Incorporating Vision Encoders into Retrieval Augmented Visual Question Answering

Xueyan Li

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
MPhil in Machine Learning and Machine Intelligence

Queens' College

August 2023

Declaration

I, Xueyan Li of Queens' College, being a candidate for the MPhil in Machine Learning and Machine Intelligence, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

This report uses the following software:

- The code and model from RA-VQA (Lin and Byrne, 2022).
- The code developed for the VQA section of the MLMI8 Neural Machine Translation and Dialogue Systems coursework.
- Specific model versions and checkpoints used are stated in the body of the report.

Code produced in this report can be found at https://github.com/xueyan-lii/RA-VQA. Access will be granted by request.

Word count: 14928

Xueyan Li
August 2023

Abstract

Knowledge-based visual question answering (KBVQA) requires external knowledge beyond images to answer questions. This requires good visual understanding, language generation ability and an external knowledge sources. Recent works commonly use three methods for KBVQA: large visual-language models, external document retrieval process and in-context learning. However, most works only use one or two of the aforementioned methods, which have limitations. Large visual-language models and in-context learning methods rely on the language model's implicit knowledge obtained during training from internet-scale data as the knowledge source, but such implicit knowledge cannot cover all topics. Documents retrieved from external knowledge sources can be used to complement implicit knowledge. Unfortunately, current document retrieval based methods have limited visual understanding due to the use of text-based vision. Image content is translated into text-based features such as captions and object descriptions which can be irrelevant.

In this report, we combine all three aforementioned methods in a two-stage framework. In the first stage, we train a retrieval-augmented visual-language model to retrieve relevant documents for each question, and generate document-wise answer candidates which are question-aware document summaries. A final answer is chosen from the candidates. In the second stage, we compose in-context examples consisting of question-aware captions and document-wise answer candidates generated in the first stage. A different frozen language model is prompted with in-context examples for the final answer. The frozen language model should be able to selected a final answer from document-wise candidates, or generate an alternative answer using its implicit knowledge. Thus, both explicit external document-based knowledge and implicit knowledge are utilised. In the first stage, we obtain 62.83% in the Outside-Knowledge Visual Question Answering (OKVQA) dataset, outperforming all other retrieval based models. In the second stage, we obtain 61.69% which outperforms all other in-context methods designed for OKVQA.

Table of contents

Chapter 1

Introduction

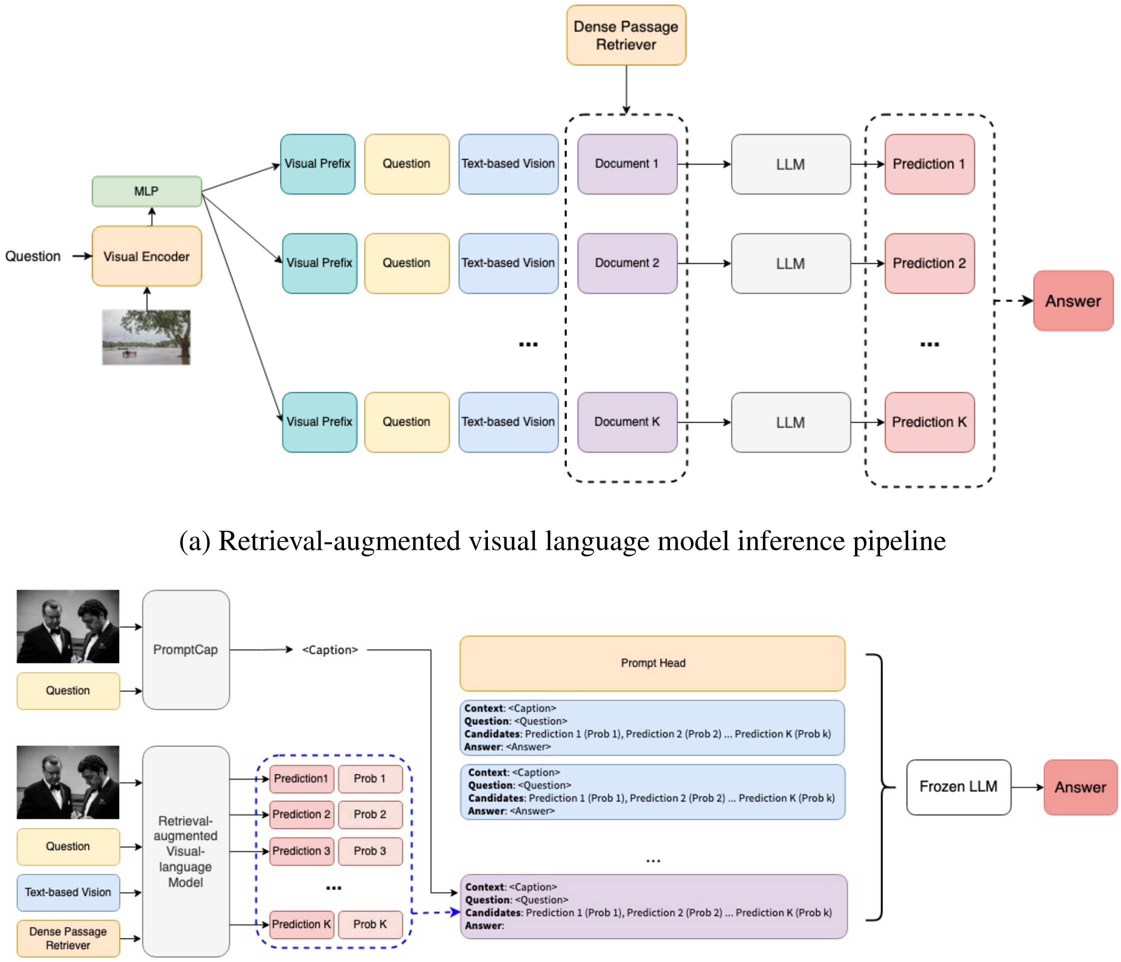
Knowledge-based visual question answering (KBVQA) is more challenging than traditional visual question answering (VQA) due to the requirement for a large knowledge source in addition to good visual understanding and language generation ability. There are two common sources of knowledge in literature: implicit knowledge and explicit outside knowledge (Pan et al., 2023). Implicit knowledge is gained by a large language model (LLM) during the training process from a large amount of text-based data in a wide range of topics. Thus, the LLM gains reasoning ability, common sense as well as specific knowledge that it has been trained on. Additionally, scaling up the model size allows LLMs to present emergent abilities that are not seen in smaller models, such as in-context prompting (Wei et al., 2022) and cross-domain knowledge transfer (Driess et al., 2023). Thus, state-of-the-art models in KBVQA tasks are large visual-language models such as PaLM-E (562B) (Driess et al., 2023) and PaLI-X (55B)(Chen et al., 2023a).

Good KBVQA performance can also be seen in frozen LLMs such as GPT-3 (Hu et al., 2023; Shao et al., 2023) which do not have a visual encoder. Visual information is represented in text-form such as captions, and in-context prompting is used to generate an answer in a desired format. However, implicit knowledge in LLMs have limitations such as hallucination (Ji et al., 2023) and lack of interpretability (Danilevsky et al., 2020). Thus, an alternative is to use an external knowledge corpus such as Google search results and Wikidata, and design a retrieval process to retrieve the most relevant passages that a LLM can make use of when generating an answer (Gao et al., 2022; Lin and Byrne, 2022; Lin et al., 2022).

We can summarize three popular methods used for KBVQA: large visual-language model, in-context learning, and external document retrieval. However, each method used on its own has limitations. In this report, we introduce a pipeline that combines all three methods such that we achieve good visual understanding by using a visual-language model. We also utilize external document-based knowledge and LLM implicit knowledge together.

External knowledge retrieval systems for VQA have two steps (Lin and Byrne, 2022; Luo et al., 2021). Firstly, a relevant passage is retrieved from a knowledge base such as Wikipedia (Gao et al., 2022) and Google Search (Lin and Byrne, 2022; Luo et al., 2021). Secondly, a language model refers to the passage retrieved and produces an answer. RA-VQA (Lin and Byrne, 2022) follows this framework and conducts joint training of Dense Passage Retrieval (DPR) and answer generation. DPR uses one encoder for questions and one encoder for documents. DPR is trained by increasing the similarity between encoded questions and encoded documents that help answer the question. Throughout training, the answer generator provides more accurate predictions while the DPR retrieves more relevant documents to the predictions. However, RA-VQA uses text-based vision, which consists of image captions, object detection and text-recognition generated by different visual models. Such text-based vision has limited relevance since they are independent of the question, meaning that the same image description is given for different questions about the same image. At the same time, object descriptions are very detailed and add unnecessary information. Ideally, the visual information used would be catered for each question. For example, if a question is asking about an animal in a picture, visual information should be focused on the animal, rather than the background. In this report, we improve upon the RA-VQA methodology by adding question-specific image encoding. We use InstructBLIP (Dai et al., 2023), which is an instruction fine-tuned visual-language model that can generate image encoding catered to the question.

Rather than using an external knowledge source, it is popular to prompt frozen LLMs to answer questions using implicit knowledge. PromptCap (Hu et al., 2023) and Prophet (Shao et al., 2023) are two examples that use in-context learning to prompt GPT-3 (Brown et al., 2020) for KBVQA. PromptCap focuses on image caption quality. A language model OFA (Wang et al., 2022b) is trained to generate question-aware image captions. This trained OFA model then generates captions for OKVQA images to be used for in-context learning. Prophet adopts a similar in-context framework except it focuses on providing a potential list of answers. It uses Oscar+ (Li et al., 2020) captions as text-based vision and trains a model MCAN-large (Yu et al., 2019) to do VQA as a classification task. It uses the highest probability inference output logits with corresponding words as a list of answer candidates. Prophet has two main weaknesses, the captions are generic and do not necessarily help answer questions. This can be improved by using PromptCap's question-aware captions. Prophet also uses answer candidates that come from a VQA trained latent space where similar words that could answer a question can be found. There is no explicit knowledge source. Thus, the answer candidates have low relevance.



(a) Retrieval-augmented visual language model inference pipeline

(b) In-context learning pipeline

Fig. 1.1 Two main methods used in this report

This report is divided into two main sections. In the first section, we use a retrieval based method with visual-language models to generate a list of potential answers that come from relevant external documents. We choose a final answer from those potential answers as seen in Figure 1.1a. This model is fine-tuned on the OKVQA dataset and outperforms other visual-language models of similar sizes. In the second section, we combine the methods in PromptCap (Hu et al., 2023) and Prophet (Shao et al., 2023) as seen in Figure 1.1b. We follow the in-context framework of Prophet but replace captions with PromptCap's question-aware captions. We provide better answer candidates by using a fine-tuned model from the first section to generate document-wise answers which are essentially question-aware document summaries. The answer candidate quality improves significantly from that in Prophet (Hit Rate@10 85.57 vs 79.83). As a result, we manage to obtain in-context performance 61.69% on OKVQA prompting frozen Flan-T5-XXL, and 61.11% prompting GPT-3.5, with only 5 in-context examples and no ensembling. This outperforms Prophet performance 61.10% that used 20 in-context examples and 5-fold ensemble.

In summary, the main contributions in this report are as follows:

- Incorporate visual encoders into RA-VQA (Lin and Byrne, 2022) by adding image embeddings directly into the text embedding space of large language models.
- Investigate the effectiveness of the 'pre-train then fine-tune' pipeline for training mapping network that bridges visual and language spaces.
- Compare captioning performance and image embedding quality of ViT (Radford et al., 2021), BLIP2 (Li et al., 2023) and InstructBLIP (Dai et al., 2023) visual encoders.
- Achieve 62.95% accuracy in OKVQA dataset with a fine-tuned 4.1B parameters model that places 4th place on the OKVQA leaderboard, after PaLM-E (562B), PaLI-X (55B) and PaLI (17B) (Chen et al., 2023a,b; Driess et al., 2023). Our model achieves accuracy close to visual-language models of much larger sizes while outperforming all other retrieval based methods.
- Propose a novel in-context learning method that utilizes external documents and the implicit knowledge of large frozen language models at the same time. In-context learning examples consist of question-aware captions and answer candidates from the previously fine-tuned visual-language RA-VQA model.
- Achieve in-context learning 61.69% accuracy for OKVQA with no ensembling, which outperforms the best models that use similar in-context learning methods (Prophet 61.10% (Shao et al., 2023), PromptCap 60.4% (Hu et al., 2023)).

Chapter 2

Background

2.1 Visual Question Answering

2.1.1 Visual Question Answering

Visual Question Answering (VQA) is a category of task that requires an answer to a question about a specific image. The challenge of VQA lies in the need for good image understanding as well as answer generation ability. There are several categories of VQA datasets. General VQA datasets include VQA 2.0 (Goyal et al., 2017) and MovieQA (Tapaswi et al., 2016). Datasets for visual reasoning that requires spatial understanding include DAQUAR (Malinowski and Fritz, 2014), GQA (Hudson and Manning, 2019)), Visual Genome (Li et al., 2019), CLEVR (Johnson et al., 2017) and Visual Spatial Reasoning (Liu et al., 2023a). There are also datasets that focus on text understanding such as TextVQA (Singh et al., 2019) and OCR-VQA (Mishra et al., 2019).

State-of-the-art models on VQA tasks use multimodal visual-language models that follow a two-step process. The first step is pre-training on vision-language datasets, then fine-tuning on VQA datasets (Chen et al., 2020; Gan et al., 2020; Li et al., 2021, 2020; Lu et al., 2019). Other methods that do not use an integrated vision-language architecture often incorporate visual understanding through other cross-modality means (Guo et al., 2023; Jiang et al., 2020; Yu et al., 2019, 2018). Amongst those methods, the transformer is a common backbone (Vaswani et al., 2017).

2.1.2 Knowledge-Based VQA

A more challenging type of VQA contains questions requiring outside knowledge that is not directly obvious from the image. Examples include OKVQA (Marino et al., 2019), FVQA

(Wang et al., 2018) and A-OKVQA (Schwenk et al., 2022). There are two broad categories of methods to obtain knowledge: using an external knowledge source or using the language model's implicit knowledge. External knowledge sources can be structured or unstructured.

Structured knowledge sources define fixed relationships between objects or ideas. For example, the FVQA dataset (Wang et al., 2018) contains a related fact for each question. Wang et al. (2017) created their own structured knowledge base that relates objects in images to concepts. Marino et al. (2020a) constructed graph networks where the nodes are concepts and edges are the relationship between them.

