
Answering visual questions through representation learning

Sumit Agarwal¹ Suraj Tripathi¹ Syeda Nahida Akter¹ Andrew Lyubovsky¹ Feng Xiang¹

Abstract

Visual question answering challenges the issue of answering questions about images. Recently, attention networks have been shown to have a strong performance in this task when attending to information from both the vision and the language modality. However, these networks often fail to capture more complex information that is stored within images. In this paper, we propose and evaluate multiple approaches to improving visual question answering models on the Graph Question Answering (GQA) dataset. We do so by leveraging graphical information that is present within the dataset and compare to state-of-the-art baseline models. Through intensive analysis, we achieved significant performance gain in three different directions: coarse-to-fine reasoning networks (2%), prompt tuning (3%), and multi-task learning (3.14%) along with data augmentation methods.

1. Introduction

Although many sophisticated visual QA models have been proposed over the years, they are still far from reaching human performance in the task as they lack effective methods to reason about complex question queries (Section 6). In this paper, we aim to leverage scene graph information from the GQA dataset (Hudson & Manning, 2019a) in order to enhance the models’ understanding of the image and the relationships that are present within the image. We do this by evaluating a novel architecture, teaching the model multiple tasks that require a better understanding of the image, and by simplifying the task that the model needs to solve, allowing it to focus on more complex patterns. Our direction will include designing task-agnostic and task-specific pre-training methods to learn better joint representations, employing prompt-based techniques, enriching input modalities by including scene graph information, adding scene graph formulation, and predicting object attributes as tasks in multi-task learning models, etc. Overall, we aim to explore different techniques suited for visual QA which perform competitively or better than state-of-the-art models. We also check the feasibility of adding generated scene

graphs to VQA models in absence of ground truth graphs and also use scene graphs as an extra modality to help learn in low-resource settings.

Through this paper, we reason around three basic questions:

Q1. How can we combine coarse and fine grained learning for visual QA?

We explore the idea of combining coarse and fine reasoning for visual QA by reasoning over global features like object features, question embeddings using pre-trained vision language models (Tan & Bansal, 2019) and finer instruction vectors over scene graphs. We differ our approach from (Nguyen et al., 2021) by using finer neuro-symbolic features based on scene graphs. Combining the reasoning through a semantic module decides how much of each of the reasoning contribute to the answer.

Q2. How can we design task-specific pretraining strategies for visual QA?

We also wanted to investigate how scene graph and object-specific pre-training tasks would help the model for question answering. In one sitting, we used the LXMERT model and pre-trained it with a multi-label classification task of predicting the important object features pertaining to the question. In another setting, we explored the idea of multi-task learning from (Lu et al., 2019) by designing two new tasks based on scene graphs - Scene graph masking (predicting relations between objects in scene graphs) and Object feature detection (identifying object attributes) to help the model become more aware of the richer details in the image. We observed significant performance improvement over baseline within fewer epochs which denotes cleverly designed tasks enhance model performance as well as enforce faster convergence. We also explore techniques of scene graph generation for images that don’t have ground truth scene graph which can be used in visual QA as an additional modality. Over scene graphs, we also extract synthetic question phrases and answers to help the model achieve better performance in downstream visual QA.

Q3. How can we address visual QA in low-resource settings?

Often visual QA datasets contain a lot of data to train the models and not much work has been done in low-resource

settings. Through this work, we also check the feasibility of doing visual QA with very few examples. Prompt tuning has been explored (Liu et al., 2021) as a technique of adapting pre-trained LMs to downstream tasks via objective engineering, downstream problems are redesigned with a textual hint to seem more like those solved during the original LM training. We explored the idea of using prompt-based initialization using different techniques like question type-based embeddings, scene graph based features, etc. We observed consistent improvement both in low and high resource settings when doing prompt tuning along with finetuning the LM parameters. We observed lower performance when training prompt tuning only models (around 1% trainable parameters) compared to finetuning models but believe that the gap could be reduced with better prompt initializations. We also followed our work to use scene graphs to generate questions that could help in low-resource settings. Our results showed a significant improvement in low-resource settings indicating that these scene graph based QA models could serve as a good initialization for finetuning on downstream tasks with limited labels.

2. Related Work

Multiple datasets have been created for the task of visual QA. However, many early visual QA datasets have been observed to contain biases where the questions themselves were often indicative of the correct answers (Suhr et al., 2018; Hudson & Manning, 2019a). To address these biases, new datasets had been created such as the CLEVR dataset (Johnson et al., 2017), NLVR2 (Suhr et al., 2018), and GQA datasets (Hudson & Manning, 2019a). The NLVR2 removes biases by reformulating the visual QA task to comparing images (Suhr et al., 2018), while the GQA and CLEVR datasets analyze the types of questions that are asked, and balance out questions that could cause biases. We focus on the GQA dataset that uses human-generated scene graphs based on images from the Visual Genome dataset (Krishna et al., 2017). It automates a process by which unbiased questions are generated based on these scene graphs. The questions in the dataset are challenging and often require multi-hop reasoning to arrive at the answer. (Johnson et al., 2017; Hudson & Manning, 2019a).

To solve visual QA, transformer-based models are currently the most common approaches. After the introduction of the Transformer, pre-trained vision-language models started to learn joint representations using language and vision modality cross-attention. They have been shown to perform well on different vision language tasks like image retrieval, visual QA, etc... LXMERT (Tan & Bansal, 2019), ViLBERT (Lu et al., 2019), UNITER (Chen et al., 2020) are some pre-trained models that extract regions of images and attend to them with the text, in our case, the question. OSCAR

(Li et al., 2020) goes one step further and uses object tags along with the image and text to give extra supervision to the model while pre-training. The 12in1 model is trained on different types of vision language tasks and can generalize well on downstream tasks (Lu et al., 2020b).

Detailed information about objects and relations in images is crucial to perform well in multimodal QA task and that can be captured in scene graphs associated with the image. Such scene graphs extracted from image data have been used to reason about questions (Liang et al., 2021). Alternatively, neural module networks have been used where the question is broken down into smaller modules. These models provide better task interpretability. In such cases, the scene graph of the image is used as either a state machine to reason over the image (Hudson & Manning, 2019b) or as a teacher to instantiate the model for learning module functions (Chen et al., 2021). One recent approach also taught a reinforcement learning agent to autonomously generate reasoning paths in a multi-hop manner over the scene graph and served as the basis for deriving answers (Koner et al., 2021).



