095	0.3475	0.4552
Tgt5 + Ref15	**40.6113**	**2.1561**	**0.2796**	**0.4332**	**0.5572**

Table 8.3: Performance with varying the number of target and reference images. (evaluated on Stock Captions dataset)

CIDER[319], METEOR[320] and ROUGE[321]. In addition, since group descriptions are often short and compact, we put more emphasis on single word accuracy compared to traditional image captioning. We thus consider two additional metrics, Word-by-word accuracy(WordAcc), word error rate(WER), that specifically assess word-based accuracy[5]. We also note that as some group descriptions may contain as few as two words, we do not consider BLEU3 and BLEU4 scores which evaluates tri-grams and 4-grams.

The captioning performance on the testing set of Conceptual Captions and Stock Captions datasets are reported in Table 8.2. To compare with a simple baseline, we caption each image individually and summarize them using our dataset building method. The result (**Per-Img. Caption**) shows that the group captioning problem cannot be solved by simply summarizing per-image captions. More details are shown in supplementary materials. Compared to aggregating features by averaging (**Average**, as in Section 8.4.2), self-attention (**SA**) is more effective in computing group representation and leads to significant performance improvement. On top of feature aggregation, contrastive feature is critical for the model to generate context-aware caption

[5]Here we consider position-specific word accuracy. For example, prediction *woman with straw hat* with ground truth *woman with cowboy hat* has accuracy 75%, while prediction *woman with hat* has accuracy 50%.

which emphasizes the difference of target image group on context of reference group. Applying contrastive features (**Contrast**) to either feature aggregation methods leads to performance boost (**Average+Contrast, SA+Contrast**). To this end, our full model, which combines self-attention for group aggregation and contrastive feature for group comparing performs best, achieving 39.4% WordAcc on Conceptual Captions and 40.6% on Stock Captions.

8.5.2 Discussion

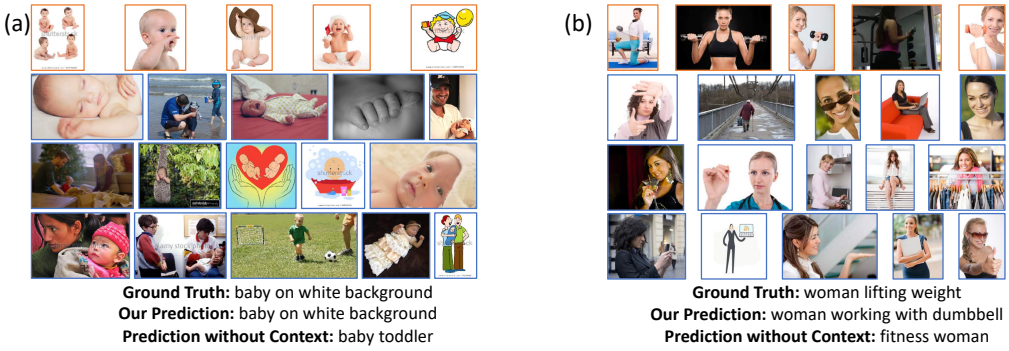


Figure 8.6: Qualitative prediction examples on Conceptual Captions (a) and Stock Captions (b) datasets. In each example, images in first row (in orange boxes) are target images while second to fourth rows (in blue boxes) are reference images. Our model can effectively summarize relevant information in the image groups during captioning. Our model also effectively takes discriminative information between the target and reference group into account during captioning to predict accurate group captioning results.

Effectiveness of self-attention on feature aggregation. To better understand the effectiveness of self-attention, in Figure 8.5, we visualize the 5×5 self-attention weight matrix between 5 target images. The i-th row of the attention matrix represents the attention weights from i-th image to each of the 5 images, which sum up to 1. In (a), images with larger and centered balloons (Image2 and Image3) gets higher attention. In (b), image5 where the woman doing

yoga is larger and easier to recognize gets higher attention. In both examples, images with more recognizable features get higher attention weights and thus contribute more to the aggregated group representation.

Contrastive + Group	Group	Contrastive
woman with cowboy hat	woman	country with cowboy straw hat
white girl	girl	white rule white and...
woman in boxing glove	woman	is go in boxing...

Table 8.4: Analysis of contrastive representation. Column `Contrastive + Group` is the prediction of our full model. Column `Group` and column `Contrastive` are the predictions when only the group or only the contrastive representation is fed into the decoder respectively. Blue text denotes the common part while red text denotes the contrastive part.

Importance of multiple target and reference images. To investigate the effectiveness of giving multiple images in each group, we vary the number of target and reference images. Results are shown in Table 8.3. Fewer target or reference images results in performance decline, which indicates that a larger number of images is more informational for the model to get better descriptions. We also qualitatively study the importance of the reference image group. Examples are shown in Figure 8.6. The examples indicate that when not giving reference group the predictions tend to be more generic and less discriminative.

Contrastive representation versus group representation. Table 8.4 shows example descriptions when only the group representations or only the contrastive representations are fed into LSTM decoder. Although the model does not treat the features independently and removing the features might break the grammar structure of the caption, looking at the lexicons returned by the

two variants, we can clearly observe the focus of two features. When the decoder uses only the group representations, the predictions emphasize the common part of two image groups. On the other hand, when the decoder only uses the contrastive representations, the predictions emphasize the difference between two image groups. This reveals that the group representation encodes similarity information, while the contrastive representation encodes discriminative information.

Robustness to noise images. To investigate the model's robustness to noise in the image group, we tried adding random unrelated images to the target group. Figure 8.7 shows performances of models trained and tested with different number (0-4) of noise images on Conceptual Captions dataset. Training with more noise increases robustness of the model but hinder performance when tested with no noise. The model shows robustness to small noise. Qualitatively, when testing with small (1 or 2) noise (trained with 0 noise), the caption loses details, e.g. woman in red dress becomes woman in dress. The generated caption is broken when the noise is severe, which is reasonable.

8.6 Conclusion

In this chapter, I present the novel context-aware group captioning task, where the objective is to describe a target image group in contrast to a reference image group. To explore this problem, we introduce two large scale datasets, Conceptual Captions and Stock Captions respectively, both of which will be released for future research. We also propose a framework with self-attention for grouping the images and contrastive representation for capturing

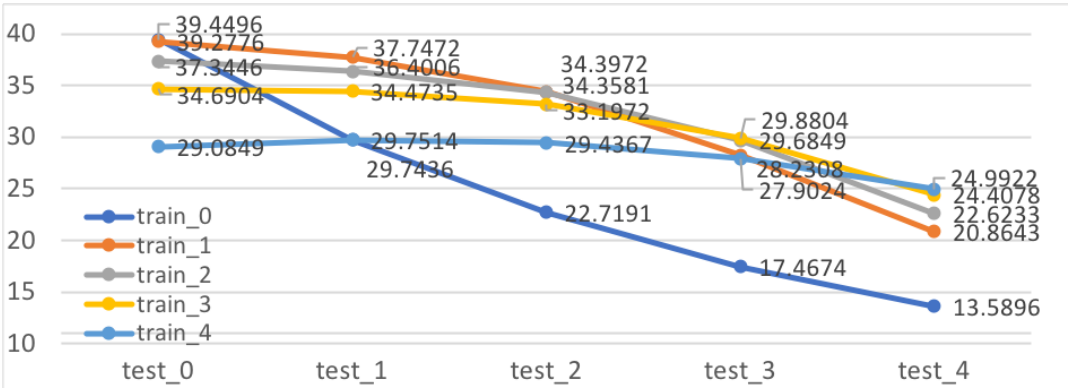


Figure 8.7: Performance change on Conceptual Captions dataset when trained and tested with 0-4 random images in the target group. Training with more noise increases robustness of the model but hinder performance when tested with no noise.

discriminative features. We show the effectiveness of our proposed model both quantitatively and qualitatively on our datasets. We also thoroughly analyze the behavior of our models to provide insights into this new problem.

Chapter 9

Discussion and Conclusion

9.1 Summary

In this dissertation, I present a total of seven research projects, all of which focus on reasoning with vision and language.

In Part I, I study the visual question answering task, from the perspective of model diagnosis. Chapter 2 perturbs the model input by feature shuffling and shows that current VQA models are surprising not robust to changes in features of irrelevant visual elements. Chapter 3 benchmarks the domain robustness of existing VQA models by introducing a new synthetic benchmark named Super-CLEVR that is controllable over various factors. Chapter 4 further extends Super-CLEVR with 3D-aware questions, and show that without explicit 3D understanding, current models cannot correctly answer questions about parts, poses, and occlusions.

In Part II, I dive deeper into compositional visual reasoning models, proposing various techniques to enhance the models for better generalization on real data. Chapter 5 analyzes the challenges in generalizing the synthetically

successful compositional models to real-world images, and addresses the challenges by calibration of concepts and operations. Chapter 6 improves the recent LLM-based compositional reasoning system with introducing verification modules, which examine and correct the errors in the step-wise reasoning results. Chapter 7 studies the visual representations in foundation models, from the perspective of feature probing in semantics and localization tasks, which aims to serve as a guidebook for choosing visual foundation models.

In Part III, I explore a new task, contrastive group captioning, and show the challenges in the reasoning procedure, comparing a group of images instead of single images.

