Multihop Multimodal QA using Joint Attentive Training and Hierarchical Attentive Vision Language transformers

Deigant Yadava (dyadava)* Joao Monteiro (jmonteir)* Vinay Nair (vinayn)* Dheeraj Pai (dmohanda)*

Abstract

In this paper we address the challenges in Multihop and Multimodal Question Answering (MMQA). Through the analysis of existing MMQA approaches, we identify alignment between multimodal data and reasoning as the bottleneck in MMQA systems. We hypothesize that jointly learning to predict relevant patches from an image along with predicting an answer can prevent models from over-fitting by forcing it to learn relationships between image sections and the question, subsequently improving the reasoning process. In this paper we provide the details as well as analysis of three proposed approaches. We show that explicitly learning the alignment between images and text can allow multimodal models to focus on properties such as "color" and "shape" in images for VQA tasks. Our code can be found here. [1]

1. Introduction

Multihop and multimodal question answering (MMQA) is the cognitive process of retrieving and combining relevant information from diverse knowledge sources (e.g., textual, visual, and audio) to answer a given question. Humans routinely face MMQA problems in everyday life. For instance, when diagnosing a patient, doctors must reason over multiple medical exams (i.e., distinct knowledge sources) to answer the question "Does my patient have condition X?" The development of an artificial intelligence (AI) system that can automatically and accurately solve MMQA problems would be a groundbreaking breakthrough. However, despite the remarkable progress in AI, there is still a long way to go before AI systems can robustly solve MMQA problems.

MMQA problems are usually solved in two stages: (a) retrieval and (b) reasoning. In the retrieval stage the goal is to identify the most relevant knowledge sources when presented with a question of interest and the set of all candidate knowledge sources. The goal in the reasoning stage is to combine information from the relevant sources to answer the question. While most approaches do fairly well in retrieving the relevant sources, reasoning is still a bottleneck in the development of robust systems that can handle MMQA problems. We hypothesize that the alignment between question and the patches of the images in these approaches is not optimal, since most MMQA models typically learn to align question and image representations implicitly, through the use of contrastive learning or other techniques.

In this work, we plan to explicitly learn alignment between question and the image patches. We think that learning image representation that explicitly capture this alignment can help capture information that can be used by the model to answer the question and remove any noise/distraction.

This paper is organized as following. Section 2 presents an overview of related work. Section 3 introduces our proposed approaches. Section 4 states our research questions, provides information on datasets, baseline models and experimental methodology. Section 5 presents our results. Finally, Section 6 shows our conclusions and future research.

2. Related Work

Multi-modal Multi-hop question answering on WebQA: In-order to deal with multi-modal data for multi-hop reasoning, Solar (Yu et al., 2023) proposes to represent images and tables using textual captions/descriptions. These textual descriptions are then fed into a language model along with the query to generate the answer. On the other hand, (Yang et al., 2022) proposes to extract entities and relations from the given dataset and encode them along with the embedding of the images and text in a graphical form. These representations are then passed through a decoder to generate the answer as well as identify the relevant information. (Chen et al., 2022) recognizes that answering some of the questions required more world knowledge that was present in the provided data. They proposed MuRAG, which maintained a multi-modal memory of world facts that might be required for answering questions. These facts are retrieved at inference time to help in the generation of the answer.

Based on these previous implementations, at a high level, the task of multi-hop question answering can be divided into

[1] * Equal contribution. See Appendix A

retrieval and reasoning.

Retrieval: In this step, usually similarity scores between the question and candidate knowledge sources are computed. (Yu et al., 2023) uses BERT (Devlin et al., 2018) embeddings of image captions to calculate the similarity score. (Chang et al., 2022) represent images with 100 regions predicted by an object detection model, training a fully connected layer to align the image representation with BERT-tokenized questions. (Liu et al., 2022) learn a universal embedding space by optimizing vision-language representations contrastively.

Reasoning with Large Language Models (LLM): Given the ability of LLMs to efficiently process and reason with text, PromptCAP (Hu et al., 2022) and ImgLLM (Guo et al., 2023) proposed to convert the images to captions such that the captions can be processed easily by LLMs and subsequently better help answer the given query. On a similar note, (Cheng et al., 2023) proposed to utilize world knowledge to create specific image captions. Alternatively, (Merullo et al., 2022) proposed to make use of visual encoders to convert the image to an embedding and then use a neural network to convert this embedding into the language space. These captions or embeddings can then passed to a language model for generating the answering. (Lightman et al., 2023) demonstrate that Large language models perform better in reasoning tasks when we divide the objective into sub-tasks and perform well in each of the sub-tasks.

Other Methods of Reasoning: Some other methods of reasoning include making use of Graph neural network to reason over components/elements in images(Yasunaga et al., 2021; Liang et al., 2021; Yang et al., 2022), using step-by-step chain of thought/planning based methods to improve the reasoning process (Zhang et al., 2023b; Wu et al., 2023; Huang et al., 2022; Lu et al., 2022) and using pre-trained Vision-language models to perform VQA tasks (Wang et al., 2021; Zhang et al., 2023a; Chen et al., 2020; Kim et al., 2021; Wang et al., 2022). Please refer to Appendix E for more information.

To our knowledge, this paper is the first to propose a method to solve the questions in the WebQA dataset that makes use of explicit alignment between images and the text and then uses this aligned information to reason about the answer.

3. Proposed Approaches

Through the error analysis of the baselines models we note that a major area where all the models struggle is creating an alignment between the relevant areas in the image with the question that can help in answering the question. In most cases, the models seem to over-fit and collapse to predicting the mode of the training distribution. Hence, in this project we explore three different ideas through which we can explicitly learn to better align the questions with the parts of the images and then use only the most relevant parts to reason about the answer. In the following subsections, we will focus on the multi-modal task of generating an answer from a set of relevant images. We assume that the set of relevant images has already been extracted.

3.1. Notation

Throughout this report, we use the following notation :

1. H is the set of questions in the dataset.
2. Q_h denotes the h-th question of the dataset.
3. $I_h = \{I_{h,1}, \cdots, I_{h,n_I(h)}\}$ denotes the set of images associated with h-th question, where $h \in \{1, \cdots, |H|\}$ and $n_I(h)$ is the number of images associated with the h-th question.
4. $C_h = \{C_{h,1}, \cdots, C_{h,n_I(h)}\}$ denotes the set of image captions associated with h-th question, where $h \in \{1, \cdots, |H|\}$ and $n_I(h)$ is the number of images associated with the h-th question.
5. $I_{h,i}$ denotes the i-th image associated with the h-th question.
6. $C_{h,i}$ denotes the caption associated with the i-th image of the h-th question
7. $I^j_{h,i}$ denotes the j-th patch of the i-th image of the h-th question.

3.2. Attentive Patching

In this approach, we hypothesize that patching the image and attending to each patch cross-modally helps us find image representations that takes in consideration the most relevant patches to answer the question of interest. The general idea is to reduce the amount of noise generated by attending to unnecessary parts of the image. This approach is comprised of two separate models: (1) **Patch Selector** (to identify the most relevant image patches) and (2) **Answer Generator** (to generate an answer given the image representations along with original image captions and the question of interest). Figure 1 depicts an overview of the Attentive Patching approach. Note the Patch Selector and the Answer Generator models are trained separately.