Question	Coarse	Fine	Coarse+Fine
(Relation based)			
Who in the picture is sitting?	people	woman	woman
(Image based)			
Which place is it?	river	zoo	river

Figure 1 & Table 1. The table shows that where the coarse model fails to answer relation-based questions, the coarse+fine model picks up the answer correctly. For overall object/image-based questions, the fine model predicts the wrong answer which is corrected by the coarse+fine model.

Our proposed approaches build on top of the pretrained language vision models and neuro-symbolic models by learning a joint representation of both the coarser and finer we see that when the coarse model fails to answer finer detailed based complex questions which require a richer understanding of the image, the coarse+fine model makes the right prediction. Additionally, for object-based questions requiring more visual-based understanding, the finer model makes errors that are corrected by the joint model. We also introduce soft prompt-based initialization techniques for visual question answering which have not been addressed in the literature and show how these techniques can lead to improvements in both low-resource and high-resource settings, an observation similar to the NLP domain (Liu et al., 2021). Further, we design scene graph specific pretraining tasks for multi-task learning and show its effectiveness for GQA. We also used scene graphs to augment data with synthetic questions showing improvements in low-resource settings. In situations where ground truth scene graph is not present, we also explore the idea of generating scene graphs and using them as another modality for VQA.

3. Problem Statement

In this work, we tackle the multimodal visual QA task that involves answering a question Q given an image I . The question can be expressed as the sequence (w_1, w_2, \dots, w_n) where w_i is a single word/token. We work with the following representations of the image: **Spatial Feature:** A single vector S for the entire image **Object Features:** A set of N^v vectors $O = (o_1, o_2, \dots, o_{N^v})$ representing N^v objects in the image **Scene graph:** A graphical representation G of the contents of the image.

We work in a setting wherein, given Q and one or more representations of I as input, the model must select an answer for Q from a fixed set of n answers, $A = (A_1, A_2, \dots, A_n)$.

4. Proposed Approach

We proposed five different approaches, all multimodal in nature, that would allow models to understand the interactions between the visual and language modalities better, improving on the visual QA task.

4.1. Coarse to fine reasoning

We propose a model that incorporates both coarse-level and fine-level features and use a semantic reasoning module that selects how much of each of the features to include to answer the question.

To effectively attend to both high level features of the image and low level objects and relations in the image, the model utilizes three steps : Coarse reasoning, Fine reasoning and Semantic Reasoning. Coarse reasoning module performs multimodal reasoning by taking in the object features and question to output a joint (image, text) representation based on a pretrained visual-language model, LXMERT (Tan & Bansal, 2019). Fine reasoning module breaks the question into instruction vectors and passes it iteratively over scene graph through a Graph Attention Network (Veličković et al., 2018) similar to the GraphVQA model (Liang et al., 2021). The semantic reasoning module combines the output of the coarse and fine multimodal learning to predict the answer. The model overview is shown in Figure 2.

Coarse Reasoning - The object vectors O extracted using pre-trained bottom up Faster R-CNN (Anderson et al., 2018) along with corresponding bounding boxes B is passed along with the question Q to a pre-trained vision-language model LXMERT to generate a joint representation $j^{cg} \in \mathbb{R}^{d_{cg}}$, where d_{cg} is the dimension of the joint representation. We use LXMERT as the vision-language pre-trained model because LXMERT also uses question answering as one of its pre-training tasks and hence can generalise better for visual QA.

$$j^{cg} = LXMERT(Q, (O, B)) \quad (1)$$

Fine Reasoning - This module comprises of three main parts - Question Parsing, Scene Graph Encoding, Graph Reasoning.

Question Parsing : This uses a sequence to sequence transformer to translate the question Q into an encoded vector q and sequence of L instructions.

$$q = \text{encoder}(w_1, w_2, \dots, w_n) \quad (2)$$

$$(i_1, i_2, \dots, i_L) = \text{decoder}(q) \quad (3)$$

Scene Graph Encoding : Node features \hat{x}_i of scene graph G is initialized with the word embeddings of the object name and attributes and edge features e_{ij} with the word embeddings of the relation. We then obtain contextualized node embedding for each node i , x_i by :

$$x_i = \sigma\left(\frac{1}{|N_i|} \sum_{j \in N_i} (W_{emb}[\hat{x}_j; e_{ij}])\right) \quad (4)$$

Graph Reasoning : This module executes instruction vectors over the graph neural network in a similar fashion as (Liang et al., 2021), which conditions the message passing in layer L on the L^{th} instruction vector. To achieve that, the L^{th} instruction vector is concatenated to every node and edge before running the L^{th} GNN layer.

$$\hat{x}_j^{(L-1)} = [x_j^{(L-1)}; i^{(L)}] \quad (5)$$

$$\hat{e}_{ij}^{(L-1)} = [e_{ij}^{(L-1)}; i^{(L)}] \quad (6)$$

where $\hat{x}^{(L-1)}$ and $\hat{e}^{(L-1)}$ denote the node features and edge features input to the L^{th} GNN layer. Graph attention network (GAT) is used for the GNN which introduces attention mechanism to learn neighbour weights. The attention score α_{ij} for message passing from node i to node j at L^{th} layer is calculated as :

$$\alpha_{ij}^{(L)} = \text{softmax}_{N_i}(MLP(\hat{x}_i^{(L-1)}; x_j^{(L-1)}; e_{ij}^{(L-1)})) \quad (7)$$

where softmax_{N_i} ensures that the score for one node's neighbour nodes sum to 1. These attention scores are then used to calculate the node embeddings for the next layer.

$$x_i^{(L)} = \sigma\left(\sum_{j \in N_i} \alpha_{ij}^{(L)} W_{GAT}^{(L)} \hat{x}_{ij}^{(L-1)}\right) \quad (8)$$