9.2 Limitations

In this section, I discuss the limitations of my works in this thesis. If I were to repeat the work, here are a few things that I would be focusing more on:

9.2.1 Generalization of compositional methods.

Despite the stronger generalization ability towards domain shifts as shown in Chapter 3, generalization means more than that. Here I show a few cases. First, the current symbolic methods as in Chapter 5 and Chapter 3 is limited to a closed set of operations that are supported by hand-designed modules, thus cannot handle open-domain questions. For example, currently there is no operation to handle "why" questions like "why is the man holding an umbrella?". Second, some questions can hardly be decomposed into isolated operations, *e.g.* "which person in this image is different from others?", thus

cannot be handled by current symbolic methods following an decompose-then-execute paradigm. Third, current compositional models do not have a holistic understanding of the image and text together, which results in failures on ambiguous questions. For example, the ambiguity in question "where is the white wine bottle?", which can refer to the bottle that holds white wine, or the white bottle that holds wine, can be resolved by looking at the image, but is hard for a text-only question parser. Fourth and further, current models are not well-calibrated, so it is hard to know what they don't know. With these limitations, there are few works that solve more challenging datasets, like VQAv2 [10] with compositional models.

The lack of generalization ability significantly hinders the applications of compositional methods, as the flexibility to handle free-form open-world questions is essential for successful models. This is probably one of the reasons why we prefer to use Chat-GPT over many domain-specific methods in our daily life. More efforts could be made to solve the generalization issue, not by investing time and labours in designing modules to support more types of operations, but by redesigning the system to allow flexible components in the model.

9.2.2 Free-form VQA versus closed-form VQA.

In this thesis, I study VQA and image captioning as two seperate tasks, where VQA serves as a discriminative probe that queries the model precisely while image captioning serves as a generative task. A better perspective is to unify the two tasks and study free-form VQA, where the answer is a free-form

description instead of a class from a closed set of candidate answers. Free-form VQA is not only more challenging from a research perspective, but also more flexible to applied into more applications. However, there is still a long way to go for compositional models to solve free-form VQA. While I put efforts in compositional models for discriminative understanding, enforcing their generative capacity remains an open problem.

9.2.3 Compositional versus non-compositional methods

So far, I show the advantages of compositional methods like robustness and interpretability, as well as disadvantages like performance and flexibility, in comparison with non-compositional end-to-end methods. Given that each type of methods has pros and cons, a next-step solution is to embrace the power of both the rigid but reliable symbolic modules and the flexible yet black-box end-to-end methods, in order to achieve the best of both worlds. Solutions like model distillation and model ensemble like mixture-of-expert, could be explored.

9.3 Future Works

I will introduce future works for both evaluating and learning of the visual reasoning system and what the next steps could be, in the current big model age.

9.3.1 Systematic evaluation of visual reasoning.

Performance on proxy tasks has long been used as the evaluation protocol to represent model performance. However, this protocol might be misleading, for two reasons.

On one hand, due to the complexity of the tasks, the overall accuracy does not necessarily reflect the models' capacity along distinct axis. Actually, visual reasoning is a high-order ability that requires a broad coverage of various factors, *e.g.*, understanding of visual contexts, visual commonsense knowledge, human intention, intuitive physics, mathematical calculation, etc. For example, given an image of a man carrying an umbrella and the challenging question "what is this umbrella for?", the model needs to correctly recognize the objects like man and umbrella, understand the weather (rainy or sunny) from the lighting or background of the image, utilize the visual commonsense knowledge of the possible usage of an umbrella, and finally answer whether the umbrella is used for sunshine or rain. With only the correctness of the final answer, it is hard to understand, thus diagnose the bottleneck of the model. How to define the different factors, thus systematically evaluate on them, remains an open topic. Recent benchmarks

On the other hand, due to possible bias and shortcuts in the data, a high performance on the proxy tasks does not necessarily reflect the model's true understanding ability. As discovered by the success of adversarial attacks of neural networks, which shows the vulnerability of the seemingly well-performed models, the good average performance on standard benchmarks does not reflect the worst-case performance. When the testing data lies out

of training distribution, a well-performed model can be significantly fooled and make incorrect predictions. Therefore, a more systematic testing protocol, which "push the model hard" and evaluate the worst-case performance beyond the average performance, should be studied.

9.3.2 Joint learning of vision and language

Most recent works plug vision into pretrained large language models in a post-hoc manner [322, 176, 323], where the language model is pre-learned using text-only data, and then the visual features is adapted and fed into the pretrained model afterwards. However, I believe that the joint learning of multiple modalities, like vision and language, offers mutual benefits. In [232], we show that multimodal models learn better visual commonsense knowledge than text-only models; in Chapter 7, I show preliminary analysis that language enhances the semantics encoded in visual representations. Inspired by these findings, I believed that an improved joint learning of multiple modalities will enhance the current large pretrained models.

However, there are challenges in the joint learning of vision and language considering the intrinsic differences between the two domains. While texts are discrete and naturally tokenized into words, images and videos are continuous natural signals. It is noticable that recent efforts explores tokenziation in visual domain, *e.g.*, VQ-VAE [264] and VQ-GAN [265] explore discrete tokenization of images, slot attention [324] explores unsupervised object-centric representation learning. Despite the progress, the success of transformers in NLP hasn't yet been fully replicated in the vision field. While recent works

show promising results using transformer backbone for tasks like classification, detection and segmentation, it is still open whether these models can successfully scale like in NLP.

9.3.3 Compositionality in large pretrained models.

Recently, LLMs and VLMs demonstrate impressive promise in the multimodal field. Evidences, like chain-of-thought [325] and program-of-thought [326], have shown that compositionality appears as an emergent ability as the pretrained models grow larger and larger.

As a next step, it is worth investigating how to better utilize the emergent compositionality to further improve the models and to better interpret and control the models. For example, can we distill the compositionality into smaller models? Can we utilize LLMs to build better compositional models? Can we apply better controllability over the models predictions by making use of its intermediate predictions? In Chapter 6, I introduce my preliminary effort in this direction. I am glad to see that there are recent research works going along this way [327, 328].

Bibliography

[1] Z. Yu, J. Yu, Y. Cui, D. Tao, and Q. Tian, "Deep modular co-attention networks for visual question answering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6281–6290.

[2] H. Tan and M. Bansal, "Lxmert: Learning cross-modality encoder representations from transformers," *arXiv preprint arXiv:1908.07490*, pp. 5100–5111, 2019.

[3] R. Cadène, H. Ben-younes, M. Cord, and N. Thome, "Murel: Multimodal relational reasoning for visual question answering," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1989–1998, 2019.

[4] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, and K.-W. Chang, "Visualbert: A simple and performant baseline for vision and language," *ArXiv*, vol. abs/1908.03557, 2019.

[5] X. Li, X. Yin, C. Li, P. Zhang, X. Hu, L. Zhang, L. Wang, H. Hu, L. Dong, F. Wei *et al.*, "Oscar: Object-semantics aligned pre-training for vision-language tasks," in *European Conference on Computer Vision*. Springer,

2020, pp. 121–137.

[6] A. Agrawal, D. Batra, D. Parikh, and A. Kembhavi, "Don't just assume; look and answer: Overcoming priors for visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4971–4980.

[7] Y. Niu, K. Tang, H. Zhang, Z. Lu, X. Hua, and J.-R. Wen, "Counterfactual vqa: A cause-effect look at language bias," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12 695–12 705, 2021.

[8] Y. Niu, K. Tang, H. Zhang, Z. Lu, X.-S. Hua, and J.-R. Wen, "Counterfactual vqa: A cause-effect look at language bias," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 12 700–12 710.

[9] A. Agrawal, D. Batra, and D. Parikh, "Analyzing the behavior of visual question answering models," *arXiv preprint arXiv:1606.07356*, 2016.

[10] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh, "Making the v in vqa matter: Elevating the role of image understanding in visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6904–6913.

[11] C. Kervadec, G. Antipov, M. Baccouche, and C. Wolf, "Roses are red, violets are blue... but should vqa expect them to?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2776–2785.

[12] C. Kervadec, T. Jaunet, G. Antipov, M. Baccouche, R. Vuillemot, and C. Wolf, "How transferable are reasoning patterns in vqa?" *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4205–4214, 2021.

[13] D. A. Hudson and C. D. Manning, "Gqa: A new dataset for real-world visual reasoning and compositional question answering," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6700–6709.

[14] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "Vqa: Visual question answering," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2425–2433.

[15] B. Zhou, Y. Tian, S. Sukhbaatar, A. Szlam, and R. Fergus, "Simple baseline for visual question answering," *arXiv preprint arXiv:1512.02167*, 2015.

[16] K. Chen, J. Wang, L.-C. Chen, H. Gao, W. Xu, and R. Nevatia, "Abc-cnn: An attention based convolutional neural network for visual question answering," *arXiv preprint arXiv:1511.05960*, 2015.

[17] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola, "Stacked attention networks for image question answering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 21–29.

[18] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach, "Multimodal compact bilinear pooling for visual question answering and visual grounding," *arXiv preprint arXiv:1606.01847*, 2016.

[19] J.-H. Kim, K.-W. On, W. Lim, J. Kim, J.-W. Ha, and B.-T. Zhang, "Hadamard product for low-rank bilinear pooling," *arXiv preprint arXiv:1610.04325*, 2016.

[20] Z. Yu, J. Yu, J. Fan, and D. Tao, "Multi-modal factorized bilinear pooling with co-attention learning for visual question answering," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1821–1830.