3.2.1. Model Description

Patch Selector. To identify the relevant patches of the image, the patches from each image are fed into a vision language transformer encoder model along with the question. We use ViLT (Kim et al., 2021) as our vision language transformer encoder model. The input to the Patch Selector model for the h-th question consists of a set of tokens for the question, image caption, and an image.

A classification head is added to the output model that predicts the probability of each fixed-size patch in the input

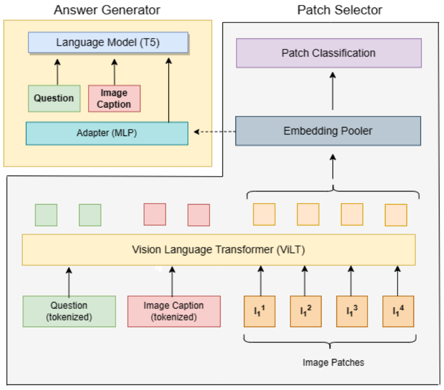


Figure 1. Attentive Patching approach.

image being relevant. We fix the patch size in such a way that the input image is divided into a 7 × 7 grid of patches.

Answer Generator. Once the Patch Selector model has been trained, we utilize the Patch Selector as an encoder. In particular, we pass the question of interest and the associated images to the Patch Selector and we use the pooler output from ViLT to represent the relevant content in the images. Let $R_{h,i}$ denote the representation from the pooling layer of the Patch Selector for the i-th image associated with the h-question. We pass $R_{h,i}$ through an adapter to convert it into the language space: $A_{h,i} = \texttt{Adapter}(R_{h,i})$. We create a prompt for the language model, where we pass the embedded questions and original captions and also $A_{h,1}, \cdots, A_{h,n_I(h)}$. Notice that $A_{h,i}$ is one token in the language model. We use T5 (Raffel et al., 2020) as our language model, and for the Adapter we considered a feed-forward neural network with three layers, where each layer comprises a linear transformation followed by the GELU (Gaussian Error Linear Unit) activation function.

3.2.2. Learning

Patch Selector. We train the Patch Selector in a supervised manner, where the binary cross entropy loss is used for optimizing the parameters in the Patch Selector model:

$$\mathcal{L}_{ps} = -\sum_{h,i,j} \left[y^j_{h,i} \log(p^j_{h,i}) + (1 - y^j_{h,i}) \log(1 - p^j_{h,i}) \right]$$

where $p^j_{h,i}$ denotes the probability of the j-th patch in the i-th image for the h-th question be relevant and $y^j_{h,i}$ denotes the true label (0: non-relevant, 1: relevant) for that patch.

Answer Generation. The image representations from the Patch Selector (converted into the language space), the original image captions along with the question of interest are fed into T5 to generate the answer. A standard masked language modelling loss is used for this task as shown in the equation below.

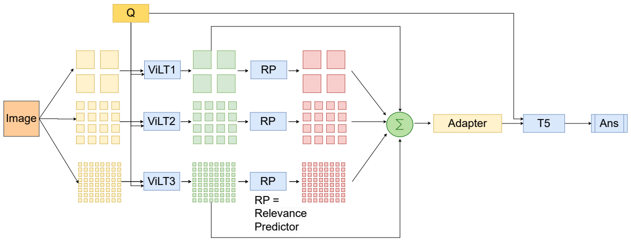


Figure 2. Pyramidal Patch Networks during inference.

$$\mathcal{L}_{as} = -\sum_{h=1}^{|H|} \sum_{k=1}^{T_h} \log P \left(a^k_h | Q_h, C_h, A_h, a^{1:k-1}_h \right)$$

where a^k_h denotes the k-th token of the answer of the h-th question, A_h denotes the set of image embeddings in the language space for the h-th question, T_h is the number of tokens in the answer for the h-th question.

3.3. Pyramidal Patch Networks

One disadvantage of dividing an image into fixed size patches is that it is hard to determine how large the relevant patches would be before seeing the question. A good example would be of a bird that is extremely small as compared to all other objects in the patch. In this case, we would benefit more by taking a small sub-patch of the original fixed size patch and then try to detect what's happening in that sub-patch.

3.3.1. Model Description

The Pyramidal Networks approach, instead of fixed-size patches, divides the image into independent grids of size 2 × 2, 4 × 4, and 8 × 8. This division results in patches of different sizes, denoted as level 3, level 2, and level 1 divisions for 2 × 2, 4 × 4, and 8 × 8 grids, respectively. We independently train three models to predict the probability of each patch's relevance to the question. Similar to the Attentive Patching approach, Pyramidal Networks use ViLT as the relevance prediction network, taking the question and image patch as input. For each image patch, we obtain one embedding with an associated weight (total of 84 embeddings). A weighted sum of patches yields a single embedding vector for the image. This vector, combined with the question, is input into a language model to generate the answer.This is inspired by Feature Pyramid Networks (Lin et al., 2017).

3.3.2. Learning

Data Annotation: The data annotation process (described in section 4.2) involved resized images divided into 7 × 7 grids (224 × 224 pixels). To extend this annotation to various grid sizes (2 × 2, 4 × 4, and 8 × 8) in the original full-size image, we reverted each 224 × 224 annotated image to its original dimensions. The source image was segmented into a 7 × 7 grid, aligning with the annotated data patches. We assigned a value of 1 to pixels in relevant patches and 0 to others, maintaining the original image size with pixel values of 0 or 1 indicating relevance to the question. Afterward, we applied max-pooling to this binary pixel map with kernel sizes computed in such a wat that this process produced 8 × 8, 4 × 4, and 2 × 2 attention grids, where each grid had a value of 1 for patches containing relevant data and 0 otherwise. These grids serve as labels for image patches obtained by dividing the image into 2 × 2, 4 × 4, and 8 × 8 grids, indicating the relevance of each patch to the question.

Relevance Predictor: With the preparation of our 2×2, 4×4, and 8×8 annotated data grids, we facilitate the training of three distinct relevance prediction models, each designed to operate at a different scale.

We train the Relevance predictor models independently in a supervised manner where the binary cross entropy loss is used. The loss for each model is calculated as following:

$$\mathcal{L}_{rp} = -\sum_{h,i,j} \left[y^j_{h,i} \log(p^j_{h,i}) + (1 - y^j_{h,i}) \log(1 - p^j_{h,i}) \right]$$

where $p^j_{h,i}$ denotes the probability of the j-th patch in the i-th image for the h-th question being relevant and $y^j_{h,i}$ denotes the true label (0: non-relevant, 1: relevant) for that patch. $j \in \{1, \cdots, 4\}$ for level 3 , $j \in \{1, \cdots, 16\}$ for level 2 and $j \in \{1, \cdots, 64\}$ for level 1 division.

Multilevel Dense Representations: The novel part of this approach as compared to the previous Patch Selector approach discussed is the inference. At inference time, we divide the original image into 2×2; 4×4 and 8×8 grids. Then we pass these patches to the appropriate relevance prediction models. For each patch, the relevance prediction models output an embedding along with the relevance weight for the patch. We perform a weighted sum of the embeddings to get the final representation of the image (Figure 2).

Answer Generation: The final image representation obtained in the previous step is passed through an adapter which is then fed into the T5 model along with the question for generating the answer. A standard masked language modelling loss is used for this task as shown in the following equation

$$\mathcal{L}_{as} = -\sum_{h=1}^{|H|} \sum_{k=1}^{T_h} \log P\left(a^k_h | Q_h, A_h, a^{1:k-1}_h\right)$$

where a^k_h denotes the k-th token of the answer of the h-th question, A_h denotes the set of image embeddings in the language space for the h-th question, T_h is the number of tokens in the answer for the h-th question.