The final states of each of the nodes after passing through L instruction vectors is weighted through attention scores with the encoded question vector q to get the pooled embedding x which is then concatenated with the question vector q to get the fine representation $j^{fg} \in \mathbb{R}^{d_{fg}}$

$$x_{pool} = \text{attention}(x^{(L)}, q) * x \quad (9)$$

$$j^{fg} = [x_{pool}; q; x_{pool} * q] \quad (10)$$

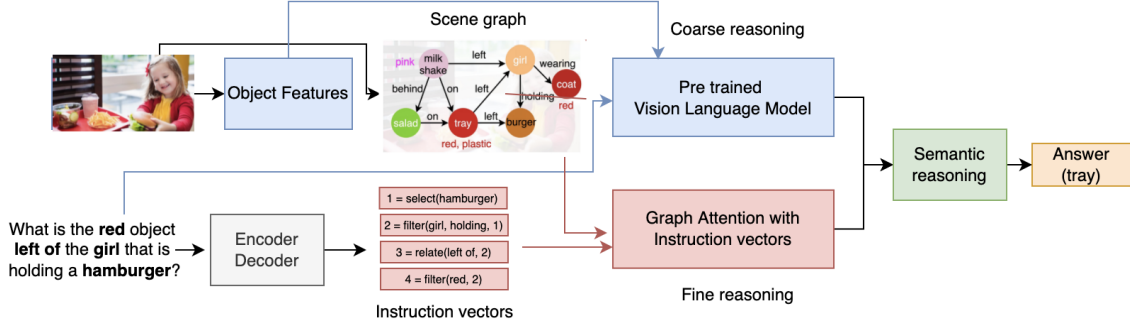


Figure 2. The figure shows the end-to-end coarse to fine reasoning model. The input image has object features and scene graph information. The question is passed through an encoder-decoder to get the instruction vectors which is then iterated over a graph attention network over scene graphs which form the fine-reasoning module. (marked in red). A pre-trained vision language model is used to learn the coarser features using object features and question. (marked in blue). The semantic reasoning learns adaptive weights that sums up to 1 to select from each of the coarse and fine reasoning module to arrive at the probability distribution for the final answer.

Semantic Reasoning The semantic reasoning module is similar to (Nguyen et al., 2021) which learns weights $W \in \mathbb{R}^{|A|}$ to selectively choose coarse features j^{cg} and fine features j^{fg} , where A is the answer set. This joint representation is then used to generate the probability distribution p over A

$$p = \text{softmax}(Wf(j^{cg}) + W'f'(j^{fg})) \quad (11)$$

$$W_a + W'_a = 1 \quad (\forall a \in A) \quad (12)$$

where W and W' are learnable weights of coarse-grained learning (cg) and fine-grained learning; f and f' are learnable projection functions that project $j^{cg} \in \mathbb{R}^{cg}$ and $j^{fg} \in \mathbb{R}^{fg}$ into $p^{cg} \in \mathbb{R}^A$ and $p^{fg} \in \mathbb{R}^A$ respectively. The constraint of the weights mentioned in Eq 11 is maintained by passing the weights over a Softmax function before doing the computation. The end-to-end training of the semantic module with the coarse and fine modules can do away with the noise within each of the individual modules.

4.2. Prompt Tuning

Finetuning has been used to employ large pre-trained models to perform well on downstream tasks. One of the main disadvantages of this approach is that it updates all LM parameters which makes it both computationally and memory intensive. Recent research work (Liu et al., 2021) (Li & Liang, 2021) (Shin et al., 2020) in NLP has proposed a new paradigm named Prompt Tuning which aims to train limited task/input specific prompts while keeping LM parameters fixed to achieve comparable accuracy to the finetuning approach. Our literature study shows that there is limited work that makes use of prompts in multimodal settings.

We believe the reason behind the limited work on prompts in multimodal settings lies in the difficulty of defining/learning prompts that could work well enough to reach comparable

performance as to finetuning approach. Multimodal settings require an end-to-end model to handle more challenges such as alignment, translation, etc. that are not necessarily present in an NLP task. Our work aims to start a discussion of using prompts in multimodal settings and proposing a few directions which are crucial in achieving better performance in prompt tuning only settings.

For prompt learning, we define multimodal QA task $T = \{Q, P, O, A\}$ where Q is the question set, P is the prompt token set, O is the object features set and A is the answer set. As shown in figure 3, question tokens, $q = \{w_1, w_2, \dots, w_{|q|}\}$, are prepended with prompt tokens, $p = \{t_1, t_2, \dots, t_{|p|}\}$, before passing to the embedding layer. Following layers consider prompt tokens as learnable parameters concatenated with input tokens which are used in both self and cross attention layers for making answer predictions. We evaluate our approach with various prompt initializations (general, input specific, different modalities, etc.) in both low and high resource settings.

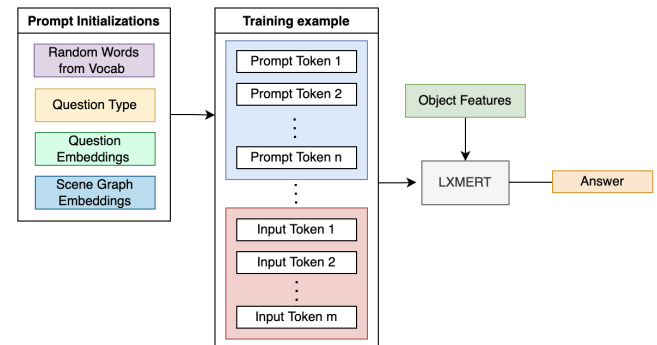


Figure 3. Prompt Based Learning: Prompt tokens initialized with one of the mentioned techniques are prepended to the input question tokens before passing to the embedding layer of the LXMERT model which is then used in both self and cross attention layers along with object features to make answer predictions

4.3. Multitask Training

Pretrained models trained on multiple tasks have shown significant improvements over other state-of-the-art visual QA models (Lu et al., 2020a; Nguyen & Okatani, 2019; Pramanik et al., 2019) on diverse set of visual QA datasets - proposing a generalized QA module. In this paper, we propose a new multi-task model for discriminative vision and language tasks based on the recently proposed MTL model (Lu et al., 2020a) which has been trained jointly on a total of 12 different datasets covering four categories of tasks. While analyzing the type of incorrect labels predicted by state-of-the-art models, we observe two limitations: (1) models are good at detecting the object but not the underlying object features or characteristics, (2) models can not summarize the image due to lack of understanding about relations among all objects in a single image (Appendix 8). Focusing on these limitations, we design two new tasks: *Object Feature Extraction (OFE)* and *Scenegraph Masking (SM)* by introducing two new task-aligned datasets from scene graphs (Table 2). The whole architecture has been trained using *Dynamic Stop-and-Go (DSG)* training module (Appendix 8).

4.3.1. OBJECT FEATURE EXTRACTION (OFE)

In the Visual Genome dataset (Krishna et al., 2017), each object in an image has been annotated with a list of attributes that represents specific characteristics of that object. These features reveal finer details of an object which sometimes contradict with the universal properties (e.g., leaves are not always green) and these features can be retrieved successfully if and only if the models provide explicit attention to individual image rather than memorizing the universal properties without looking into the images. Focusing on these details and the limitation of the previous models, we propose a new task *Object Feature Extraction (OFE)* that ask for specific attributes of an object given the image and the object details. We formulate this task by taking the attribute list of each object from the scene graph and masking one or more attributes for a single object that are needed to be predicted by the models. An object can occur multiple times in a single image (e.g., there can be more than one person in the image). To remove this ambiguity from the dataset, we provide additional details of each object based on the relation of it with other objects in the image. For example, in the example mentioned Table 2, instead of asking about the ‘tree trunk’ directly, we add details ‘to the right of rock’ to pinpoint which specific ‘tree trunk’ we are taking about.