[21] Z. Yu, J. Yu, C. Xiang, J. Fan, and D. Tao, "Beyond bilinear: Generalized multimodal factorized high-order pooling for visual question answering," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 12, pp. 5947–5959, 2018.

[22] H. Ben-Younes, R. Cadene, M. Cord, and N. Thome, "Mutan: Multimodal tucker fusion for visual question answering," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2612–2620.

[23] J.-H. Kim, J. Jun, and B.-T. Zhang, "Bilinear attention networks," *arXiv preprint arXiv:1805.07932*, vol. 31, 2018.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, vol. 30, 2017, pp. 5998–6008.

[25] P. Cascante-Bonilla, H. Wu, L. Wang, R. Feris, and V. Ordonez, "Simvqa: Exploring simulated environments for visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 5056–5066.

[26] L. A. Hendricks, K. Burns, K. Saenko, T. Darrell, and A. Rohrbach, "Women also snowboard: Overcoming bias in captioning models," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 771–787.

[27] V. Manjunatha, N. Saini, and L. S. Davis, "Explicit bias discovery in visual question answering models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9562–9571.

[28] S. Ramakrishnan, A. Agrawal, and S. Lee, "Overcoming language priors in visual question answering with adversarial regularization," *arXiv preprint arXiv:1810.03649*, 2018.

[29] R. Cadene, C. Dancette, M. Cord, D. Parikh *et al.*, "Rubi: Reducing unimodal biases for visual question answering," *Advances in neural information processing systems*, vol. 32, pp. 841–852, 2019.

[30] L. Chen, X. Yan, J. Xiao, H. Zhang, S. Pu, and Y. Zhuang, "Counterfactual samples synthesizing for robust visual question answering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 800–10 809.

[31] J. Wu and R. J. Mooney, "Self-critical reasoning for robust visual question answering," *arXiv preprint arXiv:1905.09998*, 2019.

[32] R. R. Selvaraju, S. Lee, Y. Shen, H. Jin, S. Ghosh, L. Heck, D. Batra, and D. Parikh, "Taking a hint: Leveraging explanations to make vision and language models more grounded," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2591–2600.

[33] C. Clark, M. Yatskar, and L. Zettlemoyer, "Don't take the easy way out: Ensemble based methods for avoiding known dataset biases," *arXiv preprint arXiv:1909.03683*, 2019.

[34] T. Gokhale, P. Banerjee, C. Baral, and Y. Yang, "Mutant: A training paradigm for out-of-distribution generalization in visual question answering," *arXiv preprint arXiv:2009.08566*, 2020.

[35] M. Malinowski and M. Fritz, "A multi-world approach to question answering about real-world scenes based on uncertain input," *Advances in neural information processing systems*, vol. 27, pp. 1682–1690, 2014.

[36] D. Bahdanau, H. de Vries, T. J. O'Donnell, S. Murty, P. Beaudoin, Y. Bengio, and A. Courville, "Closure: Assessing systematic generalization of clevr models," *arXiv preprint arXiv:1912.05783*, 2019.

[37] C. Dancette, R. Cadène, D. Teney, and M. Cord, "Beyond question-based biases: Assessing multimodal shortcut learning in visual question answering," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1554–1563, 2021.

[38] A. Torralba, K. P. Murphy, W. T. Freeman, and M. A. Rubin, "Context-based vision system for place and object recognition," in *Computer Vision, IEEE International Conference on*, vol. 2. IEEE Computer Society, 2003, pp. 273–273.

[39] A. Oliva and A. Torralba, "The role of context in object recognition," *Trends in cognitive sciences*, vol. 11, no. 12, pp. 520–527, 2007.

[40] Q. Zhong, C. Li, Y. Zhang, D. Xie, S. Yang, and S. Pu, "Cascade region proposal and global context for deep object detection," *Neurocomputing*, vol. 395, pp. 170–177, 2020.

[41] A. Wang, Y. Sun, A. Kortylewski, and A. L. Yuille, "Robust object detection under occlusion with context-aware compositionalnets," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 645–12 654.

[42] A. Kortylewski, J. He, Q. Liu, and A. L. Yuille, "Compositional convolutional neural networks: A deep architecture with innate robustness to partial occlusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8940–8949.

[43] A. Kortylewski, Q. Liu, A. Wang, Y. Sun, and A. Yuille, "Compositional convolutional neural networks: A robust and interpretable model for object recognition under occlusion," *International Journal of Computer Vision*, vol. 129, no. 3, pp. 736–760, 2021.

[44] G. Ren, L. Ren, Y. Liao, S. Liu, B. Li, J. Han, and S. Yan, "Scene graph generation with hierarchical context," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 2, pp. 909–915, 2020.

[45] W. Wang, R. Wang, S. Shan, and X. Chen, "Sketching image gist: Human-mimetic hierarchical scene graph generation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*. Springer, 2020, pp. 222–239.

[46] X. Lin, C. Ding, J. Zeng, and D. Tao, "Gps-net: Graph property sensing network for scene graph generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3746–3753.

[47] J. He, Z. Deng, L. Zhou, Y. Wang, and Y. Qiao, "Adaptive pyramid context network for semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7519–7528.

[48] D. Lin, R. Zhang, Y. Ji, P. Li, and H. Huang, "Scn: switchable context network for semantic segmentation of rgb-d images," *IEEE transactions on cybernetics*, vol. 50, no. 3, pp. 1120–1131, 2018.

[49] H. Sharma and A. S. Jalal, "Visual question answering model based on graph neural network and contextual attention," *Image and Vision Computing*, vol. 110, p. 104165, 2021.

[50] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-up and top-down attention for image captioning and visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6077–6086.

[51] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 1137–1149, 2015.

[52] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[53] E. Million, "The hadamard product," 2007.

[54] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu, "The Neuro-Symbolic Concept Learner: Interpreting Scenes, Words, and Sentences From Natural Supervision," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=rJgMlhRctm

[55] K. Yi, J. Wu, C. Gan, A. Torralba, P. Kohli, and J. B. Tenenbaum, "Neural-Symbolic VQA: Disentangling Reasoning from Vision and Language Understanding," in *Advances in Neural Information Processing Systems (NIPS)*, 2018.

[56] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[57] A. Kamath, M. Singh, Y. LeCun, I. Misra, G. Synnaeve, and N. Carion, "Mdetr–modulated detection for end-to-end multi-modal understanding," *arXiv preprint arXiv:2104.12763*, pp. 1780–1790, 2021.

[58] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh, "Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[59] D. A. Hudson and C. D. Manning, "Gqa: A new dataset for real-world visual reasoning and compositional question answering," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6700–6709.

[60] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The journal of machine learning research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[61] F. Qiao, L. Zhao, and X. Peng, "Learning to learn single domain generalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 556–12 565.

[62] D. Li, J. Zhang, Y. Yang, C. Liu, Y.-Z. Song, and T. M. Hospedales, "Episodic training for domain generalization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1446–1455.

[63] W.-L. Chao, H. Hu, and F. Sha, "Cross-dataset adaptation for visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5716–5725.

[64] M. Zhang, T. Maidment, A. Diab, A. Kovashka, and R. Hwa, "Domain-robust vqa with diverse datasets and methods but no target labels," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7046–7056.

[65] Y. Xu, L. Chen, Z. Cheng, L. Duan, and J. Luo, "Open-ended visual question answering by multi-modal domain adaptation," *arXiv preprint arXiv:1911.04058*, 2019.

[66] A. Agrawal, D. Batra, D. Parikh, and A. Kembhavi, "Don't just assume; look and answer: Overcoming priors for visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4971–4980.

[67] Z. Li, E. Stengel-Eskin, Y. Zhang, C. Xie, Q. H. Tran, B. Van Durme, and A. Yuille, "Calibrating concepts and operations: Towards symbolic reasoning on real images," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 14 910–14 919.

[68] A. Akula, S. Changpinyo, B. Gong, P. Sharma, S.-C. Zhu, and R. Soricut, "Crossvqa: scalably generating benchmarks for systematically testing vqa generalization," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 2148–2166.

[69] J. Jiang, Z. Liu, Y. Liu, Z. Nan, and N. Zheng, "X-ggm: Graph generative modeling for out-of-distribution generalization in visual question answering," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 199–208.

[70] J. Johnson, B. Hariharan, L. Van Der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick, "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2901–2910.

[71] Q. Liu, A. Kortylewski, Z. Zhang, Z. Li, M. Guo, Q. Liu, X. Yuan, J. Mu, W. Qiu, and A. Yuille, "Learning part segmentation through unsupervised domain adaptation from synthetic vehicles," in *CVPR*, 2022.

[72] Z. Yu, J. Yu, Y. Cui, D. Tao, and Q. Tian, "Deep modular co-attention networks for visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 6281–6290.

[73] D. A. Hudson and C. D. Manning, "Compositional attention networks for machine reasoning," 2018.

[74] L. Li, Z. Gan, Y. Cheng, and J. Liu, "Relation-aware graph attention network for visual question answering," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 10 313–10 322.

[75] J. Lu, D. Batra, D. Parikh, and S. Lee, "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," *arXiv preprint arXiv:1908.02265*, vol. 32, 2019.

[76] P. Zhang, X. Li, X. Hu, J. Yang, L. Zhang, L. Wang, Y. Choi, and J. Gao, "Vinvl: Making visual representations matter in vision-language models," *CVPR 2021*, 2021.