On the other hand, unstructured knowledge bases contain raw text-based documents that have no defined relationship between the documents or any features of interest. Sources include ConceptNet (Liu and Singh, 2004), Wikipedia and Google search (Luo et al., 2021). Luo et al. (2021) collected Google search results with question-answer pairs as the query. Gao et al. (2022) retrieved relevant material from over 21 million Wikipedia passages. KAT (Gui et al., 2022) and REVIVE (Lin et al., 2022) used a subset of Wikidata (Vrandečić and Krötzsch, 2014). MAVEx (Wu et al., 2021) and KRISP (Marino et al., 2020b) used a combination of Wikipedia and ConceptNet. While unstructured knowledge bases might contain more information than structured ones, it is challenging to find the right document and extract the most pertinent information.

It is also feasible to use a LLM's implicit knowledge without any additional knowledge retrieval step. The LLM can be prompted to obtain the answer directly (Yang et al., 2022), or used as an intermediate step to extract useful information. PromptCap (Hu et al., 2023) used GPT-3 (Brown et al., 2020) to generate question-aware captions, and then again to generate answers from in-context prompting. Prophet (Shao et al., 2023) used GPT-3 to select the final prediction from answer candidates generated by a different VQA model. REVIVE (Lin et al., 2022) and KAT (Gui et al., 2022) used GPT-3 to generate explanations to answer candidates that combines with other information as the input to a different LLM for end-to-end fine-tuning. Since many LLMs are proprietary or too large to fine-tune end-to-end, in-context learning is a popular choice to adapt the LLM to new domains or tasks without fine-tuning on new data.

There are two limitations to using implicit knowledge of LLMs. Firstly, the LLM's knowledge depend on the training data, as it is not able to know about events that occurred after the training data cut-off date. For example, the popular ChatGPT has a cut-off date of September 2021. Unstructured knowledge bases can circumvent this issue by conducting Google or Wikipedia searches for up-to-date information. Although most VQA datasets do not require up-to-date information, the issue of limited training data still means that the LLM cannot contain information that it had not been trained upon. At the same time, certain proprietary models might also have been trained on VQA datasets. Thus, information leak might occur when evaluating them on certain datasets.

This report focuses on the OKVQA dataset for which the current leaderboard is shown in Table 2.1[1]. We identify three main categories of methods used: retrieval-based systems, in-context learning and visual-language model. Retrieval-based systems have an additional document retrieval step to get the most relevant information from explicit knowledge bases. Methods that use in-context learning do not fine-tune the LLM end-to-end, but use examples to prompt the LLM to generate desired answers. Visual-language models can bridge the visual-language space directly, rather than represent visual information through text form. In the next 3 sections, each of these methods will be discussed in detail. We contextualize those models for VQA tasks and focus on state-of-the-art methods.

Table 2.1 OKVQA leaderboard

Name	Source	VQA Score	Methods
PaLM-E	Driess et al. (2023)	66.1	Visual-Language Model
PALI-X	Chen et al. (2023a)	66.1	Visual-Language Model
InstructBLIP	Dai et al. (2023)	62.1	Visual-Language Model
FLMR	Not yet published	62.1	Visual-Language Model Retrieval-Based System
Prophet	Shao et al. (2023)	61.1	In-Context Learning
PromptCap	Hu et al. (2023)	60.4	In-Context Learning
REVIVE	Lin et al. (2022)	58.0	Visual-Language Model Retrieval-Based System
Flamingo	Alayrac et al. (2022)	57.8	Visual-Language Model In-Context Learning
BLIP-2	Li et al. (2023)	54.7	Visual-Language Model
RA-VQA	Lin and Byrne (2022)	54.5	Retrieval-Based System
TRiG	Gao et al. (2022)	50.5	Retrieval-Based System
PICa	Yang et al. (2022)	48.0	In-Context Learning

[1] As of Aug 14, 2023 from https://paperswithcode.com/sota/visual-question-answering-on-ok-vqa

2.2 Visual-Language Model

It is a trend in recent years to train models of increasing capacity. For unimodal tasks, models such as BERT (Devlin et al., 2019), T5 (Raffel et al., 2020), GPT-3 (Brown et al., 2020), PALM (Chowdhery et al., 2022) and LLaMA (Chowdhery et al., 2022) were trained on large amount of text for general purpose language tasks like dialogue, translation, question answering and reading comprehension. For visual understanding, Vision Transformers (Dosovitskiy et al., 2021) inspired models to use patch-based image understanding for tasks such like semantic segmentation (Xie et al., 2021), object detection (Carion et al., 2020) and image classification (Zhai et al., 2022a), and these similarly benefit from larger model sizes. Following the success of unimodal models in getting better performances in single-domain tasks as the size of the models scales, it has become popular to use a large unified model for both image and text representations. State-of-the-art models on VQA task that were released in 2021 such as VLMo-Large (Bao et al., 2022) and SimVLM-Huge (Wang et al., 2022d) had 562 million and 632 million parameters. Those released in 2021 such as BEiT (Wang et al., 2022c), PaLI-17B (Chen et al., 2023b) and Flamingo-80B (Alayrac et al., 2022) had 1.9 billion, 17 billion, and 80 billion parameters. PaLI-X (Chen et al., 2023a) and PaLM-E (Driess et al., 2023) were published in 2023 and have 55 billion and 562 billion parameters.

However, there are downsides to relying on larger and larger models to achieve better performance for VQA. For example, PaLM-E's 562B version (Driess et al., 2023) currently achieves state-of-the-art performance on the OKVQA dataset (Marino et al., 2019) with 66.1% accuracy. However, its 12B version has 60.1% accuracy, which is worse than many models of similar or smaller size (Dai et al., 2023; Hu et al., 2023; Shao et al., 2023). Such large models are often inaccessible to the public, and incur huge training costs. Their main contributions are as general-purpose models that can be used in a variety of tasks, rather than as specific solutions for one task.

In this report, we use medium sized visual-language models such as BLIP2 (Li et al., 2023) and InstructBLIP (Dai et al., 2023). We design a specific solution for outside-knowledge retrieval using visual-language models. This chapter next introduce various types of architectures used for single modality understanding, then discuss cross-modality bridging mechanisms and how visual representation is incorporated into language models.

2.2.1 Architectures for single-modality components

Question answering is a sequence-2-sequence (seq2seq) task that requires a language model to receive an input sequence of text, and output a sequence of text. Seq2seq tasks can be solved by encoder-decoder or decoder-only models. Encoder-decoder models first encode the input sequence into a latent space, before a decoder generate the output. A decoder-only model combines the input and target sequences for training. The advantage of decoder-only models is that the lack of an encoder means bigger models can be trained (Aghajanyan et al., 2023). Additionally, decoder-only models can be trained on unlabelled text, which is much more abundant and easy to obtain. Thus, we see a trend in recent years where the largest language models are decoder-only, such as GPT-3 (175B) (Brown et al., 2020), PaLM (540B) (Chowdhery et al., 2022), Megatron-TuringNLG (530B) (Smith et al., 2022) and LaMDa (137B) (Thoppilan et al., 2022). In contrast, encoder-decoder models' bidirectional-attention mechanism allows them to process context in both directions. Thus, for very long inputs, encoder-decoder models are less likely to miss information placed in the middle of the input (Liu et al., 2023b).

Vision Transformer (ViT) with image-text pre-training is a popular choice for visual understanding in VQA tasks. CoCa (Yu et al., 2022) trained an image-text encoder-decoder model jointly with contrastive and captioning loss. The visual encoder follows the setup of ViT-g (Zhai et al., 2022b). BEiT-3 (Wang et al., 2022c) also uses ViT-g but regards the image as a foreign language. Image patches and text tokens are masked for recovery in pre-training to align modalities. GIT (Wang et al., 2022a) used ViT-B/16 and ViT-L/14 from CLIP (Radford et al., 2021) with a text decoder to train a model with captioning loss. In this report, we use pre-trained ViT-L/14 and ViT-g as visual encoders.

2.2.2 Cross modality bridging mechanism

From the ViT pre-training strategies described in the previous paragraph, we observe that contrastive loss and captioning loss are commonly used. Constrastive loss is used to jointly train image and text encoders such that the distance between encoding of similar image and text is minimized (Jia et al., 2021; Radford et al., 2021). Image captioning is used in models such as Frozen (Tsimpoukelli et al., 2021), CoCa (Yu et al., 2022) and GIT (Wang et al., 2022a) as a pre-training task. Image-caption pairs come from large-scale datasets like Conceptual Captions (Sharma et al., 2018) and CC12M (Changpinyo et al., 2021). Frozen maps the visual encoder's output to the embedding space of the language model by a trainable linear layer. This type of visual linear mapping is a common cross modality bridging mechanism used in models such as MiniGPT4 (Zhu et al., 2023), MAGMA (Eichenberg et al., 2022) and VL-T5 (Cho et al., 2021). We also adopt linear mapping layer with pre-training on Conceptual Captions to create visual-language models using frozen image encoders and LLMs.

2.3 In-Context Learning

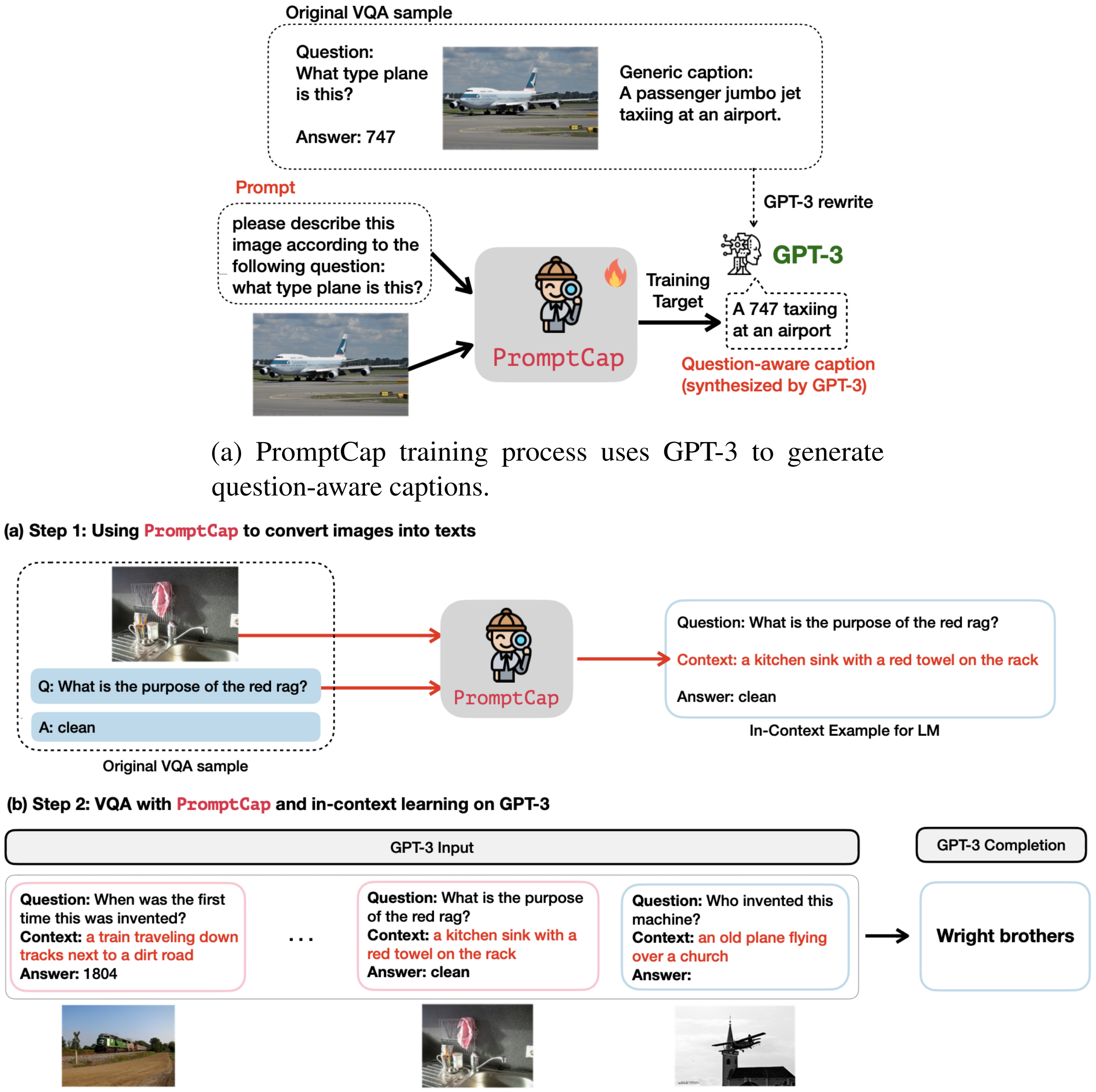
In-context learning is used to adapt LLMs to unseen tasks by giving them a few examples of input-label pairs before the evaluation example. Thus, no training is necessary. There are two general types of in-context learning, unimodel and multimodal in-context learning. In this section, we describe the usage of each type for KBVQA.

2.3.1 Unimodal In-Context Learning

LLMs's in-context ability is enabled by semantic priors and input-label mappings (Wei et al., 2023). Semantic priors means using prior knowledge from pre-training to predict labels. Input-label mappings means to learn the pattern that is presented in in-context examples, and to generate a label/prediction following that pattern.

Two best performing in-context learning models on the OKVQA dataset are Prophet (Shao et al., 2023) and PromptCap (Hu et al., 2023). Both models make use of knowledge that GPT-3 (Brown et al., 2020) gained from pre-training as semanic prior, and as the main knowledge source. For Prophet, the input-label mappings consist of context, question, candidates and answer as seen in Figure 2.2. The context is caption generated by Oscar+ (Li et al., 2020). The candidates are generated by a pre-trained VQA model based on MCAN-large (Yu et al., 2019). MCAN-large architecture is modified and pre-trained on VQAv2 (Goyal et al., 2017) and Visual Genome (Li et al., 2019). Finally it is fine tuned on OKVQA (Marino et al., 2019). The answer prediction task is framed as a classification task over a pre-defined set of vocabulary. After training, MCAN-large receives the testing question, and outputs probability logits that represent the most likely words that can answer the question. The most likely answers and corresponding probabilities make up the candidates in in-context examples. For Prophet, the in-context examples are very important since they lie in a latent space that contains information of the relationship between image, question and answer candidates. In-context examples are selected by the most similar CLIP (Radford et al., 2021) encoding in the training set to the testing CLIP encoding. With more in-context examples, GPT-3 can understand better the kind of answers expected for the particular question. Thus, it can choose an answer from the testing candidates, or choose one of the candidates that appeared in in-context examples, or generate an alternative answer.