3.4. Joint optimization of patch detection and answer generation

3.4.1. Model Description

In this idea, given a question and a set of relevant images, we plan to jointly optimize the prediction of the relevant (fixed size) patches in the image as well as maximizing the log-likelihood of the generated answer given the image sources and question.

Figure 3 provides an overview of the network architecture for this approach. It can be summarized in a sequence of steps: First, the question is tokenized, and the image is segmented into patches. The tokenized inputs are concatenated and passed through a visual language transformer. Subsequently, contextualized pooled embeddings of the image patches and the questions are used to predict the relevant patches for the question. In parallel, we make use of cross-attention to merge the question embeddings with the image patch embeddings. The final embeddings from the preceding step (cross-attention) as well as the raw image embeddings are converted to the space of a regular language model using an adapter. Finally, these converted embeddings are fed into a language model to generate the answer.

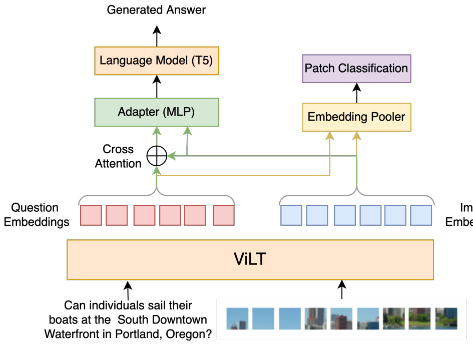


Figure 3. Jointly optimising patch prediction and answer generation.

The input to the model will consist of the tokenized question and a set of related images (all positive images are concatenated together). The output from the vision language transformer will consist of a set of question embeddings and image embeddings. Let eQ^j_h represent the embedding of the

j-th token of the question and $eI^j_{h,i}$ represent the embedding of the j-th patch of the i-th question.

Predicting the relevant patches: To predict the relevant patches, the embeddings of the image patches are pooled along with the embeddings of the questions. These pooled embedding are then passed through a classification head to predict all the patches which are relevant for the question. In our implementation, the classification layer accepts the pooled embeddings and then provides one output for each image patch in the input (indicating the probability that the patch is relevant for the question):

$$p^j_{h,i} = \sigma(\texttt{Linear}(\texttt{Pooled Embedding}_h)_{i,j})$$

where $p^j_{h,i}$ denotes the probability of j-th patch in i-th image for the h-th question being relevant for the question. Pooled Embedding$_h$ denotes the pooled embedding generated from the h-th question along with the corresponding image patches. $\texttt{Linear}(\texttt{Pooled Embedding}_h)_{i,j}$ represents the output from the linear layer corresponding to the j-th patch of the i-th image for the h-th question.

Generating the answer: We will make use of cross-attention between the question and the image embeddings to further contextualize the embeddings of the question with the information from the image as shown below.

$$eQ_h^{j*} = eQ_h^j + \Sigma_k w_k * (eI^k_{h,i})$$

The contextualized question embeddings eQ_i^{j*} along with the image embeddings are fed into an adapter. In our case, the adapter consists of a 3-layer MLP with GELU activation. The output of the adapter is fed into a language model to generate the answer.

3.4.2. Learning

Predicting the relevant patches: To train the classification head to predict the relevant patches we make use of the labels for the patches that are available in the manually annotated dataset. The binary cross entropy loss is used for this part of the model:

$$\mathcal{L}_1 = -\sum_{h,i,j}\left(y^j_{h,i}\log(p^j_{h,i}) + (1-y^j_{h,i})\log(1-p^j_{h,i})\right)$$

where $y^j_{h,i}$ denotes the true label (0: non-relevant, 1: relevant) of the j-th patch in the i-th image.

Generating the answer: In order to fine-tune the language model to generate the answer using the embeddings provided by the adapter, we make use of the standard Masked Language Modelling loss (with the golden answers in the data-set as the labels).

$$\mathcal{L}_2 = -\sum_h\sum_{k=1}^{T}\log(P(a_h^{k*}|eQ_h^*, eI_h, a_i^{\{1:k-1\}}))$$

where a_i^{k*} is the target answer token at time-step k. eQ_h^* is the set of contextualized question embeddings and eI_h is the set of image embeddings.

Combined Loss: The combined loss for the joint optimization of the network is $\mathcal{L} = \lambda\mathcal{L}_1 + (1-\lambda)\mathcal{L}_2$ where λ will be a hyperparameter.

3.5. Chain of Thought Reasoning using MiniGPT-4

3.5.1. Model Description

We experimented with Chain of Thought Reasoning using MiniGPT4 (Zhu et al., 2023). This model consists of a frozen Vision Transformer (Dosovitskiy et al., 2020). The image embeddings are fed through a linear adapter layer before feeding it to a fine-tuned Llama-V2 (7B) Language Model (Touvron et al., 2023) along with the related text. This model has been fine-tuned end-to-end to perform a variety of vision language tasks.

In our approach, we feed in the question along with the set of related images directly to the MiniGPT-4 model. We try a vanilla approach in which we prompt the model only with the question and the images and a chain-of-thought based approach in which we prompt the model with prefixes such as “think step by step” or “explain your reasoning” to see the impact of chain-of-though reasoning on the performance of question answering.

3.5.2. Learning

Since this is a zero shot approach there is no learning happening as we are not fine-tuning the model for our dataset. The zero shot approach involved passing the image along with the question into the model and prompting it to answer the question in as concise a way as possible; thus trying to reduce any noise from the same.

4. Experimental Setup

4.1. Research Questions

In this project our goal is to answer the following research questions:

1. Does teaching the model to focus on the relevant parts of the image based on the question improve its ability to answer questions?
2. Does having high level and low level information represented together augment alignment of Question and Image?
3. Is a linear adapter enough to translate the information as well as alignment generated in the embedding space of ViLT for answer generation in T5?

4.2. Dataset and Input Modalities

The WebQA dataset (Chang et al., 2022) is a MMQA dataset that contains questions, knowledge sources (images, captions and text snippets) and the golden answers. This dataset contains image-based questions and text-based questions. The image-based questions can be based on one or two images: double-image questions require stitching two images to answer the question of interest, while single-image questions are more complex. The text-based questions involve combining knowledge from at least two text snippets. Note, WebQA does not contain questions that require image and text snippets as knowledge sources.

In this project we use a subset of the WebQA dataset. In particular, we only consider the image-based questions (i.e. which contain image and text modality both) . Additionally, we assume we know the positive knowledge sources (i.e. we do not perform retrieval). This subset contains 21,465 samples, of which 18,954 samples are used for training and 2,511 samples for validation.

To train the Patch Selector in a supervised manner, we annotated the patches of 2,940 images from 2018 (9.4%) questions where each image was resized to 224 × 224 pixels and split into 49 patches, where each patch is 32 × 32 pixels. Therefore, we have the ground truth (i.e. relevant versus not relevant) for each patch in the annotated images. To facilitate the annotation we developed a simple React and flask app where the annotator only had to click the patches relevant to the question (Figure 10).