[77] Y. Zhu, O. Groth, M. Bernstein, and L. Fei-Fei, "Visual7w: Grounded question answering in images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4995–5004.

[78] D. Gurari, Q. Li, A. J. Stangl, A. Guo, C. Lin, K. Grauman, J. Luo, and J. P. Bigham, "Vizwiz grand challenge: Answering visual questions from blind people," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3608–3617.

[79] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma *et al.*, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," *International Journal of Computer Vision*, vol. 123, no. 1, pp. 32–73, 2017.

[80] M. Ren, R. Kiros, and R. Zemel, "Exploring models and data for image question answering," *Advances in neural information processing systems*, vol. 28, 2015.

[81] V. Gupta, Z. Li, A. Kortylewski, C. Zhang, Y. Li, and A. Yuille, "Swapmix: Diagnosing and regularizing the over-reliance on visual context in visual question answering," *arXiv preprint arXiv:2204.02285*, 2022.

[82] R. Liu, C. Liu, Y. Bai, and A. L. Yuille, "Clevr-ref+: Diagnosing visual reasoning with referring expressions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4185–4194.

[83] S. Kottur, J. M. Moura, D. Parikh, D. Batra, and M. Rohrbach, "Clevr-dialog: A diagnostic dataset for multi-round reasoning in visual dialog," *arXiv preprint arXiv:1903.03166*, 2019.

[84] K. Yi, C. Gan, Y. Li, P. Kohli, J. Wu, A. Torralba, and J. B. Tenenbaum, "Clevrer: Collision events for video representation and reasoning," *arXiv preprint arXiv:1910.01442*, 2019.

[85] C. Zhang, F. Gao, B. Jia, Y. Zhu, and S.-C. Zhu, "Raven: A dataset for relational and analogical visual reasoning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5317–5327.

[86] L. Salewski, A. Koepke, H. Lensch, and Z. Akata, "Clevr-x: A visual reasoning dataset for natural language explanations," in *International Workshop on Extending Explainable AI Beyond Deep Models and Classifiers*. Springer, 2022, pp. 69–88.

[87] Y. Hong, L. Yi, J. Tenenbaum, A. Torralba, and C. Gan, "Ptr: A benchmark for part-based conceptual, relational, and physical reasoning," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[88] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International conference on machine learning*. PMLR, 2015, pp. 1180–1189.

[89] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, "Cycada: Cycle-consistent adversarial domain adaptation," in *International conference on machine learning*. PMLR, 2018, pp. 1989–1998.

[90] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," *Advances in neural information processing systems*, vol. 31, 2018.

[91] W. Ford and D. Olson, "The elaboration of the noun phrase in children's description of objects," *Journal of Experimental Child Psychology*, vol. 19, no. 3, pp. 371–382, 1975.

[92] S. Sonnenschein, "The development of referential communication skills: Some situations in which speakers give redundant messages," *Journal of Psycholinguistic Research*, vol. 14, no. 5, pp. 489–508, 1985.

[93] T. Pechmann, "Incremental speech production and referential overspecification," 1989.

[94] R. Koolen, M. Goudbeek, and E. Krahmer, "Effects of scene variation on referential overspecification," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 33, no. 33, 2011.

[95] M. Mitchell, K. Van Deemter, and E. Reiter, "Generating expressions that refer to visible objects," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013, pp. 1174–1184.

[96] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu, "Large-scale long-tailed recognition in an open world," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2537–2546.

[97] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao, "Range loss for deep face recognition with long-tailed training data," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5409–5418.

[98] R. He, J. Yang, and X. Qi, "Re-distributing biased pseudo labels for semi-supervised semantic segmentation: A baseline investigation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6930–6940.

[99] G. Van Horn, O. Mac Aodha, Y. Song, Y. Cui, C. Sun, A. Shepard, H. Adam, P. Perona, and S. Belongie, "The inaturalist species classification and detection dataset," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8769–8778.

[100] L. Ju, X. Wang, L. Wang, T. Liu, X. Zhao, T. Drummond, D. Mahapatra, and Z. Ge, "Relational subsets knowledge distillation for long-tailed retinal diseases recognition," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2021, pp. 3–12.

[101] R. Cadene, C. Dancette, M. Cord, D. Parikh *et al.*, "Rubi: Reducing unimodal biases for visual question answering," *Advances in neural information processing systems*, vol. 32, 2019.

[102] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[103] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[104] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[105] Z. Li, X. Wang, E. Stengel-Eskin, A. Kortylewski, W. Ma, B. Van Durme, and A. Yuille, "Super-clevr: A virtual benchmark to diagnose domain robustness in visual reasoning," *arXiv preprint arXiv:2212.00259*, 2022.

[106] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Neural module networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 39–48.

[107] R. Hu, J. Andreas, T. Darrell, and K. Saenko, "Explainable neural computation via stack neural module networks," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 53–69.

[108] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[109] X. Ma, S. Yong, Z. Zheng, Q. Li, Y. Liang, S.-C. Zhu, and S. Huang, "Sqa3d: Situated question answering in 3d scenes," in *International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=IDJx97BC38

[110] Y. Hong, C. Lin, Y. Du, Z. Chen, J. B. Tenenbaum, and C. Gan, "3d concept learning and reasoning from multi-view images," *arXiv preprint arXiv:2303.11327*, 2023.

[111] X. Yan, Z. Yuan, Y. Du, Y. Liao, Y. Guo, Z. Li, and S. Cui, "Clevr3d: Compositional language and elementary visual reasoning for question answering in 3d real-world scenes," *arXiv preprint arXiv:2112.11691*, 2021.

[112] S. Ye, D. Chen, S. Han, and J. Liao, "3d question answering," *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[113] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra, "Embodied question answering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1–10.

[114] D. Azuma, T. Miyanishi, S. Kurita, and M. Kawanabe, "Scanqa: 3d question answering for spatial scene understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 19 129–19 139.

[115] R. Chen, Y. Liu, L. Kong, X. Zhu, Y. Ma, Y. Li, Y. Hou, Y. Qiao, and W. Wang, "Clip2scene: Towards label-efficient 3d scene understanding by clip," *arXiv preprint arXiv:2301.04926*, 2023.

[116] S. Peng, K. Genova, C. M. Jiang, A. Tagliasacchi, M. Pollefeys, and T. Funkhouser, "Openscene: 3d scene understanding with open vocabularies," in *CVPR*, 2023.

[117] X. Zhou, A. Karpur, L. Luo, and Q. Huang, "Starmap for category-agnostic keypoint and viewpoint estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 318–334.

[118] W. Ma, A. Wang, A. Yuille, and A. Kortylewski, "Robust category-level 6d pose estimation with coarse-to-fine rendering of neural features," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX*. Springer, 2022, pp. 492–508.

[119] T. Luo, C. Rockwell, H. Lee, and J. Johnson, "Scalable 3d captioning with pretrained models," *arXiv preprint arXiv:2306.07279*, 2023.

[120] Y. Hong, H. Zhen, P. Chen, S. Zheng, Y. Du, Z. Chen, and C. Gan, "3d-llm: Injecting the 3d world into large language models," *arXiv preprint arXiv:2307.12981*, 2023.

[121] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, and A. Yuille, "Detect what you can: Detecting and representing objects using holistic models and body parts," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1971–1978.

[122] A. Wang, A. Kortylewski, and A. Yuille, "Nemo: Neural mesh models of contrastive features for robust 3d pose estimation," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=pmj131uIL9H

[123] S. Liu, T. Li, W. Chen, and H. Li, "Soft rasterizer: A differentiable renderer for image-based 3d reasoning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7708–7717.

[124] Y. Bai, A. Wang, A. Kortylewski, and A. Yuille, "Coke: Contrastive learning for robust keypoint detection," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 65–74.

[125] X. Yuan, A. Kortylewski, Y. Sun, and A. Yuille, "Robust instance segmentation through reasoning about multi-object occlusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 141–11 150.

[126] Y. Xiang, R. Mottaghi, and S. Savarese, "Beyond pascal: A benchmark for 3d object detection in the wild," in *IEEE winter conference on applications of computer vision*. IEEE, 2014, pp. 75–82.

[127] Y. Ze and X. Wang, "Category-level 6d object pose estimation in the wild: A semi-supervised learning approach and a new dataset," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 469–27 483, 2022.

[128] A. Agrawal, D. Batra, D. Parikh, and A. Kembhavi, "Don't just assume; look and answer: Overcoming priors for visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4971–4980.

[129] J. Johnson, B. Hariharan, L. Van Der Maaten, J. Hoffman, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick, "Inferring and executing programs for visual reasoning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2989–2998.

[130] K. Yi, J. Wu, C. Gan, A. Torralba, P. Kohli, and J. B. Tenenbaum, "Neural-symbolic vqa: Disentangling reasoning from vision and language understanding," *arXiv preprint arXiv:1810.02338*, 2018.

[131] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu, "The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision," *arXiv preprint arXiv:1904.12584*, 2019.

[132] D. A. Hudson and C. D. Manning, "Learning by abstraction: The neural state machine," *arXiv preprint arXiv:1907.03950*, 2019.

[133] X. Yang, G. Lin, F. Lv, and F. Liu, "Trrnet: Tiered relation reasoning for compositional visual question answering."

[134] L. Chen, X. Yan, J. Xiao, H. Zhang, S. Pu, and Y. Zhuang, "Counterfactual samples synthesizing for robust visual question answering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 800–10 809.