4.3.2. SCENE GRAPH MASKING (SM)

To address the second limitation in the previous models, we again exploit scene graph information from the Visual Genome dataset by proposing another task *Scene Graph*

Masking (SM). Scene graphs contain relations among the objects in a single image (‘fries’ *on* the ‘tray’ in Figure 6) and to get a holistic overview about an object, the models need to understand all the relations that the object shares with other objects. In this task, we take two objects from the scene graph, mask out the relation between them and the models need to predict how these two objects are correlated. Again, an object can occur multiple times in a single image. Similar to OFE task, we remove this ambiguity by adding object features along with the object in the query text which can easily be extracted from the scene graph. In the example mentioned in Table 2, we differentiate between two men by adding their individual features (man: *kneeling*, man: *large*).

4.3.3. VOCAB-BASED VISUAL QA OUTPUT

For both tasks (OFE and SM), we design the datasets as given an image and a natural-language question, select an answer from a fixed vocabulary. We compute an overall image-query representation as an element-wise product between the holistic h_{IMG} and h_{CLS} representations and consider the vocab-based visual QA as a multi-label classification task – assigning a soft target score to each answer based on its relevancy to the ground truth answer. To ensure task specific training, we augment the query with a task token $TASK_t$, so the final input format becomes $\{IMG, v_1, \dots, v_n, CLS, TASK_t, w_1, \dots, w_m, SEP\}$. We compute scores for a set of the pre-defined answers A by using a *two-layer MLP* on top of the overall representation:

$$P(A|I, Q) = \text{sigmoid}(MLP(h_{IMG} \odot h_{CLS})) \quad (13)$$

Due to the answer vocabulary differences, SG and OFE learn separate MLP each.



Task	Train Dataset	Validation Dataset	Example
Scenegraph Masking (SG)	714263	100930	 <p>Input text: ‘The relation between man with the following feature: (kneeling) and man with the following feature: (large) is:’ Output: ‘photographing’</p>
Object Feature Extraction (OFE)	1534871	132062	 <p>Input text: ‘tree trunk, to the right of rock, has following features: tall, ’ Output: ‘gray’</p>

Table 2. Dataset distribution of two new tasks (SG, OFE) along with the example

4.4. Scene Graph Generation

We made use of a pre-trained scene graph generator and pre-trained attribute prediction model to extract object relationships and individual attributes, respectively, from images. Artificially generated scene graphs were then used to train and test a scene graph and language visual QA model. Based on case studies and metrics, we compare the performance of generated scene graph trained models with ground truth based models to showcase performance between the

two, especially in cases where ground truth scene graphs are not inherently available.

4.5. Data Augmentation

We explored data augmentation using scene graph structures that are present in the GQA dataset with the hypothesis that it will help adapt pretrained multimodal QA-based models to a new domain or low-resource dataset efficiently and could further enhance model’s performance by explicitly modeling relations present in scene graphs. Scene graphs structures present in GQA dataset consists of objects, attributes, and relations between objects. For example ”doorway to the left of stairs”, this statement defines relation between two objects: doorway and stairs. Our synthetic question generator will extract two questions from these type of statements involving 1 object each.

This approach follows scene graph generation work as object attributes/relations present in scene graphs are used to generate questions. Scene graphs were generated from images which makes this combined work multimodal in nature.

4.6. ROI Feature Selection

We also design a pre-training task specific to question answering guiding the model to know the importance of each of the objects predicted by the Faster R-CNN model. Often, pre-trained vision language models leave it to the model to attend to particular objects while learning cross attention between vision and language pairs. This can make the model confused and it might attend on lesser important objects to answer the question in hand.

We create a multi-label classification task to map object features extracted from Faster R-CNN to a binary label indicating whether the object is important for the question or not. We check whether the object label o_i is present in the question q and the answer a to create these labels. $O = [o_1, o_2, \dots, o_N] = [0, 1, \dots, 0]$, if $o_2 \in q, a$. We allow the model to learn both this multi-label classification task and the question-answering task together to guide the model better in selecting objects for the question.

$$Loss_{combined} = Loss_{ROI} + Loss_{VQA}$$

5. Experimental Setup

To test the previous approaches, we evaluate our performance on the GQA dataset, as it provides a strong evaluation of the QA task. We also introduce the baselines that we have chosen for our analysis and experimental methodologies for the different approaches. Lastly, we discuss different evaluation metrics that provide insightful information about how our proposed approaches impact the model’s performance.

5.1. Dataset

The GQA(Hudson & Manning, 2019a) dataset was developed based on real-world visual reasoning and compositional QA. The dataset provides annotated scene graph information about Visual Genome (Krishna et al., 2017) images. Many models that had performed well on previous visual QA datasets have been shown to perform poorly on GQA as it requires a more complex understanding of the scene and semantic relationships between objects.

5.2. Multimodal Baseline Models

5.2.1. GRAPHVQA (LIANG ET AL., 2021)

GraphVQA is a language-guided graph neural network that performs instruction vector iterations over scene graphs to arrive at the final answer. This also corresponds to the *FINE* part of the coarse to fine reasoning model.

5.2.2. LXMERT (TAN & BANSAL, 2019)

LXMERT is a large-scale transformer vision language model which models cross-modality interactions using pre-training tasks - Masked Cross-Modality LM, Masked Objective Prediction, Cross-Modality Matching, and Image Question Answering task loss function respectively. Since it is trained on question answering as a pre-training task, we use it as a baseline for most of the approaches. We use 4 different variations of LXMERT -

- a) LXMERT: This is the pretrained LXMERT model trained on multiple objectives as described above.¹.
- b) LXMERT-SUB-36: This is LXMERT finetuned on the *SUB* dataset created from GQA using fixed 36 bottom-up attention (Anderson et al., 2018) object features.
- c) LXMERT-FULL-36: This is LXMERT finetuned on the *FULL* dataset using 36 object features. This also forms the *COARSE* part of the coarse to fine reasoning.
- d) LXMERT-SUB-100: This is LXMERT finetuned on the *SUB* dataset created from GQA using adaptive 10-100 object features given as a part of the GQA dataset.