4.3. Multimodal baseline models

4.3.1. Solar

Instead of directly dealing with non-textual data, Solar (Yu et al., 2023) represents the input images as textual descriptions. To ensure that there is minimal loss of information, Solar generates two types of captions: one at a high level generically defining the image and one at a low level defining the objects and their properties.

In the context of the WebQA dataset, Solar consists of two parts: re-ranking and answer generation. The re-ranking module helps in filtering the source and selecting the most relevant ones. A simple neural network model (on top of the BERT embeddings) is used to compute the closeness score between the query and the sources. This neural network is trained using the binary cross entropy loss. Finally, once the most relevant set of sources are selected, the query along with the relevant sources are fed into a T5 transformer model (Roberts et al., 2019) which generates the answer. During training, the T5 model is trained with the ground truth answers. For more information about Solar please see Appendix C

4.3.2. Structured Knowledge and Unified Retrieval-Generation (SKURG)

SKURG (Yang et al., 2022) proposes a two-stage approach:

1. Entity-centered Fusion Encoder: This stage aligns sources from different modalities by focusing on shared entities. This allows the model to capture relationships between different pieces of information and build a comprehensive understanding of the context.
2. Unified Retrieval-Generation Decoder: This stage integrates intermediate retrieval results with the answer generation process, allowing the model to dynamically adjust the number of retrieval steps and generate answers based on the most relevant information.

The training of SKURG involves optimizing various loss functions responsible for alignment between sources and entities, confidence estimation of connecting the source to the knowledge graph, retrieval and answer generation. For details see Appendix B.

4.4. VLP + x101FPN

The VLP + x101FPN baseline proposed in (Chang et al., 2022) makes use of two separate VLP (Zhou et al., 2019) models for the sub-problems of identifying relevant clues and answering questions in the WebQA dataset.

To form the input to the model, text inputs were tokenized using BERT (Devlin et al., 2018), while each image was characterized by 100 regions extracted from a variant of the Faster RCNN architecture. In the retrieval/re-ranking process, a VLP model processes candidate clues and a question, outputting probabilities of relevance.For answer generation, another VLP model uses relevant clues to generate answers. Please refer to Appendix D for more information.

4.5. Experimental Methodology

We used the same training validation split provided in the WebQA dataset. For evaluation of the baseline models, we locally computed the accuracy and the fluency for each question in the validation dataset. The locally computed metrics are approximate since we do not have access to the exact reference solutions/techniques that are used by WebQA.

The following experiments (hyper-parameters) were used for executing the ideas :

1. **Attentive Patching.** We used the Adam optimizer (`lr=5e-5, batch size=16`) for both the Patch Selector (trained for 5 epochs) and Answer Generator models (trained for 30 epochs). In our experiments, we tested a model without fine-tuning the ViLT model for creating image embeddings and compared its perfor-

Table 1. Overall performance (%) for answering image-based questions in the WebQA validation dataset.

Model	QA-Fl ↑	QA-Acc ↑	QA ↑
VLP + x101FPN	36.0	45.0	16.2
SKURG	**52.5**	**63.8**	**36.3**
Solar	46.0	60.1	27.6
Attentive Patching	51.7	59.3	33.3
Pyramidal Networks	47.4	45.4	21.7
Joint Patching Answering	34.9	52.7	19.9

mance with our approach that involves using embeddings from a fine-tuned ViLT model.

2. **Pyramidal Networks:** Relevance prediction models in Pyramidal Networks were trained using Adam optimizer (`lr=2e-5, batch size=32`) for 20 epochs. The answer generator was trained for 30 epochs (`lr=5e-5, batch size=16`). To test our research question, we experimented by replacing image embeddings in the final language model with zeros and random numbers. This helped assess the impact of hierarchical embeddings on improving question answering.
3. **Joint Optimization** The joint optimization model was trained using the Adam optimizer (`lr=5e-5, batch size=32`). Additional experiments were conducted:
   (a) **Different Learning Rates:** We explored a learning rate of `1e-3` for the adapter and `5e-5` for the language model and ViLT model. This did not enhance the model's learning, possibly due to convergence to the same final loss over epochs.
   (b) **Contextualized Image Embeddings Only:** We investigated using only contextualized image embeddings, excluding question embeddings, as input to the language model. Refer to the results in the next section.
   (c) **Multi-layer Adapter:** Instead of a linear adapter, we experimented with a multi-layer adapter. Refer to the results in the next section.

5. Results and Discussion

5.1. Overall Performance and Analysis of Ideas

Table 1 displays the overall performance in answering image-based question in the WebQA validation datset for all proposed approaches and baselines.

5.1.1. Analysis: Joint optimization of Patch classification and Answer Generation

High Level Discussion. Table 2 provides a comparison between the performance of our model and the current state of the art (Solar). From the improved performance in the question of the "color" and "shape" category, see that our model is actually able to capture the elements of color and shape from the image(s) that are being passed. The increase in the performance of the model could be because our approach actually considers the embeddings from the images instead of the just making use of the captions from images (like Solar). The embeddings from the image would contain more information as compared to the limited information in the caption which can be utilised by the language model to answer the question.

Table 2. Accuracy comparison between Ours and Solar.

Category	Accuracy (Ours)	Accuracy (Ours Without classification)	Accuracy (Solar)
Others	0.72	0.77	0.77
YesNo	0.33	0.31	0.42
Shape	**0.22**	**0.21**	**0.14**
Color	**0.36**	**0.31**	**0.32**
Choose	0.76	0.75	0.85
Number	0.38	0.36	0.42

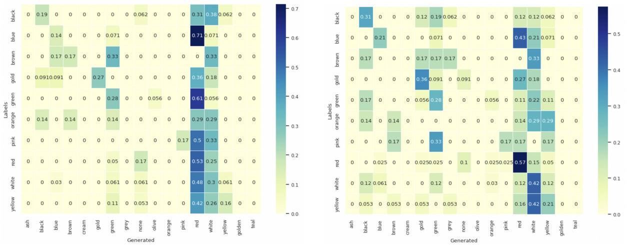


Figure 4. Color based Confusion Matrices (a) left: Solar (b) right: Ours

Further, in Figure 4 we see the comparison of the confusion matrix of Solar with our method (color based questions). We see that while Solar tends to predict the majority class as compared to our method in which we can see more diversity in the predictions as well as the diagonal forming in the confusion matrix.

Discussion; Impact of joint optimization: To assess the importance of optimizing both image patch classification and answer generation, we conducted an ablation study in which the model was optimised with as well as with-

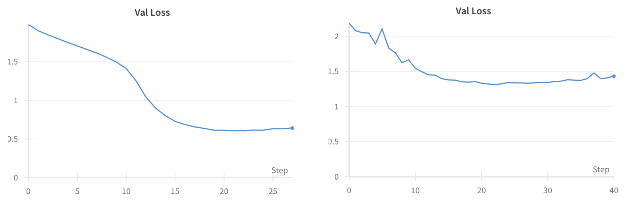


Figure 5. (a)left: Validation loss with joint optimization (b) right: Validation loss with contextualized image embeddings only

out the classification objective. The model's performance, especially in "color" category questions, improved when incorporating the classification objective (Table 2). This enhancement suggests that the classification objective helps the ViLT model generate image patch representations more relevant to questions, reducing noise and enabling the language model to produce better answers.