[135] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*. PMLR, 2015, pp. 2048–2057.

[136] C. Liu, J. Mao, F. Sha, and A. Yuille, "Attention correctness in neural image captioning," in *Thirty-First AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[137] J. Lu, C. Xiong, D. Parikh, and R. Socher, "Knowing when to look: Adaptive attention via a visual sentinel for image captioning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 375–383.

[138] Z. Li, Q. Tran, L. Mai, Z. Lin, and A. L. Yuille, "Context-aware group captioning via self-attention and contrastive features," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3440–3450.

[139] L. Yu, Z. Lin, X. Shen, J. Yang, X. Lu, M. Bansal, and T. L. Berg, "Mattnet: Modular attention network for referring expression comprehension," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1307–1315.

[140] S. Yang, G. Li, and Y. Yu, "Dynamic graph attention for referring expression comprehension," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4644–4653.

[141] P. Wang, Q. Wu, J. Cao, C. Shen, L. Gao, and A. v. d. Hengel, "Neighbourhood watch: Referring expression comprehension via language-guided graph attention networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1960–1968.

[142] C. Zhang, B. Jia, F. Gao, Y. Zhu, H. Lu, and S.-C. Zhu, "Learning perceptual inference by contrasting," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[143] C. Zhang, B. Jia, S.-C. Zhu, and Y. Zhu, "Abstract spatial-temporal reasoning via probabilistic abduction and execution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9736–9746.

[144] C. Zhang, B. Jia, M. Edmonds, S.-C. Zhu, and Y. Zhu, "Acre: Abstract causal reasoning beyond covariation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[145] W. Zhang, C. Zhang, Y. Zhu, and S.-C. Zhu, "Machine number sense: A dataset of visual arithmetic problems for abstract and relational reasoning," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[146] D. Mascharka, P. Tran, R. Soklaski, and A. Majumdar, "Transparency by design: Closing the gap between performance and interpretability in visual reasoning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4942–4950.

[147] W. Chen, Z. Gan, L. Li, Y. Cheng, W. Wang, and J. Liu, "Meta module network for compositional visual reasoning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 655–664.

[148] R. Hu, J. Andreas, M. Rohrbach, T. Darrell, and K. Saenko, "Learning to reason: End-to-end module networks for visual question answering," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 804–813.

[149] Q. Li, S. Huang, Y. Hong, and S.-C. Zhu, "A competence-aware curriculum for visual concepts learning via question answering," in *European Conference on Computer Vision*. Springer, 2020, pp. 141–157.

[150] S. Amizadeh, H. Palangi, A. Polozov, Y. Huang, and K. Koishida, "Neuro-symbolic visual reasoning: Disentangling "Visual" from

"Reasoning"," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, 13–18 Jul 2020, pp. 279–290. [Online]. Available: https://proceedings.mlr.press/v119/amizadeh20a.html

[151] S. Zhang, X. Ma, K. Duh, and B. Van Durme, "Amr parsing as sequence-to-graph transduction," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 80–94.

[152] ——, "Broad-coverage semantic parsing as transduction," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3777–3789.

[153] E. Stengel-Eskin, A. S. White, S. Zhang, and B. Van Durme, "Universal decompositional semantic parsing," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 8427–8439.

[154] T. Gupta and A. Kembhavi, "Visual programming: Compositional visual reasoning without training," June 2023, pp. 14 953–14 962.

[155] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Neural module networks," *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 39–48, 2015.

[156] R. Hu, J. Andreas, M. Rohrbach, T. Darrell, and K. Saenko, "Learning to reason: End-to-end module networks for visual question answering,"

in *International Conference on Computer Vision (ICCV)*. IEEE Computer Society, 2017, pp. 804–813.

[157] J. Johnson, B. Hariharan, L. van der Maaten, J. Hoffman, L. Fei-Fei, C. L. Zitnick, and R. B. Girshick, "Inferring and executing programs for visual reasoning," *International Conference on Computer Vision (ICCV)*, pp. 3008–3017, 2017.

[158] R. Hu, J. Andreas, T. Darrell, and K. Saenko, "Explainable neural computation via stack neural module networks," in *European Conference on Computer Vision (ECCV)*, ser. Lecture Notes in Computer Science, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., vol. 11211. Springer, 2018, pp. 55–71.

[159] H. Le, N. F. Chen, and S. C. H. Hoi, "VGNMN: video-grounded neural module networks for video-grounded dialogue systems," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, M. Carpuat, M. de Marneffe, and I. V. M. Ruíz, Eds. Association for Computational Linguistics, 2022, pp. 3377–3393.

[160] Z. Qian, X. Wang, X. Duan, H. Chen, and W. Zhu, "Dynamic spatio-temporal modular network for video question answering," in *MM '22: The 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10 - 14, 2022*, J. Magalhães, A. D. Bimbo, S. Satoh, N. Sebe,

X. Alameda-Pineda, Q. Jin, V. Oria, and L. Toni, Eds. ACM, 2022, pp. 4466–4477.

[161] A. Radford and K. Narasimhan, "Improving language understanding by generative pre-training," 2018.

[162] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[163] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. J. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *ArXiv*, vol. abs/2005.14165, 2020.

[164] OpenAI, "Gpt-4 technical report," *ArXiv*, vol. abs/2303.08774, 2023.

[165] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. M. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. C. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. García, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee,

Z. Zhou, X. Wang, B. Saeta, M. Díaz, O. Firat, M. Catasta, J. Wei, K. S. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel, "Palm: Scaling language modeling with pathways," *ArXiv*, vol. abs/2204.02311, 2022.

[166] S. Dídac, S. Menon, and C. Vondrick, "Vipergpt: Visual inference via python execution for reasoning," *arXiv preprint arXiv:2303.08128*, 2023.

[167] D. A. Hudson and C. D. Manning, "Gqa: A new dataset for real-world visual reasoning and compositional question answering," *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6693–6702, 2019.

[168] L. Pan, M. S. Saxon, W. Xu, D. Nathani, X. Wang, and W. Y. Wang, "Automatically correcting large language models: Surveying the landscape of diverse self-correction strategies," *ArXiv*, vol. abs/2308.03188, 2023.

[169] L. Yu, P. Poirson, S. Yang, A. C. Berg, and T. L. Berg, "Modeling context in referring expressions," *Lecture Notes in Computer Science*, p. 69–85, 2016.

[170] S. Kazemzadeh, V. Ordonez, M. Matten, and T. Berg, "ReferItGame: Referring to objects in photographs of natural scenes," in *Annual Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 787–798.

[171] A. Suhr, S. Zhou, A. Zhang, I. Zhang, H. Bai, and Y. Artzi, "A corpus for reasoning about natural language grounded in photographs," in *Proceedings of the 57th Annual Meeting of the Association for Computational*

Linguistics. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 6418–6428.

[172] A. Ji, N. Kojima, N. Rush, A. Suhr, W. K. Vong, R. D. Hawkins, and Y. Artzi, "Abstract visual reasoning with tangram shapes," in *Annual Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

[173] K. Zhang, L. Mo, W. Chen, H. Sun, and Y. Su, "Magicbrush: A manually annotated dataset for instruction-guided image editing," *ArXiv*, vol. abs/2306.10012, 2023.

[174] M. Tsimpoukelli, J. Menick, S. Cabi, S. M. A. Eslami, O. Vinyals, F. Hill, and Z. Janssen, "Multimodal few-shot learning with frozen language models," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[175] J. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J. L. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Binkowski, R. Barreira, O. Vinyals, A. Zisserman, and K. Simonyan, "Flamingo: a visual language model for few-shot learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[176] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," *arXiv preprint arXiv:2301.12597*, 2023.

[177] P. Gao, J. Han, R. Zhang, Z. Lin, S. Geng, A. Zhou, W. Zhang, P. Lu, C. He, X. Yue, H. Li, and Y. Qiao, "Llama-adapter V2: parameter-efficient visual instruction model," *CoRR*, vol. abs/2304.15010, 2023.

[178] B. Li, Y. Zhang, L. Chen, J. Wang, J. Yang, and Z. Liu, "Otter: A multi-modal model with in-context instruction tuning," *CoRR*, vol. abs/2305.03726, 2023.

[179] W. Dai, J. Li, D. Li, A. M. H. Tiong, J. Zhao, W. Wang, B. Li, P. Fung, and S. C. H. Hoi, "Instructblip: Towards general-purpose vision-language models with instruction tuning," *ArXiv*, vol. abs/2305.06500, 2023.

[180] Y. Zhang, R. Zhang, J. Gu, Y. Zhou, N. Lipka, D. Yang, and T. Sun, "Llavar: Enhanced visual instruction tuning for text-rich image understanding," *CoRR*, vol. abs/2306.17107, 2023.

[181] Y. Hao, H. Song, L. Dong, S. Huang, Z. Chi, W. Wang, S. Ma, and F. Wei, "Language models are general-purpose interfaces," *ArXiv*, vol. abs/2206.06336, 2022.

[182] S. Huang, L. Dong, W. Wang, Y. Hao, S. Singhal, S. Ma, T. Lv, L. Cui, O. K. Mohammed, B. Patra, Q. Liu, K. Aggarwal, Z. Chi, J. Bjorck, V. Chaudhary, S. Som, X. Song, and F. Wei, "Language is not all you need: Aligning perception with language models," *CoRR*, vol. abs/2302.14045, 2023.