5.2.3. 12-IN-1 (LU ET AL., 2020B)

12-in-1 proposes a simple multi-task training regime which is based on ViLBERT(Lu et al., 2019) that consists of two parallel BERT-style models operating over image regions and text segments. As in ViLBERT, the 12-in-1 paper takes h_{IMG} and h_{CLS} as holistic image and text representations along with the task-specific ‘head’ network that branches off a common, shared ‘trunk’ ViLBERT model. As such, the model learns shared trunk parameters θ_s and a set of

¹<https://github.com/airsplay/lxmert/>

task-specific layers $\{\theta_t\}_{t=1}^\tau$ for τ tasks. The goal is to learn parameters $\theta_s \cup \{\theta_t\}_{t=1}^\tau$ that minimize loss across all tasks. In each step, the model performs the following estimations:

$$\text{Compute task loss } L_t(\theta) \text{ and gradient } \nabla t(\theta) \quad (14)$$

$$\text{Update } \theta \leftarrow \theta - \epsilon \nabla t(\theta), \text{ where } \theta = \theta_s \cup \theta_t \quad (15)$$

5.3. Methodology

To evaluate our approaches, we collected a subset *SUB* of the dataset by taking questions according to their question type which included 100k questions over 45k images. Balancing the questions according to their question type helps approximate the larger dataset which also helps approximate the answer distribution of the entire dataset. Out of the 100k questions collected we followed a 70-20-10 split for creating the training, validation, and test sets with scene graphs being available only for training and validation data. All questions relating to a particular image belong to the same category of data. For some models, we use the full balanced dataset, *FULL* of 1.7M questions over 150k images because it required end-to-end training.

For coarse to fine reasoning, we use a pre-trained LXMERT (*COARSE*) and GraphVQA model (trained after 10 epochs) (*FINE*) and trained it in an end-to-end fashion with the semantic reasoning module for 15 epochs using an Adam optimizer. We use GLOVE (Pennington et al., 2014) embeddings to initialize the node and edge embeddings for scene graphs and we limit the number of instructions decoded from the question to 5 for the fine learning part. For LXMERT, we use 9 BERT-like layers for the language encoder, 5 for the object-based encoder, and 5 for the cross-modality encoder.

For scene graph generation, we used a pre-trained MotifNet (Tang et al., 2020) to generate scene graphs from image inputs and a bottom-up attention network (Anderson et al., 2018) to predict attributes for each object bounding box and label. MotifNet was pre-trained using a causal-graph-based training framework on the Visual Genome dataset to solidify good object relation bias over negative bias (Zellers et al., 2018). Overall, the scene graph generation pipeline was:

1. Pass images into pre-trained MotifNet to output scene graph object labels, bounding boxes, and object relation labels
2. Pass generated bounding boxes through pre-trained bottom-up attention network to output attribute features of each object
3. Pass attribute-enriched, generated scene graph data to train, validate, and test GraphVQA model

For multi-task training, we used Vilbert (Lu et al., 2019) model pretrained on Conceptual Caption dataset (Sharma

et al., 2018) and trained the model on 12 different datasets for 20 epochs. To obtain the baseline results, we finetuned the trained model on our subset of the GQA dataset (Train: 70k, Validation: 20k, Test: 10k) for 20 epochs. Then we retrain the trained model on different combinations of the proposed tasks on varying numbers of epochs (Section 6.4). Due to limited time and resources, we could not experiment on hyperparameter tuning over these various combinations, which will further improve our current results.

5.4. Evaluation Metrics

Accuracy is the standard metric used to evaluate the visual QA task. We report accuracy validation accuracy in most of the cases since scene graphs are not available for test data. The GQA dataset also includes four metrics to get further insight into the visual reasoning capabilities of the models.

Consistency This metric measures answer consistency across questions that are entailed to a particular question. A model should not answer *green* to a question about the color of an apple which it has just answered as *red*.

Validity and Plausibility This metric checks whether a given answer is in the question scope, e.g. responding some color to a color question. The plausibility score goes a step further, measuring whether the answer makes sense. E.g., red and green as plausible apple colors and, conversely, purple as implausible since red and green occur at least once in the dataset about the question’s subject.

Distribution This metric measures the overall match between the true answer distribution and the model predicted distribution, using Chi-Square statistic. It allows us to see if the model predicts not only the most common answers but also the less frequent ones.

6. Results and Discussion

In this section, we dive deeper to answer the questions that we raised in Section 2. We try to answer why including coarse and fine based learning can help in visual QA. We also discuss results from pre-training and multi-task strategies that we designed for GQA. Additionally, we highlight prompt-based learning techniques for visual QA using different kinds of initialization and show their effectiveness in low-resource settings. We investigate the effect of synthetic data augmentation from scene graphs in pre-trained models as well. Further, we evaluate how in the absence of scene graphs, we can use scene graph generation techniques to achieve better results.

6.1. Combining coarse & fine learning

Table 3 captures the results for the coarse-to-fine model and compares it with only coarse and only fine model settings.

	Accuracy	Binary	Open	Consistency	Validity	Plausibility	Distribution
Only Coarse	83.74	91.32	76.74	94.69	95.42	93.96	1.34
Only Fine	81.77	79.95	83.48	92.61	95.11	93.74	0.57
Coarse + Fine + Semantic Reasoning	85.73	84.41	86.97	94.68	95.33	94.50	0.29

Table 3. The table consists of GQA specific evaluation metrics across three models, only coarse, only fine and coarse+fine+semantic reasoning model. It shows an increase in accuracy which is basically accounted by an increase in the open type question accuracy.

We see that there is an increase in 10% over the coarse model and 3% over the fine model for open-type questions which leads to an increase in the overall accuracy for the model. The consistency metric is also similar to the coarse model which shows that the model can predict the same answers for similar questions. The model makes more plausible answers than others which also accounts for the fact that the model looks for both coarse and fine grained features. The model prediction distribution is also closest to the true answer distribution which shows the model predicts rare answers as well.

Figure 4 also does a question type based analysis supporting the claim in Table 1 showing that the combined model performs better on relation and object-based questions learning from both coarse and fine multimodal features. Relation questions like "What is hanging on the wall?" and query based questions like "What is the refrigerator?" require detailed information from scene graphs to know what is happening in the image and hence the finer model performs better. Logical and object based questions like "Are there both sand and grass in this photo?" require object features to arrive at the answer which is why the coarse model has higher accuracy than the fine one. The Coarse+Fine model combines the best of both worlds by choosing how much of each of the model feature to select to arrive at the answer.