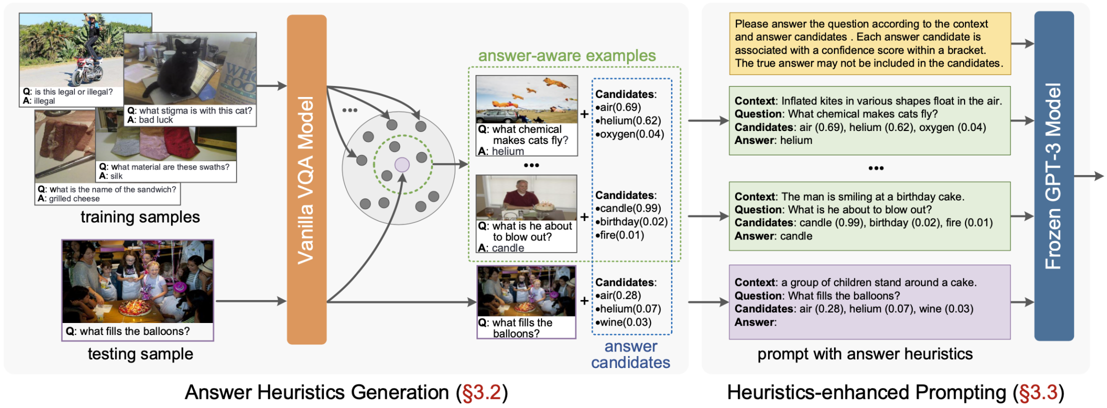
There are three limitations to this method. Firstly, the VQA pre-training process is a classification task, meaning phrases in the OKVQA testing set that do not appear in the training vocabulary will not be represented at all. In this report, we improve this by using a retrieval process that generate answer candidates that come from external documents. Thus, answer candidates come from an explicit knowledge source and are not limited by vocabulary


Original VQA sample
Question:
What type plane is this?
Answer: 747
Generic caption:
A passenger jumbo jet taxiing at an airport.
GPT-3 rewrite
Prompt
please describe this image according to the following question: what type plane is this?
GPT-3
PromptCap
Training Target
A 747 taxiing at an airport
Question-aware caption (synthesized by GPT-3)
(a) Step 1: Using PromptCap to convert images into texts
Q: What is the purpose of the red rag?
A: clean
Original VQA sample
PromptCap
Question: What is the purpose of the red rag?
Context: a kitchen sink with a red towel on the rack
Answer: clean
In-Context Example for LM
(b) Step 2: VQA with PromptCap and in-context learning on GPT-3
GPT-3 Input
GPT-3 Completion
Question: When was the first time this was invented?
Context: a train traveling down tracks next to a dirt road
Answer: 1804
...
Question: What is the purpose of the red rag?
Context: a kitchen sink with a red towel on the rack
Answer: clean
Question: Who invented this machine?
Context: an old plane flying over a church
Answer:
Wright brothers

(a) PromptCap training process uses GPT-3 to generate question-aware captions.

(b) In-context learning pipeline.

Fig. 2.1 PromptCap methodology includes training a question-aware captioning model, and using captions for few-shot prompting. Figure taken from Hu et al. (2023).



Q: is this legal or illegal?
A: illegal
Q: what stigma is with this cat?
A: bad luck
Q: what material are these swaths?
A: silk
Q: what is the name of the sandwich?
A: grilled cheese
training samples
Q: what fills the balloons?
testing sample
Vanilla VQA Model
answer-aware examples
Q: what chemical makes cats fly?
A: helium
Candidates:
•air(0.69)
•helium(0.62)
•oxygen(0.04)
Q: what is he about to blow out?
A: candle
Candidates:
•candle(0.99)
•birthday(0.02)
•fire(0.01)
Q: what fills the balloons?
Candidates:
•air(0.28)
•helium(0.07)
•wine(0.03)
answer candidates
Answer Heuristics Generation (§3.2)
Frozen GPT-3 Model
helium
prompt with answer heuristics
Heuristics-enhanced Prompting (§3.3)

Please answer the question according to the context and answer candidates . Each answer candidate is associated with a confidence score within a bracket. The true answer may not be included in the candidates.

Context: Inflated kites in various shapes float in the air.
Question: What chemical makes cats fly?
Candidates: air (0.69), helium (0.62), oxygen (0.04)
Answer: helium

•••

Context: The man is smiling at a birthday cake.
Question: What is he about to blow out?
Candidates: candle (0.99), birthday (0.02), fire (0.01)
Answer: candle

Context: a group of children stand around a cake.
Question: What fills the balloons?
Candidates: air (0.28), helium (0.07), wine (0.03)
Answer:

Fig. 2.2 Prophet in-context prompting uses answer candidates generated by a VQA model. Figure taken from Shao et al. (2023).

of the training set. The second issue is that Prophet heavily rely on ensembling. GPT-3 is prompted 5 times for the same question to majority vote for the final answer. Without ensembling, performance degrades drastically by 3.2 points. In this report, ensembling is not used. We still surpass the performance of Prophet. Lastly, Prophet has limited visual representation. Oscar+ captions are generic and do not always encapsulate relevant visual features. This can be improved by using captions generated by PromptCap (Hu et al., 2023).

PromptCap also uses in-context prompting of GPT-3 for OKVQA. The challenge in using GPT-3 for VQA is converting visual information into text descriptions that are sufficiently detailed and relevant. PromptCap first trains a captioning model to generate question-relevant image captions. Ground truth captions are generated by giving GPT-3 question-answer pairs and the prompt "Summarize the context to help answer the question." GPT-3 generates a caption that includes the answer for each question in VQAv2 (Goyal et al., 2017). Those captions are used to train a language model OFC (Wang et al., 2022b). This trained OFC model is called PromptCap. In the second step, GPT-3 receives in-context examples of question, question-aware caption generated by PromptCap and answer from the OKVQA dataset. GPT-3 learns the input-label mapping and generates a final prediction as shown in Figure 2.1. This method heavily relies on GPT-3's implicit knowledge, since no answer candidates are given. In this report, we combine PromptCap and Prophet by using the prompting framework of Prophet, but replace generic captions by question-aware captions generated by the pre-trained PromptCap. We also replace Prophet's answer candidates with more accurate document-based answer candidates.

2.3.2 Multimodal In-context Learning

Frozen (Tsimpoukelli et al., 2021) is one of the first papers that implemented multimodal few-shot learning. It trained a visual encoder to generate image prefix that can be incorporated into a frozen language model. Pairs of image prefix and text are interleaved in few-shot examples. Flamingo (Alayrac et al., 2022) extended this idea by adding Perceiver Resampler to pre-trained vision-only models and adding GATED XATTN-DENSE layers to frozen pre-trained language-only encoder-decoder models. Flamingo achieves good OKVQA performance 57.8% with 32 in-context examples and no prior fine-tuning on the OKVQA dataset. Compared to Flamingo, GIT (Wang et al., 2022a) uses a much simpler architecture of one image encoder and one text decoder. GIT is also much smaller with 0.7B parameters in total whereas Flamingo has 80B parameters. GIT achieves lower performance in VQAv2 (Goyal et al., 2017) compared to Flamingo. However, it shows better performance with 32-shot evaluation on the Flickr30K dataset (Young et al., 2014) than Flamingo. GIT achieves similar performance to other models of much larger sizes by scaling up the amount of pre-training data. In contrast, PaLI investigates the effect of scaling up visual encoder and language model sizes. In PaLI (Chen et al., 2023b), a 4B visual encoder ViT and a 17B language model are trained. In PaLI-X (Chen et al., 2023a), a 22B visual encoder and 32B language model are trained. Despite a larger visual component, PaLI-X under-performs in VQAv2 compared to Flamingo. The authors of PaLI-X hypothesized that this is due to a fine-tuned language model in PaLI-X, compared to Flamingo's frozen language model. Fine-tuning on task specific datasets seems to interfere with in-context ability.

Taking into account the conflict between fine-tuning and in-context performance, we opt to not use multimodal few-shot prompting in this report. Flamingo's best performing version with 70B parameters incur too much inference cost as the number of shots increases. PaLI-X is not open source. GIT only presented good few-shot prompting result in one dataset. It observed that better performance is achieved with few-shot training. However, that is not the focus of this report. Thus, GIT is also not tested. We instead focus on text-only few-shot prompting.

2.4 Retrieval-Based System

The only papers that combine both explicit knowledge source and implicit knowledge source are KAT (Gui et al., 2022) and REVIVE (Lin et al., 2022). Both used Wikidata (Vrandečić and Krötzsch, 2014) as an explicit knowledge source and GPT-3 (Brown et al., 2020) as an implicit knowledge source. They both constructed knowledge bases from Wikidata that contained topics relevant to the OKVQA dataset. Relevant knowledge entries with the highest similarity to image regions are retrieved. GPT-3 is prompted for implicit knowledge by an instruction, several in-context examples of context-question-answer triplets, object based description of the image and testing question. GPT-3 then generates an answer and reasoning for this answer. The explicit knowledge retrieved and GPT-3's implicit knowledge based answer and reasoning are concatenated and sent through a transformer based reasoning module for the final answer. However, image regions are based on sliding windows. They are not all useful or relevant.

REVIVE improves upon this by using object-centric regions of interest and by retrieving a document for each image region. GPT-3 is prompted for implicit knowledge by image caption. The explicit knowledge from Wikidata, image regions encoding and implicit knowledge from GPT-3 are concatenated and sent through the language model T5 (Raffel et al., 2020) for the final answer. Both KAT and REVIVE are limited by the language understanding and generation ability of the LLM used in the final answer generation step. In this paper, we use a much stronger and newer language model Flan-T5 for retrieval based fine-tuning.

TRiG (Gao et al., 2022) is another paper that uses knowledge retrieval. It uses DPR (Karpukhin et al., 2020) to encode queries and Wikipedia passages to find the most similar passages to each query. TRiG, KAT and REVIVE share the limitation where the documents retrieved are generic and do not always contain relevant information. RA-VQA (Lin and Byrne, 2022) improves retrieval performance by jointly training document retriever and answer generator. It obtains PRRecall 96.55 with 50 documents compared to TRiG's 85.56 with 100 documents. PRRecall measures the relevance of documents to a certain question. In this report, we follow the retrieval methodology of RA-VQA and adds a visual prefix for better visual understanding.

2.5 Conclusion

In this chapter, we reviewed the literature related to large visual-language models, in-context learning and document retrieval for VQA. We found that each method used on its own has limitations. State-of-the-art large visual-language models are often too big and computationally expensive to use. In-context learning methods for KBVQA only rely on LLM's implicit knowledge which is limited. Document retrieval methods are single-modality and lacks good visual understanding. In the next chapter, we propose using retrieval-augmented visual-language models and in-context learning together that make use of all three aforementioned methods.

Chapter 3

Methodology

This chapter introduces two main methods – fine-tuning retrieval-augmented visual-language model and in-context prompting of frozen LLMs. Firstly, we establish a baseline of how an encoder-decoder language model is trained. Then we describe the process of visual-language alignment that requires two steps:

1. Pre-training mapping network with frozen LLMs using a captioning task.

2. Fine-tuning mapping network and LLM together on a VQA dataset.

We will also describe how Dense Passage Retrieval (DPR) is used to retrieve external knowledge. Lastly, we use answers generated by DPR as candidates in in-context learning. We describe each prompt component and in-context example selection criteria in detail.

3.1 Fine-tuning LLMs with Text-Based Inputs

3.1.1 Fine-tuning LLM with Question Only Input

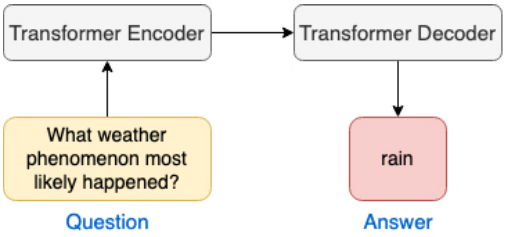


Fig. 3.1 Q-Only training process uses only question-answer pairs.

Firstly, only question and answer pairs are given to a LLM for training, with no visual information as shown in Figure 3.1. Following the naming in OKVQA (Marino et al., 2019), this fine-tuning method is called Q-Only. Training data is represented as $D = \{q, s\}$ where q is an encoded question, s is the encoding of the most frequent answer out of 10 annotated answers S for each question, also called the 'gold answer.' Each LLM takes q and is trained by teacher forcing to generate a sequence of L tokens $s = (s^1, s^2, \ldots, s^L)$. The training objective at each decoding step is to minimize the negative log-likelihood loss

$$\mathscr{L}_{\text{Q-Only}} = -\sum_{t=1}^{L} \log p\left(s^t | q, s^{<t}\right) \tag{3.1}$$

For models that are too large to fine-tune end-to-end, LoRA (Hu et al., 2022) can be used. LoRA freezes the original model parameters $W_0 \in \mathbb{R}^{d \times k}$ and injects additional rank-decomposition matrices ΔW which are trainable. This greatly reduces training time and the number of trainable parameters. During back propagation, gradient updates become $W_0 + \Delta W = W_0 + BA$ where $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times k}$. The rank r controls the number of trainable parameters. We vary r to investigate its effect on training time and performance.

3.1.2 Incorporating Text-based Vision

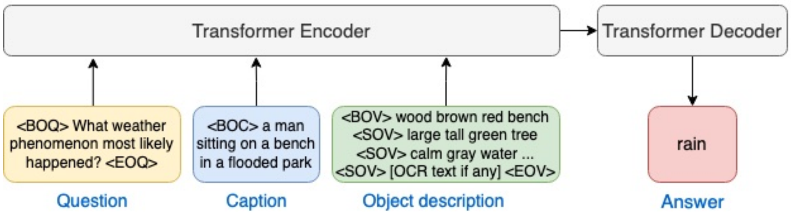

Transformer Encoder
Transformer Decoder
<BOQ> What weather phenomenon most likely happened? <EOQ>
<BOC> a man sitting on a bench in a flooded park
<BOV> wood brown red bench <SOV> large tall green tree <SOV> calm gray water ... <SOV> [OCR text if any] <EOV>
rain
Question
Caption
Object description
Answer

Fig. 3.2 Visual information can be converted to text-based vision. LLM is fine-tuned with question and text-based vision as input.