[183] Z. Peng, W. Wang, L. Dong, Y. Hao, S. Huang, S. Ma, and F. Wei, "Kosmos-2: Grounding multimodal large language models to the world," *CoRR*, vol. abs/2306.14824, 2023.

[184] A. Zeng, M. Attarian, brian ichter, K. M. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. S. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, and P. Florence, "Socratic models: Composing zero-shot multimodal reasoning with language," in *International Conference on Learning Representations (ICLR)*, 2023.

[185] R. Zhang, X. Hu, B. Li, S. Huang, H. Deng, Y. Qiao, P. Gao, and H. Li, "Prompt, generate, then cache: Cascade of foundation models makes strong few-shot learners," *Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2023.

[186] S. Liu, L. Fan, E. Johns, Z. Yu, C. Xiao, and A. Anandkumar, "Prismer: A vision-language model with an ensemble of experts," *arXiv preprint arXiv:2303.02506*, 2023.

[187] Z. Yang, L. Li, J. Wang, K. Lin, E. Azarnasab, F. Ahmed, Z. Liu, C. Liu, M. Zeng, and L. Wang, "Mm-react: Prompting chatgpt for multimodal reasoning and action," 2023.

[188] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe, "Let's verify step by step," *ArXiv*, vol. abs/2305.20050, 2023.

[189] A. Parisi, Y. Zhao, and N. Fiedel, "Talm: Tool augmented language models," *ArXiv*, vol. abs/2205.12255, 2022.

[190] T. Khot, H. Trivedi, M. Finlayson, Y. Fu, K. Richardson, P. Clark, and A. Sabharwal, "Decomposed prompting: A modular approach for solving complex tasks," in *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2023.

[191] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," *CoRR*, vol. abs/2302.04761, 2023.

[192] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang, "Hugginggpt: Solving AI tasks with chatgpt and its friends in huggingface," *CoRR*, vol. abs/2303.17580, 2023.

[193] P. Lu, B. Peng, H. Cheng, M. Galley, K. Chang, Y. N. Wu, S. Zhu, and J. Gao, "Chameleon: Plug-and-play compositional reasoning with large language models," *CoRR*, vol. abs/2304.09842, 2023.

[194] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian, S. Zhao, R. Tian, R. Xie, J. Zhou, M. Gerstein, D. Li, Z. Liu, and M. Sun, "Toolllm: Facilitating large language models to master 16000+ real-world apis," *CoRR*, vol. abs/2307.16789, 2023.

[195] J. Li, X. Cheng, W. X. Zhao, J. Nie, and J. Wen, "Halueval: A large-scale hallucination evaluation benchmark for large language models," *CoRR*, vol. abs/2305.11747, 2023.

[196] M. Zhang, O. Press, W. Merrill, A. Liu, and N. A. Smith, "How language model hallucinations can snowball," *ArXiv*, vol. abs/2305.13534, 2023.

[197] O. Y. Golovneva, M. Chen, S. Poff, M. Corredor, L. Zettlemoyer, M. Fazel-Zarandi, and A. Celikyilmaz, "Roscoe: A suite of metrics for scoring step-by-step reasoning," *ArXiv*, vol. abs/2212.07919, 2022.

[198] D. N. Ribeiro, S. Wang, X. Ma, H. Zhu, R. Dong, D. Kong, J. Burger, A. Ramos, W. Y. Wang, Z. Huang, G. Karypis, B. Xiang, and D. Roth, "Street: A multi-task structured reasoning and explanation benchmark," *ArXiv*, vol. abs/2302.06729, 2023.

[199] Q. LYU, S. Havaldar, A. Stein, L. Zhang, D. Rao, E. Wong, M. Apidianaki, and C. Callison-Burch, "Faithful chain-of-thought reasoning," *ArXiv*, vol. abs/2301.13379, 2023.

[200] O. Shaikh, H. Zhang, W. B. Held, M. Bernstein, and D. Yang, "On second thought, let's not think step by step! bias and toxicity in zero-shot reasoning," *ArXiv*, vol. abs/2212.08061, 2022.

[201] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, S. Welleck, B. P. Majumder, S. Gupta, A. Yazdanbakhsh, and P. Clark, "Self-refine: Iterative refinement with self-feedback," *ArXiv*, vol. abs/2303.17651, 2023.

[202] N. Shinn, F. Cassano, B. Labash, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," 2023.

[203] S. Ye, Y. Jo, D. Kim, S. Kim, H. Hwang, and M. Seo, "Selfee: Iterative self-revising llm empowered by self-feedback generation," Blog post, May 2023.

[204] H. Yan, S. Srivastava, Y. Tai, S. I. Wang, W. tau Yih, and Z. Yao, "Learning to simulate natural language feedback for interactive semantic parsing," *ArXiv*, vol. abs/2305.08195, 2023.

[205] Z. Chen, Q. Zhou, Y. Shen, Y. Hong, H. Zhang, and C. Gan, "See, think, confirm: Interactive prompting between vision and language models for knowledge-based visual reasoning," *ArXiv*, vol. abs/2301.05226, 2023.

[206] H. You, R. Sun, Z. Wang, L. Chen, G. Wang, H. A. Ayyubi, K.-W. Chang, and S.-F. Chang, "Idealgpt: Iteratively decomposing vision and language reasoning via large language models," 2023.

[207] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning (ICML)*, 2021.

[208] J. Li, D. Li, C. Xiong, and S. C. H. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," in *International Conference on Machine Learning (ICML)*, 2022.

[209] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *Annual Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 11 2019.

[210] J. Li, X. Li, and C. Xie, "Mitigating lies in vision-language models," in *NeurIPS ML Safety Workshop*, 2022.

[211] S. Ye, Y. Xie, D. Chen, Y. Xu, L. Yuan, C. Zhu, and J. Liao, "Improving commonsense in vision-language models via knowledge graph riddles," *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2634–2645, 2022.

[212] G. Ruggeri and D. Nozza, "A multi-dimensional study on bias in vision-language models," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

[213] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," *CoRR*, vol. abs/2305.10601, 2023.

[214] M. Khalifa, L. Logeswaran, M. Lee, H. H. Lee, and L. Wang, "Discriminator-guided multi-step reasoning with language models," *ArXiv*, vol. abs/2305.14934, 2023.

[215] S. Hao, Y. Gu, H. Ma, J. J. Hong, Z. Wang, D. Z. Wang, and Z. Hu, "Reasoning with language model is planning with world model," *ArXiv*, vol. abs/2305.14992, 2023.

[216] A. Graves, "Sequence transduction with recurrent neural networks," *ArXiv*, vol. abs/1211.3711, 2012.

[217] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Audio chord recognition with recurrent neural networks," in *International Society for Music Information Retrieval Conference*, 2013.

[218] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, Montreal, CA, 2014.

[219] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," in *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, ser. Lecture Notes in Computer Science, D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., vol. 8693. Springer, 2014, pp. 740–755.

[220] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*. IEEE, 2021, pp. 9630–9640.

[221] J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, and J. Zhou, "Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond," 2023.

[222] P. Wang, A. Yang, R. Men, J. Lin, S. Bai, Z. Li, J. Ma, C. Zhou, J. Zhou, and H. Yang, "Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework," in *International Conference on Machine Learning (ICML)*, 2022.

[223] T. Brooks, A. Holynski, and A. A. Efros, "Instructpix2pix: Learning to follow image editing instructions," *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18 392–18 402, 2022.

[224] B. Tversky and K. Hemenway, "Objects, parts, and categories." *Journal of experimental psychology. General*, vol. 113 2, pp. 169–97, 1984.

[225] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8748–8763.

[226] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig, "Scaling up visual and vision-language representation learning with noisy text supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 4904–4916.

[227] P. Wang, A. Yang, R. Men, J. Lin, S. Bai, Z. Li, J. Ma, C. Zhou, J. Zhou, and H. Yang, "Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework," in *International Conference on Machine Learning*. PMLR, 2022, pp. 23 318–23 340.

[228] A. Singh, R. Hu, V. Goswami, G. Couairon, W. Galuba, M. Rohrbach, and D. Kiela, "Flava: A foundational language and vision alignment model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 15 638–15 650.

[229] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[230] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," in *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[231] G. Ilharco, R. Zellers, A. Farhadi, and H. Hajishirzi, "Probing contextual language models for common ground with visual representations," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 5367–5377.

[232] C. Zhang, B. Van Durme, Z. Li, and E. Stengel-Eskin, "Visual commonsense in pretrained unimodal and multimodal models," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2022, pp. 5321–5335.

[233] J. Hewitt and P. Liang, "Designing and interpreting probes with control tasks," *Proceedings of the 2019 Con*, 2019.

[234] K. Pham, K. Kafle, Z. Lin, Z. Ding, S. Cohen, Q. Tran, and A. Shrivastava, "Learning to predict visual attributes in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 018–13 028.

[235] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[236] J. He, S. Yang, S. Yang, A. Kortylewski, X. Yuan, J.-N. Chen, S. Liu, C. Yang, Q. Yu, and A. Yuille, "Partimagenet: A large, high-quality dataset of parts," in *European Conference on Computer Vision*. Springer, 2022, pp. 128–145.

[237] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 000–16 009.

[238] X. Chen, S. Xie, and K. He, "An empirical study of training self-supervised vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9640–9649.

[239] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3156–3164.