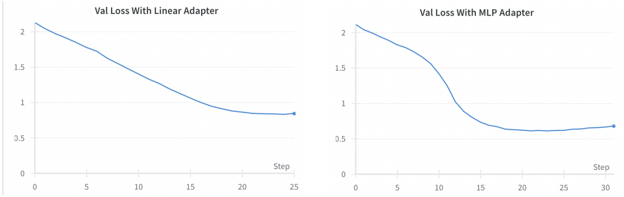


Figure 6. (a) left: Performance with linear adapter (b) right: Performance with MLP adapter

Discussion; Impact of the adapter architecture We examined the impact of adapter architecture on model performance using both linear and multi-layer adapters. A 3-layer MLP adapter with GELU activation proved more effective in converting from ViLT's embedding space to T5(Figure 6) as compared to a linear adapter. The improved performance with the MLP adapter may be due to the WebQA dataset's inclusion of proper nouns, where precise spellings are important for minimizing loss. The non-linear MLP adapter captures these intricacies better than a linear one.

Discussion; Importance of the question in answering To test if the model's focus on the image can be improved, we conducted an ablation by contextualizing image embeddings with question embeddings using cross-attention. The resulting embeddings, excluding the question embeddings, were fed into the language model. Figure 5(b) displays the validation loss for the network trained with these contextualized image embeddings. The model's performance was not as good as the one shown in 5(a). This difference may be due to the WebQA dataset, where answers often require direct information from the questions. Contextualizing image embeddings with questions may lead to a loss of information, impacting the model's ability to construct accurate answers. Refining the dataset to minimize redundancy between questions and answers could address this issue.

5.1.2. Analysis: Attentive Patching

Discussion; Importance of the Patch Selector. To test whether the Patch Selector helps with find better representation for the image(s) and consequently generate better answers, we conducted an ablation study in which the Answer Generator model receives image embeddings from the Patch Selector as originally designed (i.e. with Patch Selector), or directly from ViLT without any fine-tuning (i.e. without Patch Selector). Table 3 shows the overall performance for answering image-based questions in the WebQA validation dataset for the Attentive Patching approach with and without the Patch Selector component. We observed that the QA accuracy and QA Fluency are higher without the Patch Selector. This is somewhat surprising, as at first these results suggest the Patch Selector might not help in generate the answer.

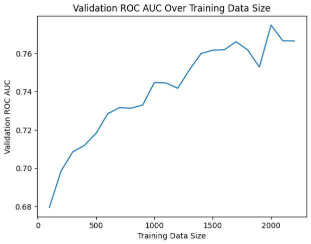


Figure 7. AUC for training Patch Selector under different training data sizes.

Table 3. Overall performance (%) for answering image-based questions in the WebQA validation dataset - Attentive Patching with and without the Patch Selector component.

Patch Selector	QA-Fl ↑	QA-Acc ↑	QA ↑
On	**51.71**	59.28	33.27
Off	51.56	**61.99**	**34.88**

To understand better the reason behind the small decrease in performance when using the Patch Selector, we computed the QA accuracy by type of question (Table 4). We notice the Attentive Patching with Patch Selector had a considerable drop in performance specially in color type questions (from 45.84% to 26.80%). This could be because the the best Patch Selector we trained yield an area under of ROC curve of only 0.78 (see Figure 11 in Appendix H). Consequently, the Attentive Patching with a Patch Selector that does not discriminate as well might find some colors important for the task of classifying the patch as relevant to the question, but those questions might not actually be relevant to the question. Figure 7 shows the AUC for the Patch Selector models trained on different data sizes. Notice the AUC increases with data sizes, which suggests with higher training dataset we would have found a better Patch Selector.

Table 4. QA Accuracy (%) for answering image-based questions in the WebQA validation dataset - Attentive Patching with and without the Patch Selector component.

Patch Selector	Y/N	MC	Color	Shape	Number	Other
On	45.17	**85.98**	26.80	**17.34**	**43.71**	76.06
Off	**50.84**	84.43	**45.84**	15.77	41.11	**76.46**

5.1.3. Analysis: Hierarchical Patch relevance prediction and Answer Generation

High Level Discussion. Table 5 compares the Hierarchical Patching method with other baselines and our ideas, trained on 10% of the WebQA dataset due to time constraints. Results suggest its performance is not as strong, potentially due to limited training data and challenges in adapter training convergence (because the model is over-fitting on the textual data). To enhance performance, we propose training on more data (entire WebQA dataset) and pre-training the adapter with text embeddings from ViLT and T5 embedding layers.

Discussion; Importance of making use of Hierarchical image embeddings: To evaluate the combined image representation, we trained a network using random noise instead of the hierarchical network's image representation. Both networks were trained with identical hyperparameters and the same number of training steps for a fair comparison. Results indicate improved performance in the color category but a decrease in other categories when using a valid image representation. We suspect this is due to the T5 model's limitation in capturing all image semantics with just one token.

Table 5. Accuracy comparison between with the image representation vs random noise representation.

Category	Accuracy (Ours)	Accuracy (Random noise)
Others	0.69	0.71
YesNo	0.22	0.37
Shape	0.10	0.20
Color	**0.17**	**0.15**
Choose	0.83	0.85
Number	0.38	0.41

5.1.4. MiniGPT4 Zero Shot Analysis

Table 6. Accuracy with chain of thought reasoning (MiniGPT-4) for different question categories

Others	YesNo	Shape	Color	Choose	Number
0.8364	0.3047	0.2197	0.3346	0.9287	0.2182

We observed that even with Chain of Thought Reasoning; the MiniGPT4 model was not performing well with zero shot prompting (Table 6). Better reasoning alone is not enough to solve the WebQA task. Chain of thought reasoning using MiniGPT-4 led to errors when the model did not focus on the right parts/elements of the image. The model, we hypothesize, is biased in some way to the data it was pre-trained on (see appendix F for more qualitative examples). The other approaches mentioned in this paper can augment such chain-of-thought reasoning based methods by improving the alignment between images and text.

5.2. Research Question 1: Does teaching the model to focus help generating answers?

In the previous section, we discussed joint optimization of the model to concentrate on specific image regions as well as generating the answers. This approach improved the model's performance, especially in answering questions about "color" and "shape". By allowing the language model to focus solely on relevant image parts, it can better predict accurate color or shape information without being distracted by other scene elements. Additionally, we suggest that further improving the model's performance is possible by pre-training the T5 language model on visual tasks like captioning and VQA. This pre-training can enhance the model's ability to recognize features from image embeddings. Overall for this research question we conclude that teaching the model to concentrate on relevant image parts improves its accuracy in answering questions.

5.3. Research Question 2: Does hierarchical information represented together augment alignment of Question and Image?

From the analysis of the Pyramidal patch networks, we do not see a major improvement in the accuracy of the model with the use of embeddings that were created in a hierarchical method. This could be because our model is still over-fitting on the textual data that is being provided and the adapter to convert the embeddings from the space of ViLT to T5 is not being trained to convergence before the T5 model over-fits. This means that the T5 model is not able to make use of the information from the images. Hence, the second research question at this point is inconclusive. One way to be able to improve the model could be to pre-train the adapter with text embedding pairs from ViLT as well as T5. This could ensure that the T5 model is able to make use of information from the images to answer the question.

5.4. Research Question 3 : How efficient is a linear adapter?

Through the analysis of the joint optimization approach in the previous section we see that a multi-layer non-linear adapter is actually better in our case since it is able to capture more intricate details as compared to a linear adapter. Hence, we conclude that in case the problem statement requires intricate details (such as spellings etc.) to be captured, a non-linear multi-layer adapter is better.