Once Q-Only baselines are established, visual features are added which are essential for image understanding. A method to include visual feature is text-based vision. Following the method used in RA-VQA (Lin and Byrne, 2022), text-based vision consists of image captions, object detection and character recognition. Each description is then concatenated with a question to be tokenized together as input to the LLM as shown in Figure 3.2. To distinguish each type of description, special tokens in brackets <> are added to the start and end of each string, and the dimension of the LLM tokenizer and embedding space are expanded to accommodate additional tokens. For each question $q =$ `<BOQ>`q`<EOQ>`,

- Oscar+ (Li et al., 2020) is used to generate a caption $C =$ `<BOC>c<EOC>`.

- VinVL (Zhang et al., 2021) is used to extract objects and their attributes $O = \texttt{<BOV>}o_1\texttt{<SOV>}\ldots\texttt{<SOV>}o_k\texttt{<EOV>}$.
- Google OCR[1] (Optical Character Recognition) is used to extract any text present in the image $T = \texttt{<SOV>}t\texttt{<EOV>}$.

Thus, the total input text to the LLM becomes $x = [q, C, O, T]$.

When LoRA is used, it is not convenient to reshape the LLM embedding dimension. Thus, words rather than special tokens are used. Separation tags are replaced by commas. The full input text becomes $x = [\text{Question: } q, \text{Caption: } c, \text{Objects: } o_1, ..., o_k, t]$. When text-based vision (TBV) is used, the training objective is to minimize

$$\mathscr{L}_{TBV} = -\sum_{t=1}^{L} \log p\left(s^t | x, s^{<t}\right) \tag{3.2}$$

3.1.3 Incorporating Dense Passage Retriever

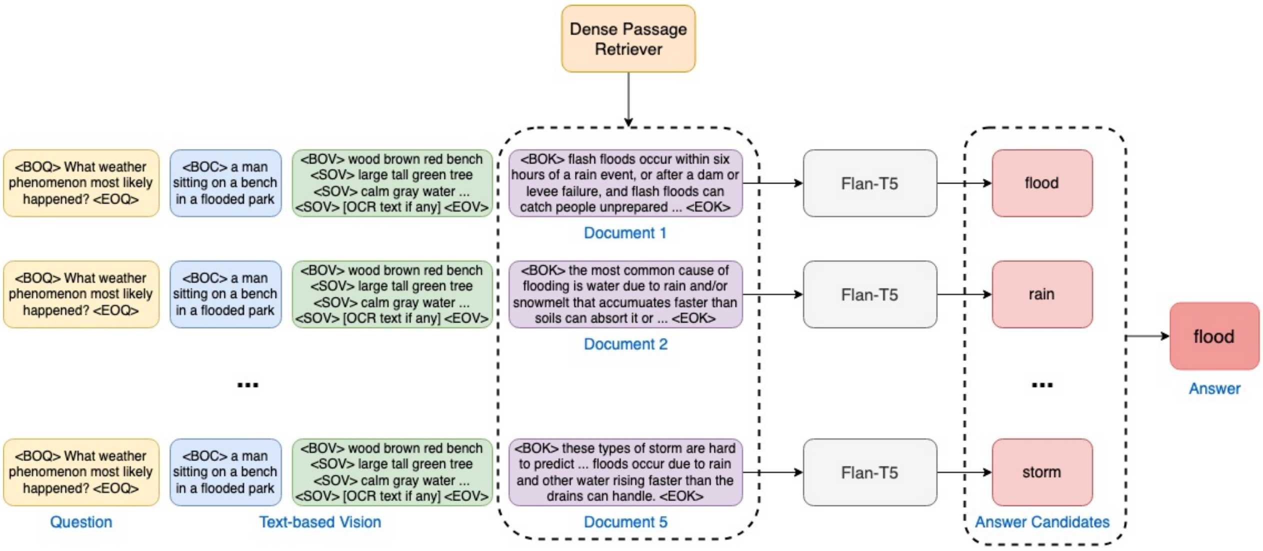

Dense Passage Retriever
<BOQ> What weather phenomenon most likely happened? <EOQ>
<BOC> a man sitting on a bench in a flooded park
<BOV> wood brown red bench <SOV> large tall green tree <SOV> calm gray water ... <SOV> [OCR text if any] <EOV>
<BOK> flash floods occur within six hours of a rain event, or after a dam or levee failure, and flash floods can catch people unprepared ... <EOK>
Document 1
Flan-T5
flood
<BOQ> What weather phenomenon most likely happened? <EOQ>
<BOC> a man sitting on a bench in a flooded park
<BOV> wood brown red bench <SOV> large tall green tree <SOV> calm gray water ... <SOV> [OCR text if any] <EOV>
<BOK> the most common cause of flooding is water due to rain and/or snowmelt that accumuates faster than soils can absort it or ... <EOK>
Document 2
Flan-T5
rain
...
...
flood
Answer
<BOQ> What weather phenomenon most likely happened? <EOQ>
<BOC> a man sitting on a bench in a flooded park
<BOV> wood brown red bench <SOV> large tall green tree <SOV> calm gray water ... <SOV> [OCR text if any] <EOV>
<BOK> these types of storm are hard to predict ... floods occur due to rain and other water rising faster than the drains can handle. <EOK>
Document 5
Flan-T5
storm
Question
Text-based Vision
Answer Candidates

Fig. 3.3 DPR is trained to retrieve relevant passages that combine with question and text-based vision for a LLM to generate document-wise answer candidates. A final answer is selected from answer candidates.

External sources of knowledge can be given to a LLM to enhance its ability to answer questions that are not immediately obvious from the image. In order to find the most relevant passages from a knowledge set, we follow the methodology (Figure 3.3) proposed by RA-VQA (Lin and Byrne, 2022):

[1] https://cloud.google.com/vision/docs/ocr

1. Pre-train a document retriever to find the most relevant k documents to each question.
2. Incorporate each document to the forward pass in an answer generator by appending the document after the question.
3. Generate an answer for each document.
4. Calculate losses to update the weights of document retriever and answer generator.

DPR Pre-training

Note that DPR pre-training is not done for this report; a pre-trained checkpoint is used instead. However, since the quality of documents retrieved directly impacts the performance of few-shot learning, the methodology for DPR pre-training is reported in detail here. The checkpoint used in this paper comes from RA-VQA (Lin and Byrne, 2022) and uses the method proposed in Karpukhin et al. (2020). Given a knowledge corpus $C = \{z_1, z_2, ..., z_M\}$ consisting of M passages of plain text and a question, a retriever R is trained to get the top k most relevant passages. Separate encoders are used for question x and documents. The question encoder F_q is trainable while the document encoder F_d is frozen. Both encoders are BERT-Base (Devlin et al., 2019) so the encoding dimension is 768. The question is a concatenation of question and text-based vision as described in Section 3.1.2. FAISS (Johnson et al., 2021) is used to index documents encodings $F_d(z_m)$ so that during training, the most similar documents to the question encoding $F_q(x)$ are found. Similarity is defined by the inner product between the question and document encoding

$$r(x,z) = F_q^{\top}(x)F_d(z) \tag{3.3}$$

After the top k most similar documents are found by FAISS, in-batch negative sampling is used to set one positive document z^+, and all other documents as negative. A positive document is defined by $H(z,S) = 1$ where the document includes at least one of the ground truth annotations by string match. The training loss is given by

$$\mathscr{L}_{\text{DPR}} = -\sum_{(x,S)\in\tau} \log \frac{\exp(r(x,z^+))}{\exp(r(x,z^+)) + \sum_{z_k \neq z^+} \exp(r(x,z_k))} \tag{3.4}$$

Thus, the question encoder is trained to maximise the similarity between the question and the positive document. After pre-training, the question encoder can be fine-tuned together with the answer generator for a VQA task, or it can be frozen.

Answer Generation

In answer generation for a VQA task, K most similar documents to the question x are retrieved through FAISS. The question has the same form as in DPR pre-training. Each document z_k is given a score

$$p_\theta(z_k|x) = \frac{\exp(r(x, z_k))}{\sum_{j=1}^{K} \exp(r(x, z_j))} \tag{3.5}$$

The answer generation process has the same form as Q-only (Section 3.1.1) with the addition of documents. Each document in text-form with tags z_k = <BOK>z_k<EOK> follows x to be tokenized together by the answer generator F_θ's tokenizer. As described in Q-only, when LoRA is used, document tags are replaced with text-based indicator "Document: z_k".

An answer $y_k = \text{argmax}_y\, p_\phi\,(y|x, z_k)$ is generated for each document. These per-document answers are used for in-context learning. For RA-VQA fine-tuning, an answer is selected to compare with the ground truth label for back propagation. The final answer selection maximizes the joint probability of generation output and document score.

$$y, z_k = \underset{y, z_k}{\text{argmax}}\, p_\phi\,(y|x, z_k)\, p_\theta\,(z_k|x) \tag{3.6}$$

Training Loss Calculation

Two training losses are calculated to update the document retriever and answer generator separately. The loss for the document retriever has two terms to reward relevant documents and discourage irrelevant documents. A relevant document is one that elicits a correct answer from F_θ, and contains a label. An irrelevant document elicits a wrong answer and does not contain any correct answer,

$$\begin{aligned} P^+(x, S) &= \{k : y_k = s_k^* \wedge H(z_k, S) = 1\} \\ P^-(x, S) &= \{k : y_k \neq s^* \wedge H(z_k, S) = 0\} \end{aligned} \tag{3.7}$$

where P^+ stands for the set of relevant documents and P^- stands for the set of irrelevant documents. $H(z_k, S) = 1$ is the pseudo relevance function where the document z_k contains a string that exactly matches an answer s_k^* in S so $s_k^* \subset S$. At the same time, the prediction needs to match the same answer $y_k = s_k^*$. In the case that z_k contains multiple answers, the most popular answer is used. Similarly, the set of irrelevant documents P^- has documents that do not contain any answer $H(z_k, S) = 0$ and do not elicit prediction of the most popular answer $y_k \neq s^*$. The overall loss term is

$$\mathscr{L}_{\text{RAVQA}} = -\sum_{(x,S)\in\tau}\left(\sum_{k=1}^{K}\log p_{\phi}\left(s_k^*|x,z_k\right)+\sum_{k\in P^{+}(x,S)}\log p_{\theta}\left(z_k|x\right)-\sum_{k\in P^{-}(x,S)}\log p_{\theta}\left(z_k|x\right)\right)$$

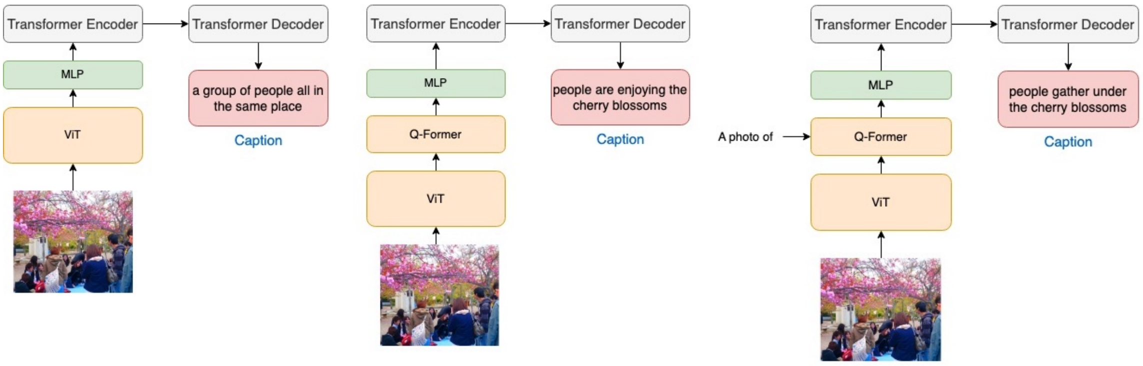
The first loss term measures answer generation correctness while the following two terms measure document retrieval quality. The DPR is encouraged to retrieve documents that contain ground truth answers recognized by the answer generator. The DPR is discouraged from retrieving documents that do not contain ground truth answer, as long as the answer generator cannot generate the gold answer. This joint training scheme that trains the DPR and answer generator at the same time boosts correct predictions and relevant document retrieval.

3.2 Incorporating Visual Prefix

The previous chapter introduces using text-based vision for visual understanding. However, this method has limitations such as long text descriptions that add unnecessary details. Thus, a more direct approach is to use the image embedding from a visual model directly. In order to bridge the gap between visual and language representation, one can use a trainable mapping framework to convert image embedding to the embedding space of the language model. These mapped image embeddings are called the image prefix and are analogous to a sequence of encoded text in the LLM space. This chapter introduces the procedure to pre-train the mapping framework on image captioning task, then fine-tune it for VQA. Image encoders from three models are tested, CLIP (Radford et al., 2021), BLIP2 (Li et al., 2023) and InstructBLIP (Dai et al., 2023).

3.2.1 Pre-training Mapping Network on Image Captioning

The mapping network pre-training process is shown in Figure 3.4. Each image is first encoded by a frozen image encoder and then passed through one fully-connected linear layer F_M to map the image encoding to the embedding dimension of LLM. When CLIP is used, its vision model ViT F_V transforms an image I into image encoding $F_V(I) \in \mathbb{R}^{d_v}$ where d_v is the CLIP encoding dimension. The mapping layer F_M transforms the image encoding to $V = F_M(F_V(I)) \in \mathbb{R}^{Ld_L}$ where L is the number of tokens in the LLM space and d_L is the embedding size of the LLM. Lastly, the output of the mapping layer is reshaped to $\mathbb{R}^{L\times d_L}$. The LLM is expected to generate the caption autoregressively. The mapping layer is trained



(a) ViT (b) BLIP2 (c) InstructBLIP

Fig. 3.4 Pre-training mapping network on captioning task for ViT, BLIP2 and InsturctBLIP require different input configurations.

to maximize the conditional likelihood of generating the ground truth, given by

$$\mathscr{L}_{CLIP} = -\sum_{t=1}^{L} \log p\left(s^t | V, s^{<t}\right) \tag{3.8}$$

where s is the encoded caption.