[240] Y.-C. Chen, L. Li, L. Yu, A. El Kholy, F. Ahmed, Z. Gan, Y. Cheng, and J. Liu, "Uniter: Universal image-text representation learning," in *European conference on computer vision*. Springer, 2020, pp. 104–120.

[241] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[242] L. Yuan, D. Chen, Y.-L. Chen, N. Codella, X. Dai, J. Gao, H. Hu, X. Huang, B. Li, C. Li *et al.*, "Florence: A new foundation model for computer vision," *arXiv preprint arXiv:2111.11432*, 2021.

[243] J. Lu, C. Clark, R. Zellers, R. Mottaghi, and A. Kembhavi, "Unified-io: A unified model for vision, language, and multi-modal tasks," *arXiv preprint arXiv:2206.08916*, 2022.

[244] J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, and Y. Wu, "Coca: Contrastive captioners are image-text foundation models," *arXiv preprint arXiv:2205.01917*, 2022.

[245] Z. Wang, J. Yu, A. W. Yu, Z. Dai, Y. Tsvetkov, and Y. Cao, "Simvlm: Simple visual language model pretraining with weak supervision," *arXiv preprint arXiv:2108.10904*, 2021.

[246] Z. Gan, L. Li, C. Li, L. Wang, Z. Liu, and J. Gao, "Vision-language pre-training: Basics, recent advances, and future trends," *arXiv preprint arXiv:2210.09263*, 2022.

[247] H. Tan and M. Bansal, "Vokenization: Improving language understanding with contextualized, visual-grounded supervision," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 2066–2080.

[248] J. Gordon and B. Van Durme, "Reporting bias and knowledge acquisition," in *Proceedings of the 2013 workshop on Automated knowledge base construction*, 2013, pp. 25–30.

[249] Y. Yang, W. Yao, H. Zhang, X. Wang, D. Yu, and J. Chen, "Z-lavi: Zero-shot language solver fueled by visual imagination," *arXiv preprint arXiv:2210.12261*, 2022.

[250] Z. Tang, J. Cho, H. Tan, and M. Bansal, "Vidlankd: Improving language understanding via video-distilled knowledge transfer," *Advances in Neural Information Processing Systems*, vol. 34, pp. 24 468–24 481, 2021.

[251] X. Shi, I. Padhi, and K. Knight, "Does string-based neural mt learn source syntax?" in *Proceedings of the 2016 conference on empirical methods in natural language processing*, 2016, pp. 1526–1534.

[252] Y. Belinkov and J. Glass, "Analysis methods in neural language processing: A survey," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 49–72, 2019.

[253] Y. Belinkov, N. Durrani, F. Dalvi, H. Sajjad, and J. Glass, "What do neural machine translation models learn about morphology?" in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 861–872.

[254] J. Hewitt and C. D. Manning, "A structural probe for finding syntax in word representations," 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)*, 2019, pp. 4129–4138.

[255] B. Z. Li, M. Nye, and J. Andreas, "Implicit representations of meaning in neural language models," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, pp. 1813–1827.

[256] Y. Adi, E. Kermany, Y. Belinkov, O. Lavi, and Y. Goldberg, "Fine-grained analysis of sentence embeddings using auxiliary prediction tasks," in *International Conference on Learning Representations*, 2017.

[257] I. Tenney, P. Xia, B. Chen, A. Wang, A. Poliak, R. T. McCoy, N. Kim, B. Van Durme, S. R. Bowman, D. Das *et al.*, "What do you learn from context? probing for sentence structure in contextualized word representations," in *International Conference on Learning Representations*, 2019.

[258] E. Salin, B. Farah, S. Ayache, and B. Favre, "Are vision-language transformers learning multimodal representations? a probing perspective," in *AAAI 2022*, 2022.

[259] A. D. Lindström, J. Björklund, S. Bensch, and F. Drewes, "Probing multimodal embeddings for linguistic properties: the visual-semantic case," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 730–744.

[260] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, and K.-W. Chang, "What does bert with vision look at?" in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 5265–5275.

[261] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2020.

[262] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 7262–7272.

[263] Y. Li, H. Mao, R. Girshick, and K. He, "Exploring plain vision transformer backbones for object detection," *arXiv preprint arXiv:2203.16527*, 2022.

[264] A. Van Den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[265] P. Esser, R. Rombach, and B. Ommer, "Taming transformers for high-resolution image synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 12 873–12 883.

[266] S. Bai and S. An, "A survey on automatic image caption generation," *Neurocomputing*, vol. 311, pp. 291–304, 2018.

[267] M. Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga, "A comprehensive survey of deep learning for image captioning," *ACM Computing Surveys (CSUR)*, vol. 51, no. 6, p. 118, 2019.

[268] X. Liu, Q. Xu, and N. Wang, "A survey on deep neural network-based image captioning," *The Visual Computer*, vol. 35, no. 3, pp. 445–470, 2019.

[269] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3128–3137.

[270] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel, "Self-critical sequence training for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7008–7024.

[271] P. Sharma, N. Ding, S. Goodman, and R. Soricut, "Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 2556–2565.

[272] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634.

[273] W. Jiang, L. Ma, Y.-G. Jiang, W. Liu, and T. Zhang, "Recurrent fusion network for image captioning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 499–515.

[274] V. Ordonez, G. Kulkarni, and T. L. Berg, "Im2text: Describing images using 1 million captioned photographs," in *Advances in neural information processing systems*, 2011, pp. 1143–1151.

[275] T. Yao, Y. Pan, Y. Li, and T. Mei, "Exploring visual relationship for image captioning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 684–699.

[276] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[277] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.-S. Chua, "Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5659–5667.

[278] C. Wang, H. Yang, C. Bartz, and C. Meinel, "Image captioning with deep bidirectional lstms," in *Proceedings of the 24th ACM international conference on Multimedia*. ACM, 2016, pp. 988–997.

[279] B. Dai, S. Fidler, R. Urtasun, and D. Lin, "Towards diverse and natural image descriptions via a conditional gan," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2970–2979.

[280] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, "Improved image captioning via policy gradient optimization of spider," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 873–881.

[281] X. Liu, H. Li, J. Shao, D. Chen, and X. Wang, "Show, tell and discriminate: Image captioning by self-retrieval with partially labeled data," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 338–354.

[282] R. Luo, B. Price, S. Cohen, and G. Shakhnarovich, "Discriminability objective for training descriptive captions," in *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition, 2018, pp. 6964–6974.

[283] R. Shetty, M. Rohrbach, L. Anne Hendricks, M. Fritz, and B. Schiele, "Speaking the same language: Matching machine to human captions by adversarial training," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4135–4144.

[284] J. Johnson, A. Karpathy, and L. Fei-Fei, "Densecap: Fully convolutional localization networks for dense captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4565–4574.

[285] D.-J. Kim, J. Choi, T.-H. Oh, and I. S. Kweon, "Dense relational captioning: Triple-stream networks for relationship-based captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6271–6280.

[286] L. Yang, K. Tang, J. Yang, and L.-J. Li, "Dense captioning with joint inference and visual context," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2193–2202.

[287] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, "Image captioning with semantic attention," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4651–4659.

[288] J. Andreas and D. Klein, "Reasoning about pragmatics with neural listeners and speakers," *arXiv preprint arXiv:1604.00562*, 2016.

[289] F. Chen, R. Ji, X. Sun, Y. Wu, and J. Su, "Groupcap: Group-based image captioning with structured relevance and diversity constraints," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1345–1353.

[290] B. Dai and D. Lin, "Contrastive learning for image captioning," in *Advances in Neural Information Processing Systems*, 2017, pp. 898–907.

[291] R. Vedantam, S. Bengio, K. Murphy, D. Parikh, and G. Chechik, "Context-aware captions from context-agnostic supervision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 251–260.

[292] T.-H. Huang, F. Ferraro, N. Mostafazadeh, I. Misra, A. Agrawal, J. Devlin, R. Girshick, X. He, P. Kohli, D. Batra *et al.*, "Visual storytelling," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1233–1239.

[293] D. H. Park, T. Darrell, and A. Rohrbach, "Robust change captioning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4624–4633.

[294] A. Suhr, M. Lewis, J. Yeh, and Y. Artzi, "A corpus of natural language for visual reasoning," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2017, pp. 217–223.

[295] A. Suhr, S. Zhou, A. Zhang, I. Zhang, H. Bai, and Y. Artzi, "A corpus for reasoning about natural language grounded in photographs," *arXiv preprint arXiv:1811.00491*, 2018.

[296] H. Tan, F. Dernoncourt, Z. Lin, T. Bui, and M. Bansal, "Expressing visual relationships via language," *arXiv preprint arXiv:1906.07689*, 2019.

[297] S. Kazemzadeh, V. Ordonez, M. Matten, and T. Berg, "Referitgame: Referring to objects in photographs of natural scenes," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 787–798.

[298] R. Luo and G. Shakhnarovich, "Comprehension-guided referring expressions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7102–7111.

[299] L. Yu, H. Tan, M. Bansal, and T. L. Berg, "A joint speaker-listener-reinforcer model for referring expressions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7282–7290.

[300] L. Yu, P. Poirson, S. Yang, A. C. Berg, and T. L. Berg, "Modeling context in referring expressions," in *European Conference on Computer Vision*. Springer, 2016, pp. 69–85.

[301] J. Mao, J. Huang, A. Toshev, O. Camburu, A. L. Yuille, and K. Murphy, "Generation and comprehension of unambiguous object descriptions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 11–20.