6. Conclusion and Future Directions

This paper introduces three novel methods to explicitly enhance multimodal question answering models by refining image-text alignment. Additionally, it investigates Chain of Thought Reasoning with MiniGPT-4. Ablation studies demonstrate that the proposed methods improve alignment, leading to enhanced reasoning. Notably, improvements are observed in question categories requiring focus on object colors and shapes. Future research directions include: (1) pre-training T5/language-model on vision-language tasks, (2) utilizing image segmentation to identify relevant continuous segments, (3) using CLIP embeddings for improved image-text embedding matching, and (4) dataset curation to remove redundant wording from gold standard answers.

References

Chang, Y., Narang, M., Suzuki, H., Cao, G., Gao, J., and Bisk, Y. Webqa: Multihop and multimodal qa, 2022.

Chen, W., Hu, H., Chen, X., Verga, P., and Cohen, W. W. Murag: Multimodal retrieval-augmented generator for open question answering over images and text. *arXiv preprint arXiv:2210.02928*, 2022.

Chen, Y.-C., Li, L., Yu, L., El Kholy, A., Ahmed, F., Gan, Z., Cheng, Y., and Liu, J. Uniter: Universal image-text representation learning. In *European conference on computer vision*, pp. 104–120. Springer, 2020.

Cheng, K., Song, W., Ma, Z., Zhu, W., Zhu, Z., and Zhang, J. Beyond generic: Enhancing image captioning with real-world knowledge using vision-language pre-training model. *arXiv preprint arXiv:2308.01126*, 2023.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Guo, J., Li, J., Li, D., Tiong, A. M. H., Li, B., Tao, D., and Hoi, S. From images to textual prompts: Zero-shot visual question answering with frozen large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10867–10877, 2023.

Hu, Y., Hua, H., Yang, Z., Shi, W., Smith, N. A., and Luo, J. Promptcap: Prompt-guided task-aware image captioning. *arXiv preprint arXiv:2211.09699*, 2022.

Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

Kim, W., Son, B., and Kim, I. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pp. 5583–5594. PMLR, 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*, 123:32–73, 2017.

Liang, W., Jiang, Y., and Liu, Z. Graghvqa: language-guided graph neural networks for graph-based visual question answering. *arXiv preprint arXiv:2104.10283*, 2021.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

Liu, Z., Xiong, C., Lv, Y., Liu, Z., and Yu, G. Universal vision-language dense retrieval: Learning a unified representation space for multi-modal retrieval. In *The Eleventh International Conference on Learning Representations*, 2022.

Lu, P., Mishra, S., Xia, T., Qiu, L., Chang, K.-W., Zhu, S.-C., Tafjord, O., Clark, P., and Kalyan, A. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

Merullo, J., Castricato, L., Eickhoff, C., and Pavlick, E. Linearly mapping from image to text space. *arXiv preprint arXiv:2209.15162*, 2022.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Roberts, A., Raffel, C., Lee, K., Matena, M., Shazeer, N., Liu, P. J., Narang, S., Li, W., and Zhou, Y. Exploring the limits of transfer learning with a unified text-to-text transformer. 2019.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Wang, P., Yang, A., Men, R., Lin, J., Bai, S., Li, Z., Ma, J., Zhou, C., Zhou, J., and Yang, H. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *International Conference on Machine Learning*, pp. 23318–23340. PMLR, 2022.

Wang, Z., Yu, J., Yu, A. W., Dai, Z., Tsvetkov, Y., and Cao, Y. Simvlm: Simple visual language model pretraining with weak supervision. *arXiv preprint arXiv:2108.10904*, 2021.

Wu, S., Fei, H., Qu, L., Ji, W., and Chua, T.-S. Next-gpt: Any-to-any multimodal llm. *arXiv preprint arXiv:2309.05519*, 2023.

Yang, Q., Chen, Q., Wang, W., Hu, B., and Zhang, M. Enhancing multi-modal and multi-hop question answering via structured knowledge and unified retrieval-generation. *arXiv preprint arXiv:2212.08632*, 2022.

Yasunaga, M., Ren, H., Bosselut, A., Liang, P., and Leskovec, J. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*, 2021.

Yu, B., Fu, C., Yu, H., Huang, F., and Li, Y. Unified language representation for question answering over text, tables, and images. *arXiv preprint arXiv:2306.16762*, 2023.

Zhang, Y., Gong, K., Zhang, K., Li, H., Qiao, Y., Ouyang, W., and Yue, X. Meta-transformer: A unified framework for multimodal learning. *arXiv preprint arXiv:2307.10802*, 2023a.

Zhang, Z., Zhang, A., Li, M., Zhao, H., Karypis, G., and Smola, A. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923*, 2023b.

Zhou, L., Palangi, H., Zhang, L., Hu, H., Corso, J. J., and Gao, J. Unified vision-language pre-training for image captioning and vqa, 2019.

Zhu, D., Chen, J., Shen, X., Li, X., and Elhoseiny, M. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

A. Appendix A

Distribution of work for the final report:

1. Deigant Yadava (25%) :
   (a) Data annotation for patch classification
   (b) Implementation of Idea 3/third proposed approach involving the joint optimization of answer generation and patch classification.

2. Joao Monteiro (25%):
   (a) Data annotation for patch classification
   (b) Implementation of Idea 1/first proposed approach involving the pre-training of the patch classification network and then using the language model to generate the answer.

3. Vinay Nair (25%):
   (a) Data annotation for patch classification
   (b) Data generation and implementing preprocessing pipeline to convert the Idea1 annotated dataset to Idea2 dataset.

4. Dheeraj Pai (25%):
   (a) Data annotation for patch classification
   (b) Implementation and training the ViLT and the adapter for Idea2 (Pyramidal approach). Implementation and training the T5 language model to adapt to the image embedding generated by ViLT.

B. Appendix B

The training of SKURG involves optimizing an overall loss function that is the sum of the five loss functions:

1. **Alignment between sources and entities**: To align the head entities from the knowledge graph with the sources (i.e. image or text) SKURG utilizes a cross-entropy loss:

$$\mathcal{L}_a = -\frac{1}{n}\sum_{i=1}^{n} \log \frac{\exp s_{i,i+}}{\sum_{j=1}^{N} \exp s_{i,j}}$$

   where n is the number of source, N is the number of head entities $s_{i,j}$ is a similarity score between the $i-$th source and the $j-$th entity.

2. **Confidence estimation of connecting the source to the knowledge graph**: the similarity score $s_{i,j}$ cannot guarantee that the $i-$th source is indeed relevant to the the $i-$th source and the $j-$th entity. Thus, SKURG computes a confidence score to represent the probability that a source should be aligned with the knowledge graph. This is done through the following loss:

$$\mathcal{L}_c = -\frac{1}{n}\sum_{i=1}^{n} \left(\hat{p}_i \log p_i + (1-\hat{p}_i)\log(1-p_i)\right)$$

   where p_i is the confidence of whether the $i-$th source is related to any head entity in the knowledge graph and $\hat{p}_i$ is a binary variable indicating whether the $i-$th source contains any head entity.