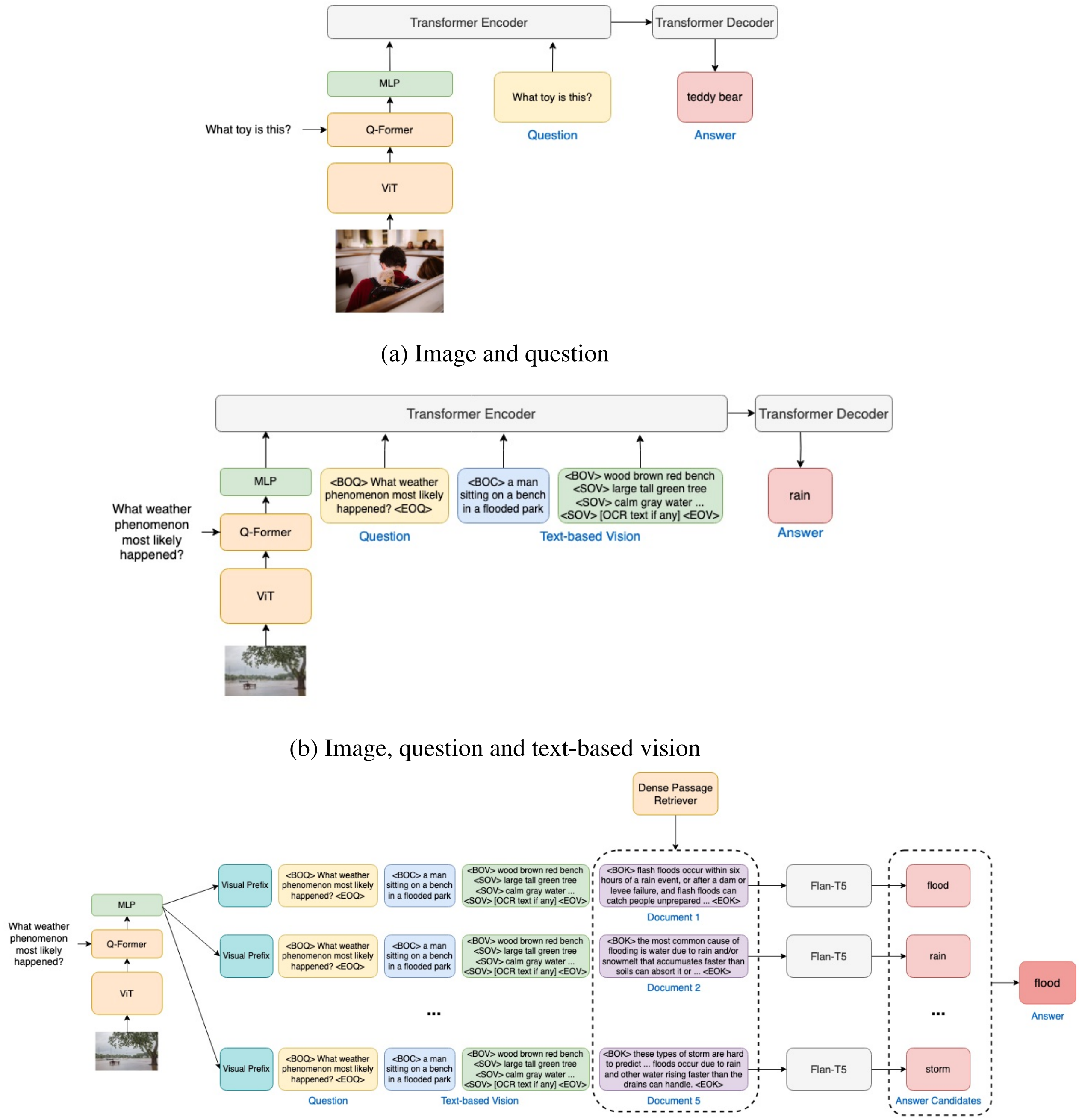
When BLIP2 is used, the visual encoder starts with ViT. The output goes into Q-Former F_Q which further transforms the image encoding to $F_Q(F_V(I)) \in \mathbb{R}^{L \times d_v}$. Q-Former was trained on image-text contrastive learning, image-grounded text generation, and image-text matching tasks (Li et al., 2023) so that it can translate the most relevant image features into text form. Mapping layer transforms Q-Former encoding to $V = F_M(F_Q(F_V(I))) \in \mathbb{R}^{L \times d_L}$, so no further reshaping is required. The training loss takes the same form as $\mathscr{L}_{CLIP}$.

When InstructBLIP is used, its Q-former takes in a text-based instruction along with an image encoding. The image prefix is represented by $V = F_M(F_Q(F_V(I), F_L(p)))$ where F_L is the encoder of LLM and p is the instruction "A photo of". The training loss becomes

$$\mathscr{L}_{BLIP} = -\sum_{t=1}^{L} \log p\left(s^t | [V, F_L(p)], s^{<t}\right) \tag{3.9}$$

3.2.2 Fine-tuning Visual-Language Model on VQA

Image encodings from the final layer of the frozen visual models pass through a pre-trained mapping network to form an image-prefix. The image prefix is then concatenated with the encoded questions to form the input into an LLM. Each model is fine-tuned on VQA tasks


Transformer Encoder
Transformer Decoder
MLP
What toy is this?
Q-Former
ViT
What toy is this?
Question
teddy bear
Answer
Transformer Encoder
Transformer Decoder
MLP
What weather phenomenon most likely happened?
Q-Former
ViT
<BOQ> What weather phenomenon most likely happened? <EOQ>
Question
<BOC> a man sitting on a bench in a flooded park
<BOV> wood brown red bench <SOV> large tall green tree <SOV> calm gray water ... <SOV> [OCR text if any] <EOV>
Text-based Vision
rain
Answer
Dense Passage Retriever
MLP
What weather phenomenon most likely happened?
Q-Former
ViT
Visual Prefix
<BOQ> What weather phenomenon most likely happened? <EOQ>
<BOC> a man sitting on a bench in a flooded park
<BOV> wood brown red bench <SOV> large tall green tree <SOV> calm gray water ... <SOV> [OCR text if any] <EOV>
<BOK> flash floods occur within six hours of a rain event, or after a dam or levee failure, and flash floods can catch people unprepared ... <EOK>
Document 1
Flan-T5
flood
Visual Prefix
<BOQ> What weather phenomenon most likely happened? <EOQ>
<BOC> a man sitting on a bench in a flooded park
<BOV> wood brown red bench <SOV> large tall green tree <SOV> calm gray water ... <SOV> [OCR text if any] <EOV>
<BOK> the most common cause of flooding is water due to rain and/or snowmelt that accumuates faster than soils can absort it or ... <EOK>
Document 2
Flan-T5
rain
...
...
Visual Prefix
<BOQ> What weather phenomenon most likely happened? <EOQ>
<BOC> a man sitting on a bench in a flooded park
<BOV> wood brown red bench <SOV> large tall green tree <SOV> calm gray water ... <SOV> [OCR text if any] <EOV>
<BOK> these types of storm are hard to predict ... floods occur due to rain and other water rising faster than the drains can handle. <EOK>
Document 5
Flan-T5
storm
Question
Text-based Vision
Answer Candidates
flood
Answer

(a) Image and question

(b) Image, question and text-based vision

(c) Image, question, text-based vision and documents

Fig. 3.5 LLM can be fine-tuned with various kinds of inputs. Figures show pipeline used with InstructBLIP's visual encoder.

with a trainable mapping network and an LLM. There are several options for the text-based input x to the LLM: it can be a question (Figure 3.5a), a question with text-based vision (Figure 3.5b), or a question with text-based vision and documents (Figure 3.5c). The training loss has the general form

$$\mathscr{L}_{\text{fine-tune}} = -\sum_{t=1}^{L} \log p\left(s^t \,|\, [V, F_L(x)]\,, s^{<t}\right) \tag{3.10}$$

For CLIP and BLIP2, their image prefix V has the same form as in pre-training. For Instruct-BLIP, its prefix is $V = F_M(F_Q(F_V(I), F_L(q)))$ where q is the question. When documents are used, the same RA-VQA framework as described in Section 3.1.3 is used with the addition of visual prefix, as shown in Figure 3.5c

3.3 In-context Learning

The few-shot learning framework is a continuation of the concepts introduced in RA-VQA (Section 3.1.3). In RA-VQA, multiple documents are retrieved as sources of knowledge, and an LLM is trained to generate an answer for each document. However, these answers are not perfect, they can be incorrect or irrelevant. We want to utilize the strong general language understanding and reasoning ability of a larger language model such as GPT-3.5 to filter out bad answer candidates. This would achieve a similar goal as RA-VQA, which selects the most appropriate answer candidate by combining answer generation likelihood and document retrieval score. Both methods aim to choose one answer from a few candidates, but RAVQA uses a specifically designed formula while in-context prompting uses the LLM's strong reasoning ability. Additionally, we expect the LLM to be able to use its implicit knowledge to provide an answer if none of the candidates are appropriate. Thus, both external knowledge source and implicit knowledge are used. We hope to achieve a balance between these two sources of knowledge.

For a frozen LLM to understand the format required for answers and the meaning of answer candidates, we provide in-context examples for few-shot learning. Thus, few-shot learning consist of two steps:

1. Use document-wise predictions of a fine-tuned RA-VQA model as answer candidates.
2. Compose in-context examples to prompt a frozen LLM for the final answer.

3.3.1 Answer Candidate Generation

Answer candidates come from the most relevant documents retrieved for a question. They can be interpreted as question-aware summaries of a document. The fine-tuned RAVQA model is trained to summarize a document to only the section that directly answers the question.

As described in Section 3.1.3, a fine-tuned document retriever produces the top K most similar documents to a question x. For each question, a set of documents $\{z_1,...,z_K\}$ are retrieved with relevance scores $\{p_\theta(z_1|x),...,p_\theta(z_K|x)\}$. The higher the score, the more similar the document encoding is to the question encoding. A set of answers $\{y_1,...,y_K\}$ is generated from the K documents. Since answers might be the same for different documents, only distinct answers are kept, with their associated document scores. We impose a maximum limit on the number of distinct answer candidates used $K_{AC} \leqslant K$. Thus, we obtain a set of distinct answers candidates $\{y_1,...,y_{K_{AC}}\}_{\neq}$ with the highest relevance scores $\{p_\theta(z_1|x),...,p_\theta(z_{K_{AC}}|x)\}$. Note that document scores might be the same for distinct answers. If the number of distinct answers is smaller than K_{AC}, then lower number of answer candidates are used.

3.3.2 In-context Few-shot Prompting

Each in-context example consists of an image caption, question, answer candidates and gold answer. An image caption is generated by an image captioning model that describe the image. It forms the context c to the question. Rather than generic image captions, we use question-aware captions generated by PromptCap which include details relevant to the question. This form of text-based vision is used instead of image encoding by a visual model because GPT-3.5 can only take text-based input. Answer candidates $\{y_i, p_\theta(z_i|x)\}_{i=1}^{K_{AC}}$ are distinct predictions from the most relevant documents. Related document scores are in brackets. The gold answer is the most common ground truth annotation. N in-context examples are used with each one following the same structure $\varepsilon = \{e_1, e_2, ..., e_N\}$.

Context: c
Question: q
Candidates: y_1 $(p_\theta(z_1|x))$, y_2 $(p_\theta(z_2|x))$,..., $y_{K_{AC}}$ $(p_\theta(z_{K_{AC}}|x))$
Answer: a

The testing example has the same components except no answer is available. Similar to document indexing, we use FAISS (Johnson et al., 2021) to find the most similar in-context examples. Given a testing input (v,q) with an image and question, there are various options to calculate the similarity between training and testing inputs. For example it is possible to use just the question encoding $F_\theta(q)$, a combination of question encoding and image encoding

$[F_\theta(q) : F_\phi(v)]$, or a combination of question encoding and caption encoding $[F_\theta(q) : F_\theta(c)]$, where [:] is concatenation. Let the testing feature encoding be z, then top N most similar feature encodings with corresponding index in training set are found

$$I = \underset{i \in \{1,2,\ldots,N\}}{\text{argTopN}} \left(z^T z_i\right) \tag{3.11}$$

where I is the the index set of top-N most similar features in the training set. Thus, the set of in-context examples are $\varepsilon = \{e_i \ : \ i \in I\}$

Lastly, a prompt head is added to provide information on the task and inputs. The prompt head explains the VQA task and what answer candidates are. Three prompt heads are tested as shown in Table 3.1. Prompt 1 follows that in Prophet (Shao et al., 2023). Prompt 2 encourages the frozen LLM to generate answers outside of the candidates list. Prompt 3 encourages the LLM to only use answer candidates.

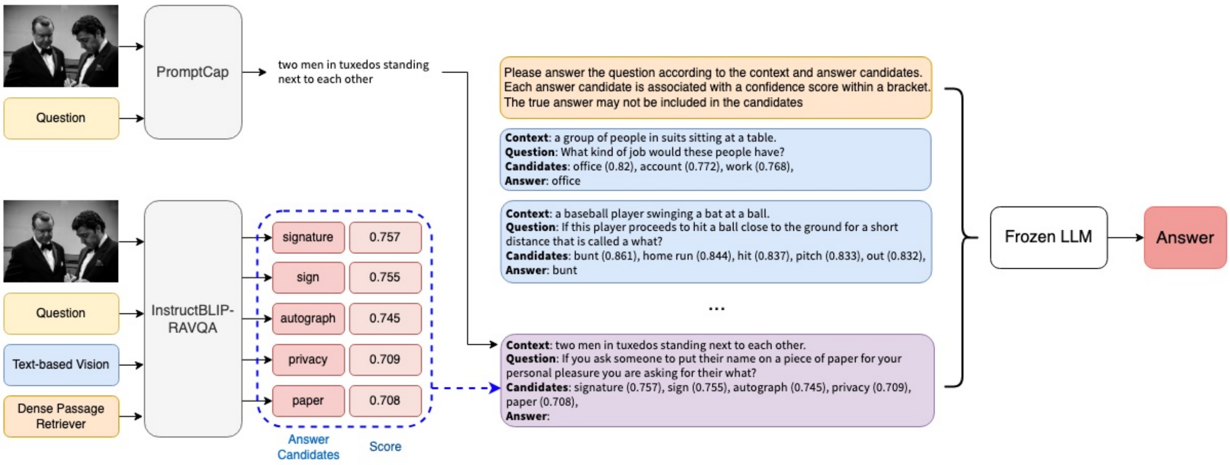
Table 3.1 Three prompt heads are tested for in-context learning.