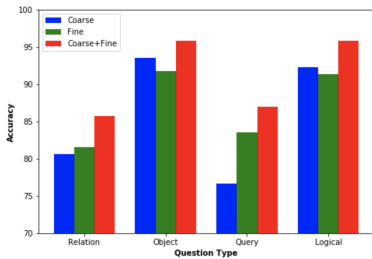


Figure 4. The image shows the comparison of accuracy values among different type of questions showing that the coarse+fine model performs better on most of the question types.

6.2. Prompt based learning

Due to the limited time for this project, we decided to evaluate the LXMERT-SUB-100 model on 4 different prompt initialization settings, refer to figure 3, involving both textual and visual modalities. Further, we perform our evaluation in low and high resource settings. Following are the 4 prompt initialization strategies that we use in this work. **Random Word:** Initialization with randomly ex-

tracted word embeddings from the vocab **Question Type Embeddings:** The GQA dataset has different question types which are categorized by high-level classes called Semantic, Structural, Detailed. We make use of Structural categorization for this initialization which is further divided into 5 sub-classes; ['verify', 'query', 'choose', 'logical', 'compare'] **Question Embeddings:** Initialization with pretrained GraphVQA model question embeddings **Scenegraph Embeddings:** Initialization with pretrained GraphVQA model scene graph embeddings

We also experimented with initializing prompts with word embeddings corresponding to objects present in the scene graph of the image but this approach didn't improve the performance when used along with finetuning. One potential reason could be for this is that there were many objects present in the images and therefore multiple prompt tokens were prepended to the input embeddings. But analysis of questions indicates that most of the questions are around only a few objects present in the image and therefore adding all the objects could have introduced unnecessary noise.

We do prompt initialization with different modalities for example words, question type from language modality, and scene graph embeddings from visual modality which makes this work multimodal.

As mentioned in table 4 and 5, in both low and high resource settings, we observed an improvement (compared to baseline in first row) by using prompt tuning along with finetuning which indicates that prompts could help improve model's performance in multimodal settings. In particular, initializing prompts with input-specific embeddings led to better performance when compared with random word-based initialization. Furthermore, an improvement of around 3% in a low resource setting indicates that models with prompt tokens could adapt to new domains efficiently. Further, we believe that better prompt initialization will help in reducing the gap between Finetuning and Prompt Tuning. In prompt tuning only setting, input specific prompt initialization techniques such as question and scene graph embeddings performed worse than the other two proposed techniques. Our analysis showed that as these prompt tokens are specific to an example, they are seen only once during an epoch by the model, and therefore the model is not able to tune these prompt parameters well to perform better. We believe using cluster-based techniques and then assigning cluster-specific question/scene graph embeddings will result

in a better performance.

Finetune	Prompt Tuning	Prompt Initialization	Validation Accuracy
Yes	No	-	59.52
Yes	Yes	Random Word	61.66
Yes	Yes	Question Type	61.66
Yes	Yes	Question	62.47
Yes	Yes	Scene Graph	63.54
No	Yes	Random Word	50.13
No	Yes	Question Type	51.74
No	Yes	Question	48.63
No	Yes	Scene Graph	49.60

Table 4. Validation accuracy of Prompt based strategies in low resource setting on GQA dataset: LXMERT-SUB-100 model

6.3. Data Augmentation: Scene Graph Based QA Task

For the task of scene graph based QA, we generated 850K unique training questions from a subset of scene graph structures present in the GQA dataset. Due to computational constraints, we used a smaller set of 150K training questions to pre-train LXMERT-SUB-100 model before finetuning it on the original GQA task.

The most common relations that were present in the generated scene graph based QA set were "to the left of", "to the right of", "on top of", "filled with", "on/in", etc. We posed these questions in a way that the output is an object. For example, as mentioned in table 7, answers are graffiti, train, which satisfies some relation with another object. Some of these questions may not sound syntactically correct but we believe answering these questions makes a model capture all the objects and relations between them and in turn provides a good initialization for the downstream multimodal QA task. We also plan to augment this data to predict relations between given objects in the input image. Further, the use of image-based scene graphs to generate questions makes this work multimodal.

Results obtained by finetuning on scene graph QA-based pre-trained model are presented in table 6. We observed consistent improvement using the pre-training model from this task. We trained on a smaller subset of 150K training examples due to computational constraints and we believe that increasing the number of examples will lead to an increase in improvement on downstream QA tasks. In the current experimental setting, this approach performs much better in the low resource domain indicating that this approach leads to a better parameter initialization for finetuning in the presence of limited data. Further, it signifies the importance of pre-training on similar tasks before finetuning.

6.4. Multi-Task Learning

To experiment with the effectiveness of our newly proposed two tasks in the multi-task training regime, we select the state-of-the-art MTL model Vilbert:12-in-1 (Lu et al., 2020a). Due to time and resource constraints, we chose a subset of our new datasets for experimentation and ran the

Finetune	Prompt Tuning	Prompt Initialization	Validation Accuracy
Yes	No	-	70.68
Yes	Yes	Random Word	71.08
Yes	Yes	Question Type	71.05
Yes	Yes	Question	71.59
Yes	Yes	Scene Graph	71.57
No	Yes	Random Word	61.38
No	Yes	Question Type	59.71

Table 5. Validation accuracy of Prompt based strategies on GQA dataset: LXMERT-SUB-100 model

Approach	Low resource setting	Validation Accuracy
Baseline	Yes	59.52
Proposed	Yes	62.73
Baseline	No	70.68
Proposed	No	71.59

Table 6. Validation accuracy using scene graph based QA pre-trained model on GQA dataset: LXMERT-SUB-100 model

model on fewer training/finetuning epochs so we can accommodate different combinations of tasks. Table 8 provides the size of the training and validation datasets for every combination we tried for this project.

Initially, we trained the model on each task individually (SM, OFE) and then finetuned the trained model on downstream QA task using the GQA dataset. Without any supervision of the GQA dataset itself, we got comparable accuracy to the baseline within a few steps - showing that our task individually allows the model to converge faster. Then we train the model combining two tasks (SG + OFE), which gets comparable test accuracy to the baseline within a single training-finetuning epoch. Lastly, we add the GQA dataset for supervision during multi-task training along with the two other tasks and our final module (GQA + SG + OFE) beats the baseline accuracy by 10.07% within a single training-finetuning epoch, proving the high effectiveness of our proposed tasks. Additionally, we have trained the model with different iteration gaps (Δ) on the DSG module (Section 8.3.1) where we observed that higher Δ / more sparse updates improve performance initially but degrade for larger values (in our setup, $\Delta = 4$ gave the best performance).