[302] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[303] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, and K.-W. Chang, "Visualbert: A simple and performant baseline for vision and language," *arXiv preprint arXiv:1908.03557*, 2019.

[304] C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid, "Videobert: A joint model for video and language representation learning," *arXiv preprint arXiv:1904.01766*, 2019.

[305] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7794–7803.

[306] Z. Zhu, M. Xu, S. Bai, T. Huang, and X. Bai, "Asymmetric non-local neural networks for semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 593–602.

[307] Y. Li, X. Jin, J. Mei, X. Lian, L. Yang, C. Xie, Q. Yu, Y. Zhou, S. Bai, and A. Yuille, "Neural architecture search for lightweight non-local networks," in *CVPR*, 2020.

[308] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[309] K. Yue, M. Sun, Y. Yuan, F. Zhou, E. Ding, and F. Xu, "Compact generalized non-local network," in *Advances in Neural Information Processing Systems*, 2018, pp. 6510–6519.

[310] M. Dehghani, S. Rothe, E. Alfonseca, and P. Fleury, "Learning to attend, copy, and generate for session-based query suggestion," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 1747–1756.

[311] J.-Y. Jiang and W. Wang, "Rin: Reformulation inference network for context-aware query suggestion," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 2018, pp. 197–206.

[312] A. Sordoni, Y. Bengio, H. Vahabi, C. Lioma, J. Grue Simonsen, and J.-Y. Nie, "A hierarchical recurrent encoder-decoder for generative context-aware query suggestion," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 553–562.

[313] B. Wu, C. Xiong, M. Sun, and Z. Liu, "Query suggestion with feedback memory network," in *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 2018, pp. 1563–1571.

[314] Z.-J. Zha, L. Yang, T. Mei, M. Wang, and Z. Wang, "Visual query suggestion," in *Proceedings of the 17th ACM international conference on Multimedia*. ACM, 2009, pp. 15–24.

[315] Z.-J. Zha, L. Yang, T. Mei, M. Wang, Z. Wang, T.-S. Chua, and X.-S. Hua, "Visual query suggestion: Towards capturing user intent in internet image search," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 6, no. 3, Aug. 2010.

[316] Y.-S. Wang, C. Liu, X. Zeng, and A. Yuille, "Scene graph parsing as dependency parsing," *arXiv preprint arXiv:1803.09189*, 2018.

[317] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[318] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.

[319] R. Vedantam, C. Lawrence Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 4566–4575.

[320] S. Banerjee and A. Lavie, "Meteor: An automatic metric for mt evaluation with improved correlation with human judgments," in *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, 2005, pp. 65–72.

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

[322] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved baselines with visual instruction tuning," *CoRR*, vol. abs/2310.03744, 2023.

[323] D. Zhu, J. Chen, X. Shen, X. Li, and M. Elhoseiny, "Minigpt-4: Enhancing vision-language understanding with advanced large language models," *CoRR*, vol. abs/2304.10592, 2023.

[324] F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and T. Kipf, "Object-centric learning with slot attention," *Advances in Neural Information Processing Systems*, vol. 33, pp. 11 525–11 538, 2020.

[325] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 824–24 837, 2022.

[326] W. Chen, X. Ma, X. Wang, and W. W. Cohen, "Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks," *arXiv preprint arXiv:2211.12588*, 2022.

[327] C.-Y. Hsieh, C.-L. Li, C.-K. Yeh, H. Nakhost, Y. Fujii, A. Ratner, R. Krishna, C.-Y. Lee, and T. Pfister, "Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes," *arXiv preprint arXiv:2305.02301*, 2023.

[328] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," *arXiv preprint arXiv:2203.11171*, 2022.

ZHUOWAN LI

https://lizw14.github.io

RESEARCH INTERESTS

My research interests lie in computer vision and natural language processing, including multimodal models, vision-and-language, compositional reasoning, model robustness, etc. I have experience in pretrained LLMs/VLMs. I am always excited to explore topics like generative AI, embodied AI and foundation models.

EDUCATION

Johns Hopkins University 2018 - 2024
Ph.D. in Computer Science
Advisors: Alan Yuille, Benjamin Van Durme

Tsinghua University 2014 - 2018
B.E. in Electronic Engineering
Second major: Journalism and Communication

RESEARCH EXPERIENCE

Amazon AWS, Santa Clara, CA May 2023 - Nov 2023
Applied scientist intern. Mentors: Bhavan Jasani, Peng Tang, Shabnam Ghadar

· Compositional reasoning for document visual question answering using LLMs.

Facebook , Menlo Park, CA May 2021 - September 2021
Research intern. Mentors: Dhruv Mahajan

· Unsupervised domain adaptation with guidance of pretrained models.

Adobe Research, San Jose, CA May 2019 - Nov 2019
Research intern. Mentors: Quan Tran, Long Mai, Zhe Lin

· Propose new datasets and methods for a new task: context-aware group captioning.
· Published in CVPR 2020.

Sensetime, Beijing, China Oct 2017 - March 2018
Part-time research intern. Mentor: Shuai Yi

· Learn discriminative human representations by reconstructing multi-view images using GAN.
· Published in NeurIPS 2018.

Tsinghua University, Beijing, China Nov 2016 - May 2017
Research Assistant. Advisor: Prof. Shengjin Wang

· Improve person re-identification with fine-grained attributes and pose information.

PUBLICATIONS

Synthesize Step-by-Step: Tools, Templates and LLMs as Data Generators for Reasoning-Based Chart VQA.
Zhuowan Li, Bhavan Jasani, Peng Tang, Shabnam Ghadar.
Under submission (2023)

Causal-CoG: A Causal-Effect Look at Context Generation for Boosting Multi-modal Language Models
Shitian Zhao, **Zhuowan Li**, YadongLu, Alan Yuille, Yan Wang.
Under submission (2023)

ExoViP: Step-by-step Verification and Exploration with Exoskeleton Modules for Compositional Visual Reasoning
Yuxuan Wang, Alan Yuille, **Zhuowan Li***, Zilong Zheng*.
Under submission (2023)

Localization vs. Semantics: How Can Language Benefit Visual Representations Learning?
Zhuowan Li, Cihang Xie, Benjamin Van Durme, Alan Yuille.
EACL 2024

3D-Aware Visual Question Answering about Parts, Poses and Occlusions.
Xingrui Wang, Wufei Ma, **Zhuowan Li**, Adam Kortylewski, Alan Yuille.
NeurIPS 2023

Super-CLEVR: A Virtual Benchmark to Diagnose Domain Robustness in Visual Reasoning.
Zhuowan Li, Xingrui Wang, Elias Stengel-Eskin, Adam Kortylewski, Wufei Ma, Benjamin Van Durme, Alan Yuille.
CVPR 2023 ***Highlight***

Visual Commonsense in Pretrained Unimodal and Multimodal Models.
Chenyu Zhang, Benjamin Van Durme, **Zhuowan Li***, Elias Stengel-Eskin*.
NAACL 2022 ***Oral***

SwapMix: Diagnosing and Regularizing the Over-Reliance on Visual Context in Visual Question Answering.
Vipul Gupta, **Zhuowan Li**, Adam Kortylewski, Chenyu Zhang, Yingwei Li, Alan Yuille.
CVPR 2022

Calibrating Concepts and Operations: Towards Symbolic Reasoning on Real Images.
Zhuowan Li, Elias Stengel-Eskin, Yixiao Zhang, Cihang Xie, Quan Tran, Benjamin Van Durme, Alan Yuille.
ICCV 2021

Context-Aware Group Captioning via Self-Attention and Contrastive Features.
Zhuowan Li, Quan Tran, Long Mai, Zhe Lin, Alan Yuille.
CVPR 2020.

FD-GAN: Pose-guided Feature Distilling GAN for Robust Person Re-identification.
Yixiao Ge*, **Zhuowan Li***, Haiyu Zhao, Guojun Yin, Shuai Yi, Xiaogang Wang, Hongsheng Li.
NeurIPS 2018

PATENTS

Contrastive Captioning for Image Groups. US Patent App. 16/998,876, 2022.
Quan Tran, Long Mai, Zhe Lin, **Zhuowan Li**.

TALKS

Towards Generalizable Visual Reasoning

- Google Research	Nov 2023
- Horizon Robotics	Oct 2023
- ByteDance	Oct 2023
- UC Santa Cruz	Sep 2023
- MIT: Computational Cognitive Science Group (Josh Tenenbaum's group)	May 2023

MENTORING

Vipul Gupta	visiting intern, now PhD at Penn State Univeristy
Chenyu Heidi Zhang	JHU undergrad, now master at Stanford University
Chenyu Zhang	JHU master, now PhD at University of Trento
Xingrui Wang	USC master, now PhD at JHU
Varun Iyer	JHU master
Shitian Zhao	visiting undergrad from ECNU
Yijiang Li	JHU master student
Yuxuan Wang	visiting master student from Peking University

TEACHING

EN.601.461/661. Computer Vision — Johns Hopkins University
Role: Teaching Assistant — Spring 2022
Instructor: Kapil Katyal

SERVICE

Reviewer for CVPR, ICCV, ECCV, NeurIPS, ICML, ICLR.

SKILLS

Programming Languages	Python, MATLAB, C++, C, LaTeX
Deep Learning Tools	Pytorch, Tensorflow, Torch, Caffe