Prompt 1	Answer the question according to the context and answer candidates. Each answer candidate is associated with a confidence score within a bracket. The true answer may not be included in the candidates.
Prompt 2	Answer the question according to the context and answer candidates. Each answer candidate is associated with a confidence score within a bracket. Come up with an answer if none of the answer candidates are suitable.
Prompt 3	Answer the question according to the context and answer candidates. Each answer candidate is associated with a confidence score within a bracket. Choose one answer from the candidates.

The inputs to the frozen LLM consists of a prompt head, in-context examples and testing input $\{p, e_i, .., e_N, e\}$. The LLM is expected to follow the format given in in-context examples and generate a single short answer selected from answer candidates, or an alternative answer from its implicit knowledge. The overall framework is shown in Figure 3.6.

3.4 Conclusion

This chapter describes two approaches to KBVQA. We adopt the RA-VQA (Lin and Byrne, 2022) framework and add a visual encoder for better visual understanding. We test various visual encoders and training strategies to find the best configuration for a visual-language retrieval-augmented model. The main training methods are



Answer Candidates	Score
signature	0.757
sign	0.755
autograph	0.745
privacy	0.709
paper	0.708

Please answer the question according to the context and answer candidates. Each answer candidate is associated with a confidence score within a bracket. The true answer may not be included in the candidates

Context: a group of people in suits sitting at a table.
Question: What kind of job would these people have?
Candidates: office (0.82), account (0.772), work (0.768),
Answer: office

Context: a baseball player swinging a bat at a ball.
Question: If this player proceeds to hit a ball close to the ground for a short distance that is called a what?
Candidates: bunt (0.861), home run (0.844), hit (0.837), pitch (0.833), out (0.832),
Answer: bunt

...

Context: two men in tuxedos standing next to each other.
Question: If you ask someone to put their name on a piece of paper for your personal pleasure you are asking for their what?
Candidates: signature (0.757), sign (0.755), autograph (0.745), privacy (0.709), paper (0.708),
Answer:

Fig. 3.6 In-context learning uses question-aware captions generated by PromptCap (Hu et al., 2023) and answer candidates generated by a fine-tuned visual-language RA-VQA model.

- Fine-tune LLMs with text-only input such as question only, question + text-based vision, and question + text-based vision + documents.
- Pre-train mapping network on image captioning task to bridge frozen visual and frozen language modalities.
- Fine-tune mapping network and LLMs on KBVQA task with or without document retrieval.

The second approach is to use in-context learning with a frozen LLM. We follow the framework in Prophet (Shao et al., 2023) but improve it with question-aware captions and external document-based answer candidates.

- Generate answer candidates from fine-tuned visual-language retrieval-augmented model.
- Generate question-aware captions from PromptCap (Hu et al., 2023).
- Select in-context examples based on similarity between features such as question encoding.
- Compose in-context prompts for a frozen LLM with different prompt heads.

In the next chapter, we describe the experiments conducted with datasets, models, hyper-parameters and evaluation metrics used.

Chapter 4

Experiments

Two types of experiments are described in this chapter, visual-language model fine-tuning and in-context learning for frozen language-only model. We will describe the datasets, models and hyperparameters used, as well as evaluation metrics used to measure model performance.

4.1 Datasets and Baseline Experiments

4.1.1 Datasets

OK-VQA (Marino et al., 2019) is a knowledge-based VQA dataset. It contains 9009 training questions and 5046 testing questions with corresponding images selected from COCO images (Lin et al., 2014). Some questions share the same images. The dataset is designed such that all questions require outside knowledge that is not immediately obvious from the image. However, this is not actually the case, as a large number of questions can be answered with object detection or common sense (Figure 5.3). Each question is annotated with 10 answers. The most popular annotated answer is the 'gold answer.'

Conceptual Captions (Sharma et al., 2018) contains 3.3 million links to online images. Image captions are extracted from raw descriptions from the web. Since certain URLs become unavailable as time go by, a subset of Conceptual Captions is used for mapping network pre-training.

4.1.2 Q-only Baseline

In order to identify the best LLM to use for following experiments, we start by fine-tuning various LLMs on OKVQA question-answer pairs. Three LLMs are tested: T5-Large (Raffel et al., 2020), Flan-T5-Large and Flan-T5-XL (Chung et al., 2022). T5-Large was used in

RA-VQA (Lin and Byrne, 2022) as the main generation LLM. The T5 family of models with the 'Flan' prefix has been instruction fine-tuned to show improved performance in question answering tasks as compared to their predecessors. Flan-T5-Large and Flan-T5-XL are the main LLMs used in this report.

For all three models, the AdamW optimizer is used with linear learning rate starting from 6e-5, batch size of 2, and gradient accumulation steps size of 16. Detailed hyperparameters can be found in Appendix D. T5-Large and Flan-T5-Large have 780 million parameters that can be fine-tuned end-to-end with 1 Nvidia A100 GPU within 10 hours. However, Flan-T5-XL has 3 billion parameters, which makes it expensive to fine-tune end-to-end. Thus, LoRA (Hu et al., 2022) is used to reduce the number of trainable parameters.

In order to find the best LoRA hyperparameter, various rank r values are tested. The value of r directly controls the number of trainable parameters. For all LoRA experiments, the α value is 32 and the dropout rate is 0.1. The training time for OKVQA fine-tuning is recorded in all three aforementioned LLMs, with only Flan-T5-XL needing LoRA.

4.1.3 Frozen DPR

Pre-trained DPR from RA-VQA (Lin and Byrne, 2022) is frozen to retrieve the $K = 5$ most relevant documents for each question from an external knowledge source. The knowledge source used is the `GS-full` dataset released by Luo et al. (2021). It used Google Search API to search for question-answer pairs to make sure each answer is present in the knowledge base. Snippets from the top 10 web pages were returned and those longer than 300 words or shorter than 10 words were removed. Each document is decoded by `BERT-base` (Devlin et al., 2019), which is frozen. The knowledge base is indexed through FAISS (Johnson et al., 2021) (Section 3.1.3). FAISS allows fast nearest neighbours search so that during training, the most similar documents to a concatenation of question and text-based vision embedding are retrieved.

4.2 Mapping network training

4.2.1 ViT

This visual model is shortened as ViT from ViT-L/14, which comes from CLIP (Radford et al., 2021) that learns image representations through contrastive pre-training on image-text pairs from the internet. The dimension of image encoding is $d_V = 768$. The length of image-prefix is chosen to be $L = 32$ to be consistent with that of the QFormer. The mapping layer for ViT-L/14 is trained with Flan-T5-Large and Flan-T5-XL. The hidden dimension of Flan-T5-Large is $d_L = 1024$; that of Flan-T5-XL is $d_L = 2048$.

All available samples in the Conceptual Captions training set are used for pre-training. The pre-training process freezes the LLM and ViT, and only trains the mapping layer with a learning rate of 0.0003. Other hyperparameters can be found in Appendix D. Then the pre-trained mapping network is fine-tuned on OKVQA with the same LLMs. The mapping framework has the learning rate set at 0.0001, and the LLM at 0.00006. To investigate the necessity of the pre-training step, further experiments are conducted for all three visual encoders where the mapping network is initialized randomly at the beginning of OKVQA fine-tuning.

To compare text-based vision with visual encoding, a baseline experiment is first conducted with question and text-based vision as input. Then, image prefix and text-based vision are combined to investigate if both types of visual information are complementary, or if either type of vision renders the other unnecessary.

4.2.2 QFormer

In this report, QFormer stands for the output of Q-Former from BLIP2 (Li et al., 2023). BLIP2 uses ViT-g/14 encoding as input to QFormer. QFormer is a transformer that extracts the most useful visual information to be understoodby the LLM. QFormer outputs an image encoding of length $L = 32$, $d_V = 768$. The same LLMs and hyperparameters are used as in ViT.

Since ViT-g/14 followed by QFormer has a large number of parameters, generating image encodings is slow. Thus, a subset of 450.7k Conceptual Captions images are used to pre-train the mapping network. Each image is first reshaped to be 400x400 pixels with bicubic interpolation and converted to the RGB format before using BLIP2's preprocessing step, which includes further resizing and normalization operations. Pre-processed images then pass through ViT-g/14 and QFormer to generate encodings. The fine-tuning procedure uses OKVQA and is the same as that of ViT.

The pre-trained then fine-tuned QFormer with Flan-T5-XL used in this report has the same architecture as the full BLIP2 checkpoint[1]. However, the model weights are different for the LLM since BLIP2 was fine-tuned on various VQA datasets. We compare the difference between our visual-language model and BLIP2 by testing a frozen BLIP2 with various prompts.

[1] https://huggingface.co/Salesforce/blip2-flan-t5-xl

4.2.3 InstructQFormer

Due to having the same architecture as BLIP2's QFormer, InstructBLIP's QFormer output is called InstructQFormer in this report to distinguish the two. InstructBLIP further improves from BLIP2 by instruction fine-tuning QFormer (Dai et al., 2023). InstructQFormer can accept an instruction, which in this report is the question in OKVQA. Thus, InstructQFormer's output image encoding is tailored to the question. The same hyperparameters and training methods are used as in QFormer except 385.7k Conceptual Caption images are used for pre-training.

Similar to BLIP2, InstructBLIP released checkpoint[2] on Huggingface with a pre-trained mapping framework. It also has a different version from the LAVIS Github[3]. Since it is unclear how to choose between these two versions, they are both frozen and tested on OKVQA to find the better version for subsequent experiments.

4.3 Integrated System Evaluation

After conducting thorough experiments to compare the performances of various visual encoders, the best model (InstructBLIP from Huggingface) is selected for FrDPR and RAVQA experiments. In FrDPR, the pre-trained document retriever is frozen. Only the mapping network and LLM are trainable. During training, the document retriever is given question and text-based vision encoded by `BERT-base` to retrieve $K = 5$ documents. Each retrieved document in text form is combined with visual prefix, question, and text-based vision, and encoded with Flan-T5-XL's encoder. An answer is generated for each document. Since 5 parallel documents are passed into the generation process, the batch size becomes equivalent to 5 times the original size. Thus, the setting for the actual batch size is reduced to 1. All other hyperparameters are kept the same as in mapping network fine-tuning. Detailed hyperparameters can be found in Appendix D.

Finally, the document retriever is also fine-tuned to observe if a trainable document retriever will boost performance further. This is called RAVQA following the naming in (Lin and Byrne, 2022). Due to long training time required for FrDPR and RAVQA, only the full InstructBLIP checkpoint from Huggingface is fine-tuned, since it is the best performing model so far.

[2] https://huggingface.co/Salesforce/instructblip-flan-t5-xl

[3] https://storage.googleapis.com/sfr-vision-language-research/LAVIS/models/InstructBLIP/instruct_blip_flanxl_trimmed.pth obtained from https://github.com/salesforce/LAVIS/blob/main/lavis/configs/models/blip2/blip2_instruct_flant5xl.yaml

4.4 In-context Few-shot Learning

Following fine-tuning of the retrieval augmented model in Section 4.3, the model with the best OKVQA performance is used to generate answer candidates for few-shot learning. Firstly, we investigate how the number of documents retrieved in testing K_{test} impacts the relevance of retrieved documents and the quality of answer generated for each document. Then we generate question-aware captions with PromptCap (Hu et al., 2023) to provide text-based vision for each image. Once in-context examples are prepared, several ablation experiments are conducted to find the best prompt format. Lastly, testing is done on multiple 13B models and GPT-3.5 models.

4.4.1 Answer Candidate Quality Evaluation

Before conducting few-shot learning experiments, we need to first find the best possible answer candidates. Preliminary tests showed a lack of diverse answers when a low number of documents is retrieved: answers tended to be the same, regardless of the documents retrieved. Many questions only have one or two distinct candidates with 5 document, thus K_{test} is increased up to 50 so to increase the number of distinct answer candidates. When a large number of documents is retrieved, there is a large variance in the number of distinct answer candidates for a question. We impose a limit on the maximum number of answer candidates. Only the top $K_{AC} \leq K_{test}$ distinct candidates with the highest corresponding document scores are kept. The rest of the candidates are discarded. If the number of distinct candidates is smaller than K_{AC}, then all distinct candidates are used. We also investigate how K_{AC} impact the relevance of answer candidates. The relevance of documents and answer candidates are measure by recall described in Section 4.5.

4.4.2 Alternative In-context Answer Candidates

A potential issue arises from different distributions between training and validation answer candidates. In-context examples use OKVQA training questions while testing examples use OKVQA validation questions. Answer candidates for training questions are very accurate and not diverse with very few distinct answer candidates. This might lead the LLM to only choose from answer candidates rather than use its implicit knowledge when necessary. Three methods are proposed to mitigate this issue.

1. The prompt head (**Prompt 1**) in Prophet (Shao et al., 2023) includes the sentence "The true answer may not be included in the candidates." We test two alternative prompt heads as described in Section 3.3.2. **Prompt 2** explicitly asks for an alternative

answer: "Come up with an answer if none of the answer candidates are suitable." We also compare this with **Prompt 3** that encourages answer selection from candidates: "Choose one answer from the candidates."

2. We train a different InstructBLIP-RAVQA model for only 1 epoch with half of the original training samples. This model is used to generate answer candidates for training questions only. Since less training data is used, the answer candidates are expected to be less accurate, and closer to the distribution and accuracy of validation candidates. The in-context examples would include more cases where the gold answer does not appear in the answer candidates. We investigate if this method can encourage the frozen LLM to generate alternative answers. In-context answer candidates generated this way are called **Reduced training**

3. An alternative training loss is proposed to punish instances where the document is ignored in an attempt to increase the diversity in answer candidates. Since this method is not successful, the detailed methods and results are covered in Appendix A.

4.4.3 Image Caption Generation

Three types of image captions are tested. We generate captions for images in the OKVQA training and validation sets and compare how each kind of caption impact in-context performance.

1. **Oscar+ captions** (Li et al., 2020) is used in RA-VQA (Lin and Byrne, 2022) and Prophet (Shao et al., 2023).

2. **BLIP2 captions** are generated with OPT-2.7B version[4] since it has the best COCO captioning performance in the BLIP2 family of models (Li et al., 2023), and outperforms Oscar+ in COCO captioning.