6.5. Scene Graph Generation

Performance results of three GraphVQA models on the validation and test subsets are shown in Table 9. The test accuracy score of the ground truth scene graph model utilized a scene graph embedding of zero because the dataset did



Question	Answer
to the right of doorway	graffiti
in front of building	trains
to the left of stairs	doorway

Table 7. Synthetically generated Question-Answer pairs using scene graph

Training Condition	Train/Val Dataset	Training/Finetuning Epochs	Validation Accuracy	Test Accuracy
Baseline	70k/20k	20/20	76.93%	58.37%
SM	114k/80k	20/1	69.58%	55.87%
OFE	153k/25k	5/5	66.06%	53.01%
SM + OFE	114k/80k + 53k/10k	1/1	70.61%	56.57%
GQA + SM + OFE	(1)	1/1	80.1%	58.01%

Table 8. MTL performance on GQA dataset trained on two new tasks (individually and combined); (1) combined size of 3

Scene Graph Type	Training Epochs	Validation Subset Acc.	Test Subset Acc.
Ground Truth (GT)	40 GT (sub.)	59.51%	31.61%
Generated w/ Att.	10 GT (full) / 20 Gen. (sub.)	49.89%	42.93%
Generated w/o Att.	10 GT (full) / 20 Gen. (sub.)	48.57	42.66%

Table 9. GraphVQA performance based on scene graph inputs (full = full training set / sub. = partial training set). NOTE: GT model did not access scene graph embeddings for test accuracy run.



	GT GraphVQA	Gen. GraphVQA w/ Attr.
Is the car in the top of the image?	no	yes

Figure 5. (a) Ground truth scene graph provided by dataset (b) Generated scene graph from scene graph generation pipeline (c) Sample image question and model predictions

not provide ground truth scene graphs for the test split for the model to use. Overall, the performance of the generated scene graph models lagged behind the ground truth trained counterpart during validation. However, performance stayed relatively consistent between validation and test accuracy for the generated scene graph models. Something that cannot be said for the ground truth scene graph trained model.

One interesting case where the generated scene graph trained GraphVQA model performed better than its ground truth counterpart is shown in Figure 5. When analyzing why the ground truth model failed to correctly predict the answer, it can be seen when looking at the ground truth relations data for the target car is shown below:

```
{'name': 'car',
 'relations': [
   {'object': 'building_1', 'name': 'to the left of'},
   {'object': 'people_1', 'name': 'to the left of'},
   {'object': 'car_1', 'name': 'to the right of'},
   {'object': 'car_2', 'name': 'to the right of'},
   {'object': 'car_3', 'name': 'to the right of'}]}
```

In the effort to answer if the car is at the top of the image, it may have been difficult to parse through the scene graph and

contextualize position from only left and right directional relationships. Looking at the generated scene graph relations data, the use of the relation label 'behind' for the macro objects in the foreground certainly may have aided in better predicting that the car is located in the background and at the top of the image. Generated scene graph information for the target car is shown below:

```
{'name': 'car',
 'relations': [
   {'object': 'man_1', 'name': 'behind'},
   {'object': 'woman_1', 'name': 'behind'},
   {'object': 'dog_1', 'name': 'behind'},
   {'object': 'shirt_1', 'name': 'behind'}]}
```

6.6. ROI Feature Selection

Approach	Validation Accuracy	Test Accuracy
LXMERT-SUB-36	82.45	59
LMXERT-SUB-36 + ROI	83.88	59.32

Table 10. Accuracy using ROI based pretraining on LXMERT-SUB-36 model

Following on the ROI feature selection task, we used the LXMERT-SUB-36 as one of the baselines and we performed the ROI task along with the QA task in LXMERT-SUB-36 + ROI. Table 10 shows an increase of around 1.5% in the validation accuracy highlighting the fact that the model is learning to attend to objects better.

7. Conclusion and Future Directions

In this paper, we analyze the limitations of the state-of-the-art visual question answering models and provide experimental results for four major approaches covering: (a) combining coarse and fine grained learning (2) designing new tasks for pretraining and multi-task training and (3) prompt based learning in both low and high resource settings and (4) scene graph based data augmentation. We observed consistent improvement across all mentioned approaches on the GQA task. In the current semantic reasoning module, we incorporated a simple weight based fusion of the features. As a future work, attention based and more complicated fusion architectures can be explored. The Coarse+Fine is confused in binary questions which can be modeled as a different classifier separate from open type questions. For prompt based learning techniques, we explore adding soft prompts to the textual modality. So, we can analyze if adding prompts to visual modality or a combination can be a more effective choice. Further, we plan to make use of cluster specific question/scene graph embeddings for prompts initialization and reparameterization (Li & Liang, 2021) technique to make prompt tuning more stable. For multi-task training, we plan to explore different scheduling algorithms other than DSG (e.g., curriculum learning) to mitigate overfitting in terms of overlapping features and want to introduce contrastive loss to learn from the differences which will improve performance in terms of new tasks/datasets.