3. **Retrieval**: The retrieval in retrieval-generation decoder starts at time step $|Q|$ (i.e length of the question). SKURG utilizes two loss functions:

$$\mathcal{L}_r = -\frac{1}{M}\sum_{t=|Q|}^{|Q|+M} \log \frac{\exp \alpha_{t,t+}}{\sum_{i=1}^{n} \exp \alpha_{t,i}}$$

$$\mathcal{L}_s = -\frac{1}{M}\sum_{t=|Q|}^{|Q|+M} \left(\hat{g}_t \log g_t + (1-\hat{g}_t)\log(1-g_t)\right)$$

   where $\alpha_{t,i}$ denotes the cross-attention scores of the $i-$th source at time step t, α_{t,t^+} denotes the cross-attention score of the target source at time step t, M is the number of retrieval steps, and $\hat{g}_t$ indicates whether to output the retrieved evidence or end retrieval.

4. **Answer generation**: SKURG minimizes the negative log-likelihood for answer generation. More specifically,

$$\mathcal{L}_g = -\sum_{t=|Q|+M}^{|Q|+M+|A|} \log P_t\left(a_i | \boldsymbol{H}_R^E, a_{<t}\right)$$

   where $\boldsymbol{H}_R^E$ is the encoded representation to the retrieved sources and $a_{<t}$ are the tokens in the answer prior to time step t.

C. Appendix C

In the context of the WebQA dataset, Solar consists of two parts: re-ranking and answer generation. The re-ranking module helps in filtering the source and selecting the most relevant ones. A simple neural network model (on top of the BERT embeddings) is used to compute the closeness score between the query and the sources. This neural network is trained using the binary cross entropy loss. Finally, once the most relevant set of sources are selected, the query along with the relevant sources are fed into a T5 transformer model (Roberts et al., 2019) which generates the answer. During training, the T5 model is trained with the ground truth answers.

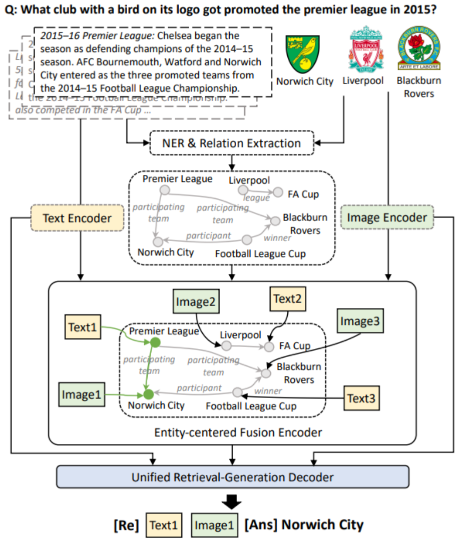

Q: What club with a bird on its logo got promoted the premier league in 2015?
2015–16 Premier League: Chelsea began the season as defending champions of the 2014–15 season. AFC Bournemouth, Watford and Norwich City entered as the three promoted teams from the 2014–15 Football League Championship.
also competed in the FA Cup ...
Norwich City
Liverpool
Blackburn Rovers
NER & Relation Extraction
Text Encoder
Image Encoder
Premier League
Liverpool
FA Cup
league
participating team
participating team
Blackburn Rovers
participant
winner
Norwich City
Football League Cup
Image2
Text2
Image3
Text1
Image1
Text3
Entity-centered Fusion Encoder
Unified Retrieval-Generation Decoder
[Re] Text1 Image1 [Ans] Norwich City

Figure 8. SKURG approach - Source: (Yang et al., 2022)

Re-Ranking: Once all the image sources have been converted into text, a simple neural network model (on top of the BERT embeddings of the text) is used to compute the closeness score between the query and each source. This score is passed through a sigmoid function to compute the probability of the source being relevant. Equation 1 provides mathematical notation to compute the score of each clue (z_i) for a question (q).

$$s_i = \text{Linear}\left(\text{BERT}\left(z_i \oplus q\right)\right) \qquad (1)$$

The re-ranking model is trained using the binary cross-entropy loss using the golden labels (y_i) as shown in Equation 2.

$$L_i = -\left(y_i \log\left(\sigma(s_i)\right) + (1 - y_i)\log\left(1 - \sigma(s_i)\right)\right) \qquad (2)$$

Answer Generation: Once the relevant set of sources have been extracted for a given question, Solar makes use of a T5 (Roberts et al., 2019) based language model to generate the answer given the question and relevant sources. The goal of the T5 based language model would be generate an answer that has the maximum probability under the given question and sources. The T5 model is trained end-to-end using the golden answers ($a*$) that are provided in the dataset. The standard masked language model loss is used for training the T5 model as show in Equation 3

$$\mathcal{L} = -\sum_{t=1}^{T} \log P(a_t^* | q, \mathbf{z}, a_{1:t-1}) \qquad (3)$$

D. Appendix D

The VLP + x101FPN baseline proposed in (Chang et al., 2022) makes use of two separate VLP (Zhou et al., 2019) models for the sub-problems of identifying relevant clues and answering questions in the WebQA dataset.

Input Representation: To form the input to the model, text inputs were tokenized using BERT (Devlin et al., 2018), while each image was characterized by 100 regions extracted from a variant of the Faster RCNN architecture with a ResNeXt-101-FPN backbone pre-trained on the Visual Genome corpus (Krishna et al., 2017). The standard 2048-dimensional features from the initial fc1 layer of Faster RCNN were utilized, and the subsequent fc2 layer was fine-tuned for the task.

Retrieval/Re-ranking: From the candidate set (S) of clues, in-order to retrieve the relevant clues VLP + x101FPN passes the representation for each candidate (s_i) along with the question through a VLP model which outputs a probability of the clue being relevant. For any question, given a set of P positive clues and N negative clues, the following loss function is used to fine-tune the VLP model (where $p(s_i)$ is the output from the VLP model):

$$\text{Loss}_{\text{retrieval}} = \sum_{s_i \in P} \log p(s_i) + \sum_{s_i \in N} \log(1 - p(s_i)) \qquad (4)$$

Answer Generation: Once the relevant clues (S_r) for a question have been identified, another VLP model is used to generate the answer given the question and the set of relevant clues. The answer is generated auto-regressively and the standard masked language modelling loss is used to train the modelling. The objective for the language modelling loss is to maximize the probability of the correct answer token (a_i^*) given the context (Q, S_r) and the set of tokens that have been generated till now $a_{1:i-1}$:

$$\text{MLM Loss} = -\sum_{i=1}^{n} \left(\log(P(a_i^* \mid Q, S_r, a_{1:i-1}))\right) \qquad (5)$$

E. Appendix E

Reasoning with Graphs: Graphs are one of the most expressive and structured ways that can be used for reasoning. From among the methods that perform question answering based on knowledge graphs (KG), (Yasunaga et al., 2021) proposes to make use of a language model to identify the important parts of the knowledge graph and then use a Graph Neural Network (GNN) to reason over the KG and predict the correct answer option. Similarly, (Liang et al., 2021) tackles the task of VQA using GNNs where each image is represented as a scene graph.