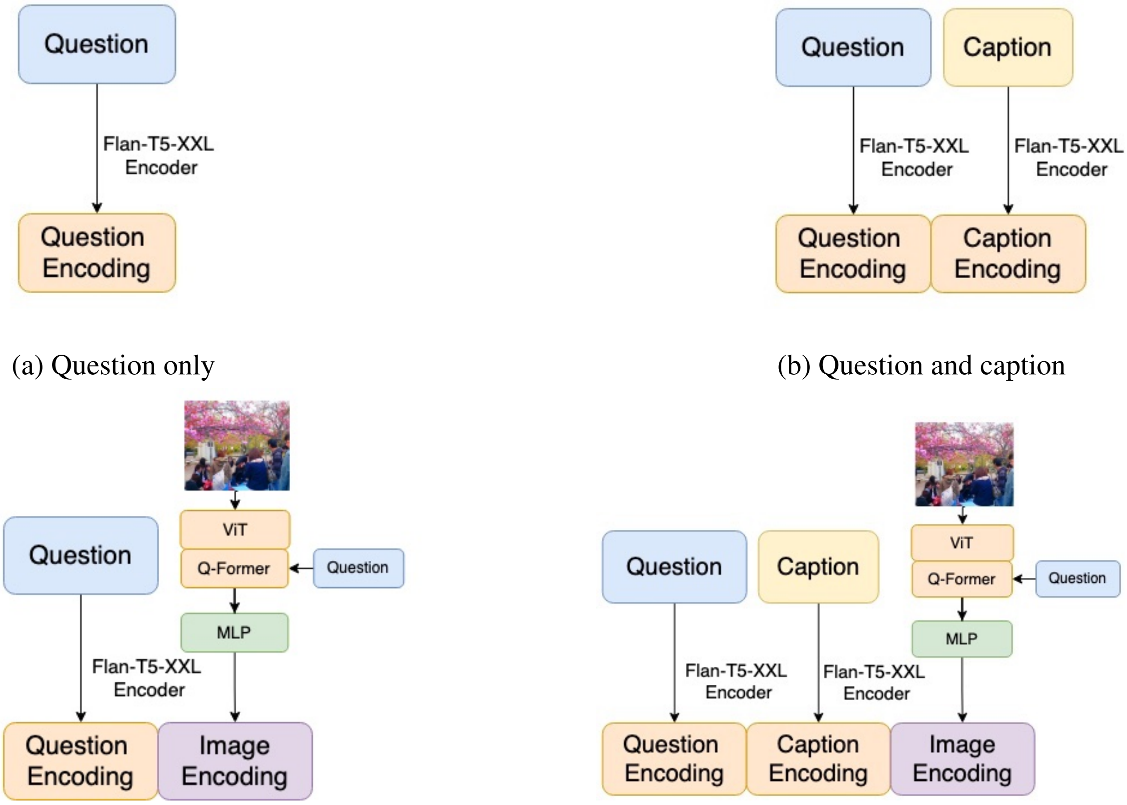
3. **PromptCap captions** (Hu et al., 2023) from the pre-trained model "vqascore/promptcap-coco-vqa"[5] can generate captions that contain details that the question is asking about.

4.4.4 In-context Hyperperameter Tuning

Before using GPT-3.5, we start with Flan-T5-XXL to conduct hyperparameter tuning since it is much cheaper. The models are frozen. The type of prompt head, captions and in-context similarity criteria are varied to find the best input combination.

[4] https://huggingface.co/Salesforce/blip2-opt-2.7b

[5] https://github.com/Yushi-Hu/PromptCap



(a) Question only

(b) Question and caption

(c) Question and image

(d) Question, caption and image

Fig. 4.1 Encoding of various types of features can be used to create index in FAISS for in-context examples selection. All text-based features and visual features are encoded to be 32×1024

In-context examples are selected using FAISS index as described in Section 3.3.2. Distance between various types of features can be used to create FAISS index. We use Flan-T5-XL to create encoding for question and captions. Each text encoding is padded to be 32 × 1024 so that they are the same shape as image encoding. We use InsturctQFormer to create question-aware image encoding. As seen in Figure 4.1, we test on using only question encoding, question + caption encoding, or question + caption + image encoding as features to create FAISS index.

We also test on various LLMs to examine their few-shot ability. Beside Flan-T5-XXL, multiple 13B models are tested including LLAMA2, LLAMA2-chat (Touvron et al., 2023) and Vicuna (Chiang et al., 2023). Ideally, we would test on even larger open-sourced LLMs such as LLAMA2-70B. However, this is not possible due to limited memory.

4.4.5 GPT-3.5 Tests

Two types of GPT-3.5 models are used for in-context learning, GPT-3.5-Turbo and Text-Davinci-003[6]. GPT-3.5-Turbo claims to be the most capable GPT-3.5 model and is optimized for chat. Text-Davinci-003 is 10 times the cost of GPT-3.5-Turbo. We choose to use it despite the high cost because it is similar to Text-Davinci-002 used in Prophet (Shao et al., 2023). Text-Davinci-003 is not optimized for chat and can follow instructions consistently.

Since the main purpose of using GPT-3.5 is to utilize its implicit knowledge, we design metrics to measure how often answers are chosen from answer candidates, and how often alternative answers are generated from the model's implicit knowledge. This metric is detailed in Section 4.5.5. We investigate how different prompt heads (Section 4.4.2), answer candidates, number of answer candidates and number of in-context examples affect GPT-3.5's performance.

4.5 Evaluation Metrics

4.5.1 VQA Score

Performance of each model on the OKVQA datasets is measured by VQA Score (Goyal et al., 2017)

$$\text{VQAScore}(y,S) = \min\left(\frac{\#s(y)}{3}, 1\right) \tag{4.1}$$

[6] https://platform.openai.com/docs/models/gpt-3-5

where $\#s(y)$ is the number of answers that are y. Since 10 human annotations are not always the same, this 'soft' score awards a model prediction as long as it matches one of the human annotations.

4.5.2 Pseudo Relevance Recall (PRRecall)@K

Since the answer candidates for few-shot learning are generated from each document, it is important to understand the relevance of documents retrieved. PRRecall proposed in (Lin and Byrne, 2022) measures the probability of a set of K retrieved documents containing at least one ground truth answer.

$$\text{PRRecall@K} = \min\left(\sum_{K=1}^{K} H(z_k, S), 1\right) \tag{4.2}$$

$H(z,S)$ is the pseudo relevance function for documents which is 1 if a document z contains any answer in a set of human annotated answers for a question, otherwise it is 0.

4.5.3 Answer Candidates Recall (ACRecall)@C

Similar to PRRecall, we propose ACRecall to measure the relevance of answer candidates, which is the probability of a set of K_{AC} answer candidates that contain at least one ground truth answer.

$$\text{ACRecall@C} = \min\left(\sum_{c=1}^{K_{AC}} H'(AC_c, S), 1\right) \tag{4.3}$$

$H'(AC,S)$ is the pseudo relevance function for answer candidates which is 1 if an answer candidate AC exactly matches any answer in a set of human annotated answers. It is 0 otherwise. The number of answer candidates $K_{AC} \leq K_{test}$ is the maximum number of distinct answer candidates allowed in a set of K_{test} answer proposals.

4.5.4 Hit Rate

Following Shao et al. (2023), hit rate is used to measure the best possible VQA Score for few-shot learning. Each answer candidate AC_c out of C total candidates is given a VQA Score according to whether it matches a ground truth answer. The hit rate for a certain question is the score of the answer candidate that gets the highest VQA Score. Thus, given C candidates in few-shot learning, if the LLM always chooses the candidate that has the highest VQA

Score, hit rate will be the overall VQA Score.

$$\text{Hit Rate} = \underset{AC_c, c=1,\ldots,C}{\operatorname{argmax}} \text{VQAScore}(AC_c, S) \tag{4.4}$$

4.5.5 External Knowledge Score (EKS)/Implicit knowledge Score (IKS)

In order to find out how likely a in-context model selects an answer from candidates, and how likely it generates alternative answers from implicit knowledge, we use Answer From Candidates (AFC) to measure the ratio of answers selected from candidates

$$\text{AFC} = \frac{\text{number of predictions that appear in answer candidates}}{\text{total number of testing questions}} \tag{4.5}$$

For answers that come from answer candidates, we use External Knowledge Score (EKS) to measure their accuracy. EKS is the VQA Score for those answers. For the rest of the answers, we use Implicit Knowledge Score (IKS) to measure the VQA Score of answers that do not appear in answer candidates. EKS and IKS add up to the overall VQA Score.

$$\text{VQA Score} = \text{AFC} \times \text{EKS} + (1 - \text{AFC}) \times \text{IKS} \tag{4.6}$$

4.6 Conclusion

In this chapter, we described the experiment setup for visual-language model fine-tuning and in-context learning. For fine-tuning, we use Flan-T5-Large and Flan-T5-XL with ViT, BLIP2 and InstructBLIP visual encoders. We bridge visual and language modalities with a linear layer which is pre-trained using Conceptual Captions. We incorporate visual encoders into the RA-VQA framework for better visual understanding. For in-context learning, we use Flan-T5-XXL and GPT-3.5. We vary the type of prompts, type of captions, type of answer candidates and type of in-context examples to investigate how they affect model behavior. In the next chapter, we will present and discuss experiment results.

Chapter 5

Results

Table 5.1 shows the results in this report compared to state-of-the-art results in the OKVQA dataset. We divide the methods into three categories, visual-language models, document retrieval based methods and in-context prompting. We can see that our InstructBLIP model without retrieval is smaller than other visual-language models but achieves similar performance. Once retrieval process is added, InstructBLIP-RAVQA improves by around 4% and outperforms all other retrieval-based models. InstructBLIP-RAVQA (4B) places the 4th place on the OKVQA leaderboard after much larger models PaLM-E (562B), PaLI-X (55B) and PaLI (17B) (Chen et al., 2023a,b; Driess et al., 2023). This shows that our method to incorporate visual encoders to RA-VQA is effective.

Our in-context learning method also outperforms all other in-context learning based methods. The currently best performing model Prophet (Shao et al., 2023) uses 5-fold ensembling to achieve 61.10%. We surpass this performance with no ensembling. This shows that our in-context methodology that combines PromptCap (Hu et al., 2023), Prophet and RA-VQA (Lin and Byrne, 2022) methods is effective.

In this chapter, we break down Table 5.1 and present results step by step.

5.1 Baseline performance with text-only features

Table 5.2 shows that instruction fine-tuned Flan-T5-Large outperforms T5-large in both Q-Only and FrDPR, despite having the same number of parameters. There is a larger improvement in FrDPR with 2.6% increase compared to Q-Only with 1.15% increase. This is due to Q-Only not providing the LLM with any visual information. Predictions from LLM are generally random guesses that relates to what the question is asking for and are devoid of meaningful understanding of the context. FrDPR includes text-based vision, providing

Table 5.1 Results achieved in this report vs methods in literature on OKVQA. Results from this report are in bold. Encoding means image encodings from visual models. Text-based vision (TBV) means text that describes the image such as caption, object description and OCR. Google means Google search results are used as an external knowledge base.

Model	Visual Features	Knowledge Source	VQA Score
Visual-language model			
PaLM-E (562B)	Encoding	-	66.1
PaLI-X (55B)	Encoding	-	66.1
PaLI (17B)	Encoding	-	64.5
InstructBLIP (7B)	Encoding	-	62.1
PaLM-E (12B)	Encoding	-	60.1
InstructBLIP (4B)	Encoding+TBV	-	58.91
Retrieval based methods			
TRiG	TBV	Wikipedia	50.50
KAT (Ensemble)	TBV	GPT-3+Wikidata	54.41
RA-VQA	Encoding+TBV	Google	54.48
REVIVE (Ensemble)	Encoding+TBV	GPT-3+Wikidata	58.0
FLMR	Encoding+TBV	Google	62.1
InstructBLIP-FrDPR	Encoding+TBV	Google	62.05
InstructBLIP-RAVQA ($K_{test} = 5$)	Encoding+TBV	Google	62.51
InstructBLIP-RAVQA ($K_{test} = 50$)	Encoding+TBV	Google	62.83
InstructBLIP-RAVQA-add1	Encoding+TBV	Google	62.94
In-context prompting			
Prophet (20-shot, Single)	Caption	GPT-3	57.91
PromptCap (16-shot)	Caption	GPT-3	58.4
PromptCap (32-shot)	Caption	GPT-3	60.4
Prophet (20-shot, Ensemble)	Caption	GPT-3	61.10
Text-Davinci-003 (5-shot)	Caption	GPT-3.5+Google	60.24
GPT-3.5-Turbo (5-shot)	Caption	GPT-3.5+Google	61.11
Flan-T5-XXL (5-shot)	Caption	Google	61.69

visual context that boosts performance. Due to the superiority of Flan-T5-Large, subsequent experiments are only conducted on Flan-T5 models.

Table 5.3 shows that LoRA greatly reduces the number of trainable parameters. For Flan-T5-Large, the number of trainable parameters in LoRA is 0.308% of the original number of parameters with $r = 8$. Training time to convergence also more than halved. However, there is 0.7 VQA score drop compared to fine-tuning the model end-to-end. When r is doubled to 16, there is less drop in performance (0.38%). This can be explained by more

Table 5.2 Baseline performance on OKVQA with text-only inputs. Models are fine-tuned end-to-end. Q-Only uses only questions as input. FrDPR uses questions, text-based vision, documents and freezes DPR.

	VQA Score	
LLM	Q-Only	FrDPR
T5-Large	26.85	51.63
Flan-T5-Large	28.00	54.23

Table 5.3 LoRA is an effective tool to reduce the number of trainable parameters and training time. Results show Q-Only performance. Training time shows time to converge when testing performance does not increase for two consecutive epochs.

LLM	Number of Trainable Parameters (Million)	Training time (mins)	Rank (r)	VQA Score
Flan-T5-Large	780	455	NA	28.00
	2.4	163	8	27.30
	4.7	218	16	27.62
Flan-T5-XL	4.7	67	8	30.79
	9.4	31	16	30.03

trainable parameters allowing the model to have more capacity to fit OKVQA data. This loss in performance is small, but to ensure optimal model performance, further experiments using Flan-T5-Large still fine-tune it end-to-end.