References

- Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., and Zhang, L. Bottom-up and top-down attention for image captioning and visual question answering, 2018.
- Chen, W., Gan, Z., Li, L., Cheng, Y., Wang, W., and Liu, J. Meta module network for compositional visual reasoning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 655–664, 2021.
- Chen, Y.-C., Li, L., Yu, L., Kholy, A. E., Ahmed, F., Gan, Z., Cheng, Y., and Liu, J. UNITER: Universal image-text representation learning. In *Proceedings of the European conference on computer vision*, 2020.
- Hudson, D. A. and Manning, C. D. 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*, pp. 6700–6709, 2019a.
- Hudson, D. A. and Manning, C. D. Learning by abstraction: The neural state machine, 2019b.
- Johnson, J., Hariharan, B., Van Der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., and Girshick, R. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2901–2910, 2017.
- Koner, R., Li, H., Hildebrandt, M., Das, D., Tresp, V., and Günnemann, S. Graphhopper: Multi-hop scene graph reasoning for visual question answering, 2021.
- Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.-J., Shamma, D. A., et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.
- Li, X., Yin, X., Li, C., Zhang, P., Hu, X., Zhang, L., Wang, L., Hu, H., Dong, L., Wei, F., Choi, Y., and Gao, J. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *ECCV*, August 2020.
- Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. *CoRR*, abs/2101.00190, 2021.
- Liang, W., Jiang, Y., and Liu, Z. GraphVQA: Language-guided graph neural networks for graph-based visual question answering. In *Proceedings of the Third Workshop on Multimodal Artificial Intelligence*, pp. 79–86, 2021.
- Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., and Neubig, G. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586, 2021.
- Lu, J., Batra, D., Parikh, D., and Lee, S. ViLBERT: Pre-training task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pp. 13–23, 2019.
- Lu, J., Goswami, V., Rohrbach, M., Parikh, D., and Lee, S. 12-in-1: Multi-task vision and language representation learning. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020a.
- Lu, J., Goswami, V., Rohrbach, M., Parikh, D., and Lee, S. 12-in-1: Multi-task vision and language representation learning. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020b.
- Nguyen, B. X., Do, T., Tran, H., Tjiputra, E., Tran, Q. D., and Nguyen, A. Coarse-to-fine reasoning for visual question answering, 2021.
- Nguyen, D.-K. and Okatani, T. Multi-task learning of hierarchical vision-language representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10492–10501, 2019.
- Pennington, J., Socher, R., and Manning, C. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL <https://aclanthology.org/D14-1162>.
- Pramanik, S., Agrawal, P., and Hussain, A. Omninet: A unified architecture for multi-modal multi-task learning. *arXiv preprint arXiv:1907.07804*, 2019.
- Sharma, P., Ding, N., Goodman, S., and Soricut, R. 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*, pp. 2556–2565, 2018.
- Shin, T., Razeghi, Y., IV, R. L. L., Wallace, E., and Singh, S. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2020.
- Suhr, A., Zhou, S., Zhang, I., Bai, H., and Artzi, Y. A corpus for reasoning about natural language grounded in photographs. *CoRR*, abs/1811.00491, 2018.

Tan, H. and Bansal, M. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.

Tang, K., Niu, Y., Huang, J., Shi, J., and Zhang, H. Un-biased scene graph generation from biased training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3716–3725, 2020.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks, 2018.

Zellers, R., Yatskar, M., Thomson, S., and Choi, Y. Neural motifs: Scene graph parsing with global context, 2018.

8. Appendix

8.1. Scene graph definition

We formally define the scene graph consisting of V nodes representing objects in the image and E edges representing relations between objects. N^v is the number of objects in the image, and N^e is the number of relations. v_i^{name} is the name of the object corresponding to the node v_i and v_i^{attr} is the set of attributes of the object. Similarly, e_j^{name} the name of the relation between a source object/node e_j^s and a receiver object e_j^r .

$$Q = (w_1, w_2, \dots, w_n), I = (S, O, G) \quad (16)$$

$$G = (E, V) \quad (17)$$

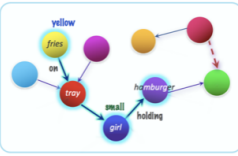
$$V = (v_1, v_2, \dots, v_{N^v}) \quad (18)$$

$$v_i = (v_i^{name}, v_i^{attr}) \quad (19)$$

$$E = (e_1, e_2, \dots, e_{N^e}); e_j = (e_j^{name}, e_j^s, e_j^r) \quad (20)$$

$$e_j^s | e_j^r \in V \quad (21)$$

8.2. GQA



What color is the food on the red object left of the small girl that is holding a hamburger, yellow or brown?

Select: hamburger → Relate: girl, holding → Filter size: small → Relate: object, left → Filter color: red → Relate: food, on → Choose color: yellow | brown

Figure 6. A typical example present in the GQA dataset with the image, corresponding question, their programs and the scene graph.

8.3. Questions and model-specific answers

Table 11 shows different types of questions and the answers predicted by different models. The question is chosen in such a way to show the variety of different answers that the models predict which are wrong.

8.3.1. DYNAMIC STOP-AND-GO (DSG)

Many visual and language tasks are built using the similar data sources and share significant overlap in terms of individual images. Multi-task learning allows models to share parameters among different tasks and enables models to learn about a similar input e.g., an image from multiple views. Most of the time, heavy pre-trained models overfit when trained separately in downstream tasks due to small or imbalanced datasets. Multi-task learning enables interaction among different tasks, but at the same time training a single model on multiple datasets of different complexities and sizes is not an easy task. Simply cycling through all tasks may drastically overtrain smaller tasks leading to overfitting which eliminates the positive effect of MTL. To mitigate the overfitting problem Lu et al. (2020a) proposed cleverly designed MTL training module *Dynamic Stop-and-Go (DSG)* (Algorithm 8) which works as a regularizer. DSG monitors the validation loss s_t of each task t and considers a task as *Converged*, if performance improvement is less than 0.1% over 2 epochs and shift it into stop mode. In DSG *stop* mode, a task only updates every iter-gap (Δ) iterations. If validation performance degrades by 0.5% from the task’s best measured performance while in *stop* mode, the task is considered *Diverged* and is returned to DSG *go*.

Algorithm 1 DSG for Multi-Task Learning

```

 $n_t \leftarrow$  number of iterations per epoch for task  $t$ 
 $\Delta \leftarrow$  size of gap between iterations in stop mode
 $DSG_t \leftarrow$  go
for  $i \leftarrow 1$  to  $MaxIter$  do
    for  $t \in Tasks$  do
        if  $DSG_t = go$  or ( $DSG_t = stop$  and  $i \% \Delta = 0$ ) then
            Compute task loss  $L_t(\theta)$  and gradient  $\nabla_t(\theta)$ 
             $\theta \leftarrow \theta - \epsilon \nabla_t(\theta)$ , where  $\theta = \theta_s \cup \theta$ 
        end
        if  $i \% n_t = 0$  then
            Compute validation score  $s_t$  on task  $t$ 
            if  $DSG_t = go$  and  $Converged(s_t)$  then
                 $DSG_t \leftarrow stop$ 
            else if  $DSG_t = stop$  and not  $Converged(s_t)$  then
                 $DSG_t \leftarrow go$ 
        end
    end
end
    
```

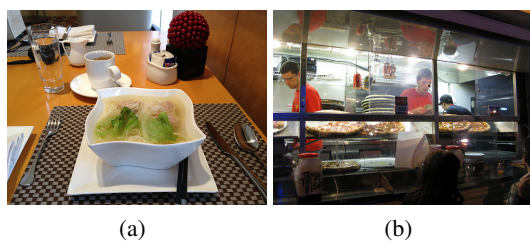



Figure 7. (a) **Question:** ‘The noodles in what?’, **Prediction:** ‘bowl’, **Answer:** ‘soup’ (b) **Question:** ‘Who is wearing a hat?’, **Prediction:** ‘man’ (‘worker’ for GraphVQA), **Answer:** ‘chef’