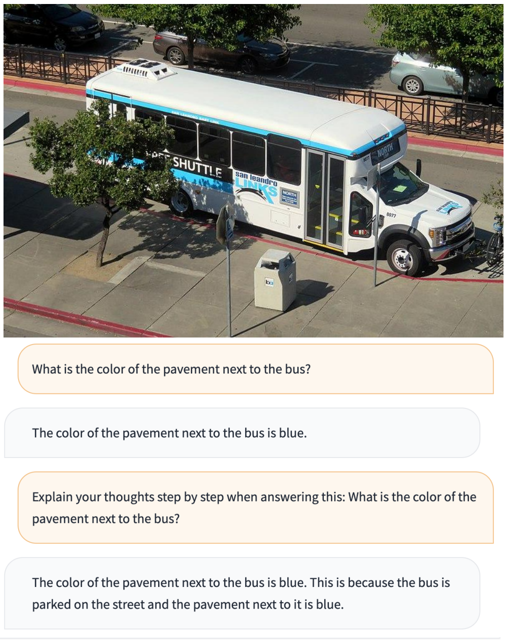
Reasoning with Planning: Recent works have used an intermediate step of planning while reasoning and generating the answers. (Zhang et al., 2023b) use multi-modal chain-of-thought reasoning for visual question answering. (Wu et al., 2023) use the ensemble of audio, video and image unimodal encoder and decoders along with an LLM to build a

“Any-to-Any” multi-modal generator. (Huang et al., 2022) explore ways to use Language Models to generate an “Inner monologue” for the robot to plan the next steps to achieve a goal. The work focuses on applying “Inner monologue” in a multi-modal setting where the query is in text and the input to the robot to be visual.

Reasoning with Vision Language Models: One of the most common way to perform reasoning for VQA is to use an existing pre-trained model and then fine-tune it according to the dataset. (Wang et al., 2021), (Zhang et al., 2023a), (Chen et al., 2020), (Kim et al., 2021), (Wang et al., 2022) introduce multi-modal transformer models that differ in their training strategies/objectives.

F. Appendix F

In Figure 9, we see an example of a question in which chain-of-thought reasoning fails because the model is not able to focus on the parts of the image that are relevant for the question. In this example, we see that for the question ”What is the color of the pavement next to the bus” , the correct answer should be ”Red” but the model answers ”blue” probably because it focuses on the color on the bus instead of the pavement. The other approaches mentioned in this paper can augment such chain-of-thought reasoning based methods by improving the alignment between images and text.



What is the color of the pavement next to the bus?

The color of the pavement next to the bus is blue.

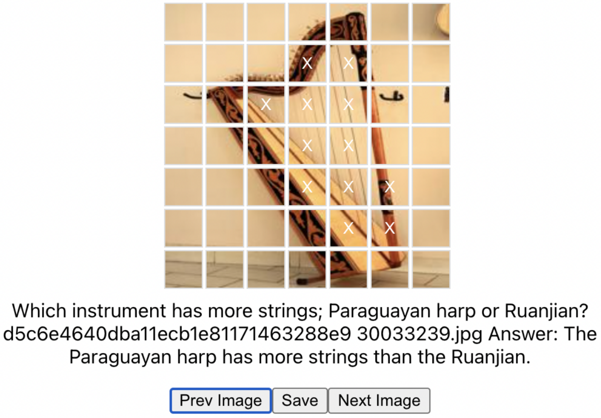
Explain your thoughts step by step when answering this: What is the color of the pavement next to the bus?

The color of the pavement next to the bus is blue. This is because the bus is parked on the street and the pavement next to it is blue.

Figure 9. Example of a failure in chain-of-thought reasoning

G. Appendix G

Figure 10 shows the interface for the data annotation application that was designed.



Which instrument has more strings; Paraguayan harp or Ruanjian? d5c6e4640dba11ecb1e81171463288e9 30033239.jpg Answer: The Paraguayan harp has more strings than the Ruanjian.

Figure 10. Annotation app example.

H. Appendix H

Figure 11 shows the ROC for the Patch Selector on the validation dataset.

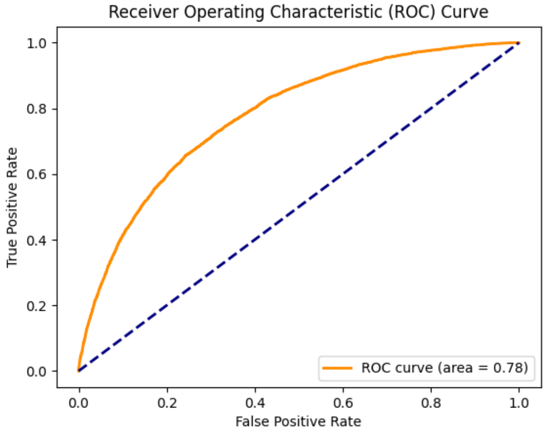


Figure 11. ROC for Patch Selector on the validation dataset.

H.1. Training Pyramidal Patch Network

Figure 12, 13, 14 shows the ROC for the Pyramidal Patch Network (Level1, Level2, Level3 respectively) on the validation dataset.

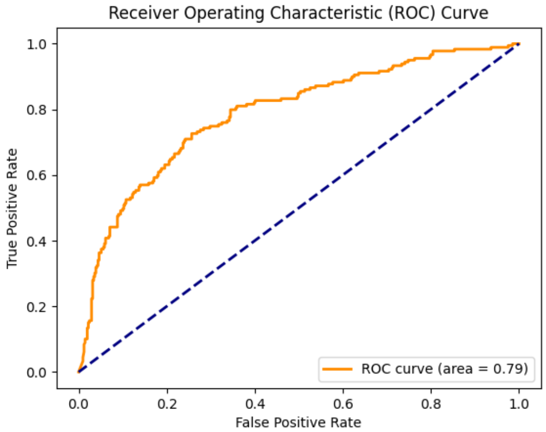


Figure 12. ROC for Pyramidal Patch Network (level1) on the validation dataset.

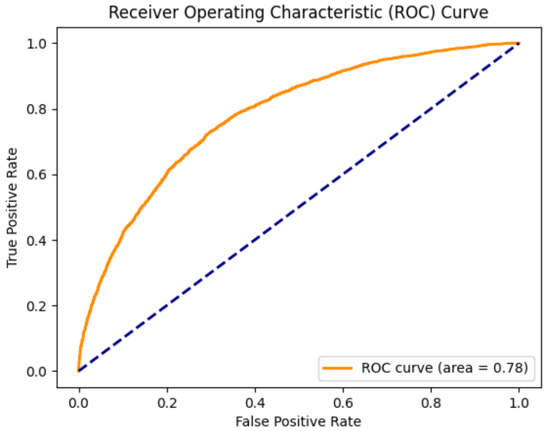


Figure 13. ROC for Pyramidal Patch Network (level 2) on the validation dataset.

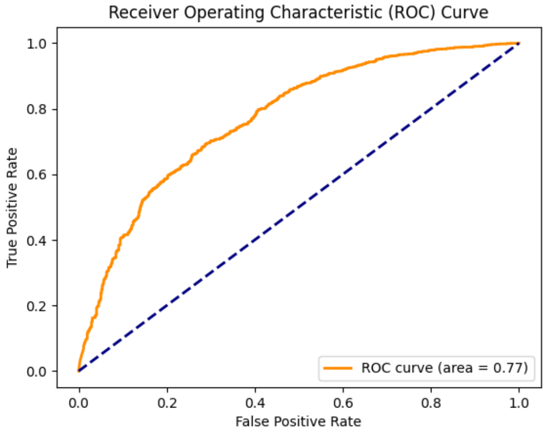


Figure 14. ROC for Pyramidal Patch Network (level 3) on the validation dataset.