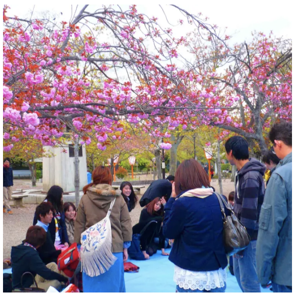
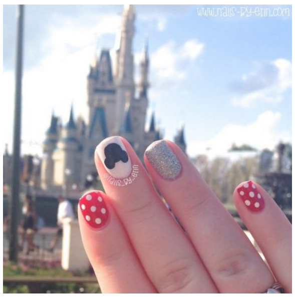
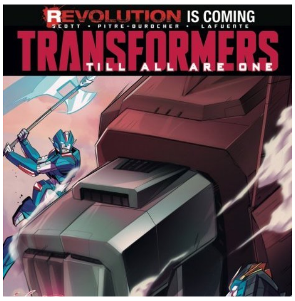
However, fine-tuning Flan-T5-XL end-to-end is undesirable due to long training time required. Using LoRA with $r = 8$ reduces training time by half compared to the Flan-T5-Large with the same LoRA setting. This is not due to shorter training time per epoch, but because much fewer epochs are required for convergence. Due to larger and better capacity for question understanding in the XL model, it converges faster when only questions are given.

For Flan-T5-XL, $r = 16$ reduces performance by 0.76 compared to $r = 8$. This could be due to the variability in performance between runs, and also shows that it is unnecessary to use $r = 16$. Thus, all further experiments with Flan-T5-XL use LoRA $r = 8$.

Table 5.4 Pre-training performance on Conceptual Captions dataset. Only the mapping network is trainable. Visual encoders and LLMs are frozen. A prompt "A photo of" is used in all experiments. The loss between generated captions and ground truth captions measures performance.

LLM	Visual Prefix Type	Loss
Flan-T5-Large	ViT	2.911
	QFormer	2.677
	InstructQFormer	2.649
Flan-T5-XL	ViT	2.651
	QFormer	2.339
	InstructQFormer	2.333

Table 5.5 Examples of captions generated by various visual encoders.

			
Ground Truth	party in the park under cherry blossoms.	where 's the best place to show off your nails ? right in front of the castle , of course !.	transformers : till all are issue # 4b.
ViT	people, a group of people, and a group of people, all in the same place.	i love the idea of a tatoo on the nails!.	person : the comic book artist, creator and illustrator of the comic book series.
QFormer	people are enjoying the cherry blossoms.	i love the idea of a polka dot manicure.	the cover by person for the #.
Instruct-QFormer	people gather under the cherry blossoms.	i love the idea of a polka dot manicure.	the cover of the comic book, transformers.

5.2 Comparison of various types of visual prefix

5.2.1 Mapping network pre-training

Captioning performance is hard to quantify since there is more than one way to caption an image. This issue is exacerbated by the fact that the annotation of Conceptual Captions has varying quality. This report uses the loss between generated captions and ground-truth captions as an approximate metric for captioning performance. Table 5.4 shows pre-training performance on Conceptual Captions. Flan-T5-XL achieves lower loss than Flan-T5-Large. This is expected since larger model capacity leads to better generation ability. InstructQFormer achieves lower loss than QFormer, which achieves lower loss than ViT for both model sizes. Table 5.5 shows that some ViT captions are repetitive or incoherent.

With Flan-T5-XL, Qformer and InstructQformer have very close loss. Table 5.5 shows that the captions they generate are similarly sensible. It is notable that InstructQFormer is the only encoder that managed to recognize the text in the third picture. It is hard to conclude that InstructQFormer has better text recognition ability without a thorough examination of captioning outputs. However, we can conduct further experiments on the OKVQA dataset to further evaluate each model's image understanding ability.

Table 5.6 The effect of pre-training on VQA score. Pre-training on captioning task only trains the mapping network and freezes the LLM. OKVQA fine-tuning without pre-training trains the mapping network from random weight initialization. OKVQA fine-tuning with pre-training loads the pre-trained mapping network weights and fine-tunes them with the LLM.

LLM	Visual Model	Pre-training Loss	VQA score	
			Without Pre-training	With Pre-training
Flan-T5-Large	ViT	2.911	36.71	41.59
	QFormer	2.677	34.82	50.01
	InstructQFormer	2.649	33.07	51.49

5.2.2 Image Prefix vs Text-Based Vision

Table 5.6 shows that fine-tuning Flan-T5-Large on OKVQA gives a large boost in performance with pre-training. When the mapping framework is trained from random initialization, all three visual prefix types yield VQA Score around 35%. With a pre-trained mapping framework, performance increases to more than 40%. This shows that pre-training with frozen LLM reshapes the image prefix to a form that is understandable by the LLM, so that the image prefix is already ‘language like’ at the start of OKVQA fine-tuning. Thus, visual prefix adds more useful information to the LLM.

With pre-training, QFormer scores 8.42 more than ViT, showing that QFormer is a much better choice of image prefix. This could be due to QFormer using output from ViT-g/14, which is shown to outperform ViT-L models on ImageNet classification (Zhai et al., 2022b). QFormer was also trained on image-text contrastive learning, image-grounded text generation, and image-text matching tasks (Li et al., 2023) so that it can translate image features into text form better than CLIP's ViT-L/14 that only employs contrastive learning. InstructQFormer further takes in a question and produces image features that cater to what the question is asking about, further boosting performance by 2.16 compared to QFormer.

Table 5.7 Fine-tuned models on OKVQA with pre-trained mapping framework. Only question and image prefix are used as input. Visual encoders are frozen. Mapping framework and LLM are fine-tuned.

Visual Prefix Type	LLM	VQA Score
None	Flan-T5-Large	28.00
	Flan-T5-XL	30.79
ViT	Flan-T5-Large	41.59
	Flan-T5-XL	46.25
QFormer	Flan-T5-Large	50.01
	Flan-T5-XL	54.71
InstructQFormer	Flan-T5-Large	51.49
	Flan-T5-XL	56.90

Table 5.7 further compares the Large and XL versions of Flan-T5, given various visual prefixes. As expected, Flan-T5-XL always performs better than Flan-T5-Large, with other variables kept constant. It is especially interesting that the largest boost in performance comes from InstructQFormer. Its performance increases by 5.41 from Large to XL, which is the largest increase of all three visual prefixes.

Table 5.8 shows that with only text-based vision, Flan-T5-Large scores 47.6%. With only ViT prefix, performance is worse at 41.59%, showing that ViT prefix does not contain as much useful information as text-based vision. However, both QFormer and InstructQFormer prefix achieves higher than 50%, showing that they add better visual information than text-based vision. The best performance comes from using visual prefix with text-based vision together, showing that the two types of visual information are complementary. Adding text-based vision to QFormer or InstructQFormer boosts performance by 2.33 and 1.83.

Table 5.8 Using text-based vision with image prefix boosts performance further. Text-based vision includes captions, objects and OCR recognition. The LLM is fine-tuned end-to-end with mapping framework. Visual encoders are frozen.

LLM	Visual Prefix Type	Text-based Vision	Visual Prefix	VQA Score
Flan-T5-Large	None	✗	✗	28.00
		✓	✗	47.60
	ViT	✗	✓	41.59
		✓	✓	48.83
	QFormer	✗	✓	50.01
		✓	✓	52.34
	InstructQFormer	✗	✓	51.49
		✓	✓	53.32

5.3 Integrated System Evaluation

Table 5.9 Comparison of frozen and fine-tuned BLIP2 and InstructBLIP. Full checkpoints are loadeds from online source except for those that use pre-trained mapping network from Section 4.2 with Flan-T5-XL. Checkpoint versions are described in Section 4.2.2 and 4.2.3. Short prompt is "Question: {Question} Answer:". Long prompt is "Use the provided image to answer the question: {Question} Provide your answer as short as possible: "

Model	Trainable	Model Source	Prompt	VQA Score
BLIP2	Frozen	Huggingface	No prompt	25.1
			Short prompt	27.89
			Long prompt	29.08
	Fine-tuned	Pre-trained MLP	No prompt	54.71
InstructBLIP	Frozen	LAVIS Github	No prompt	40.63
		Huggingface	No prompt	41.52
	Fine-tuned	Pre-trained MLP	No prompt	56.90
		Huggingface	No prompt	58.16

Various versions of BLIP2 and InstructBLIP are compared in Table 5.9. The full checkpoints released in Huggingface are compared with the pre-trained then fine-tuned model in this report. Firstly, BLIP2-Flan-T5-XL from Huggingface is tested with various prompts. Initial experiments found that the model likes to answer questions with full sentences, which impacts its score greatly since VQA Score requires the answer to match up with the ground truth labels exactly. Thus, a prompt that requires the answer to be as short as possible is used. The frozen BLIP2 performance indeed improved with this prompt.

Next, two frozen versions of InstructBLIP are tested. One is released on LAVIS's Github page, one on Huggingface as described in Section 4.2.2 and 4.2.3. Results show that the Huggingface version has slightly better frozen performance. Thus, it is fine-tuned on OKVQA to obtain 58.16 score. This is 1.16 higher than the score obtained with pre-trained-then-fine-tuned model. Despite having the same Flan-T5-XL architecture, InstructBLIP has been fine-tuned on various VQA datasets, as described in Dai et al. (2023), giving it better performance.

Table 5.10 Ablation results of fine-tuned models on various inputs. Full InstructBLIP checkpoint from Huggingface is used as training starting point. Training and testing both use 5 documents.

Model	Question	Visual Prefix	Text-based Vision	Documents	VQA Score
Flan-T5-XL	✓				28.00
InstructBLIP	✓	✓			58.16
	✓	✓	✓		58.91
InstructBLIP-FrDPR	✓	✓	✓	✓	62.05
InstructBLIP-RAVQA	✓	✓	✓	✓	62.51

The results of the fine-tuned Flan-T5-XL models on various inputs are shown in Table 5.10. Training with a question and image-prefix has VQA Score 58.16. Adding text-based vision adds 0.75 points. FrDPR retrieves 5 documents, and results in further performance boost by 3.14. This is slightly lower than the same ablation experiment presented in Lin and Byrne (2022) that showed 5.06 improvement with the addition of frozen retriever. Similarly, improvement from FrDPR to RAVQA is 0.46, lower than 2.59 in Lin and Byrne (2022). However, considering the already high baseline performance in InstructBLIP, it becomes increasingly difficult to get further improvement. Thus, it is reasonable to get smaller improvement from the addition of documents.

5.4 In-context Learning

This chapter presents results from in-context prompting of frozen LLMs such as GPT-3.5. Fine-tuned InstructBLIP-RAVQA from Section 5.3 is frozen to generate document-wise answer candidates. The answer candidates along with the image captions make up each in-context example. It is expected that the LLM will follow the patterns in in-context examples by picking an answer from the candidates, or will use its implicit knowledge when necessary to generate an alternative answer. We start with evaluating the quality of documents, answer candidates, and image captions. Next, we will vary different elements of in-context prompts to investigate how they affect system performance. Lastly, we will test the system on GPT-3.5 with more in-context examples and more answer candidates.

5.4.1 Answer Candidate Quality Evaluation

Table 5.11 Documents are more likely to include at least one ground truth answer when more documents are retrieved for testing K_{test}. InstructBLIP-RAVQA has better document retrieval performance than RAVQA from Lin and Byrne (2022). Training each model always uses $K_{train} = 5$. InstructBLIP-RAVQA is trained as in Table 5.10

K_{test}	RAVQA		InstructBLIP-RAVQA	
	PRRecall	VQA Score	PRRecall	VQA Score
5	82.84	53.81	84.78	62.51
20	93.62	54.2	93.62	62.69
50	96.47	54.45	97.05	62.83

Table 5.11 shows that InstructBLIP-RAVQA has higher PRRecall than RAVQA from Lin and Byrne (2022), meaning documents are more likely to contain ground truth answers. Since RAVQA showed optimal performance at $K_{test} = 50$, this report also tests K_{test} up to 50. When the number of documents retrieved for testing is increased, PRRecall and VQA Score both increase, showing that using more documents makes it more likely for the model to find the correct answer amongst them. It is important to retrieve a larger number of documents not only for better OKVQA performance, but to also get a more diverse range of answer candidates. Table 5.12 shows a lack of diverse answers when $K = 5$.

Table 5.12 shows an example of answer candidates for a question about the founding date of the Coca-Cola brand. The model generates the same answer for each document, despite the documents containing different possible years. The answer 1886 is not among ground truth answers. However, it is almost correct since 1886 is the date that the drink was first invented, rather than when the brand was founded. It is likely that the model ignored the documents, and used its implicit knowledge instead to generate the answer. The ideal behavior would be to generate different answers that correspond to the years mentioned in the document.

An explanation for InstructBLIP-RAVQA ignoring documents is that many questions do not actually require external knowledge. Many questions only require object recognition or

Table 5.12 Examples that demonstrate the lack of diversity in answer generation. Documents are shortened to contain only the parts that include numbers. Question: When was the cola brand on the signs founded? Ground truth annotations: 1892, 1892, 1892, 1892, 1851,1851, 1870's, 1870's, 1800, 1800

Prediction	Document Score	Document
1886	0.858	the history of coca cola dates all the way back to 1886. 1885 saw the birth of this still-popular soda, when it was created...
1886	0.849	coca-cola vs pepsi: the soda logo war by matthew roberts – on july 24, 2019 american culture is ... he established the "coca-cola company" in 1892, getting his trademark sign...
1886	0.849	. . . in 1892, the newly incorporated coca-cola company allocated $ 11,401 for advertising its drink...
1886	0.845	coca-cola was first introduced on may 8, 1886 by a pharmacist named ... he incorporated the coca-cola company in 1892 and...
1886	0.845	beginning with its birth at a soda fountain in downtown atlanta, georgia, in 1886, see all the milestones throughout coca-cola's memorable, 125+ year history.

common sense as shown in Figure 5.3. Thus, the performance of InstructBLIP in Table 5.10 without document retrieval is 58.91%. Due to a large number of questions in training data that can be answered without documents, the model learns to ignore documents even when they contain useful information. This highlights a limitation of the OKVQA dataset, which contains too many questions that might not require outside knowledge.

In order to work with the issue of lack of diverse answer candidates, we retrieve a larger number of documents to include more answer candidates. Since only distinct answer candidates are kept, more documents means more distinct candidates.

Figure 5.1 shows the distribution of the number of distinct answer candidates for train and validation sets when 50 documents are retrieved. Most of the training questions only have 1 or 2 candidates whereas validation questions are more likely to have more distinct answer candidates. This is an issue since in-context examples are drawn from the training set. Thus, a pattern forms where each in-context example contains an answer that comes from one of few answer candidates, more likely the first candidate. The language model might learn this pattern and only choose the first or second candidate in the testing question, while ignoring the rest of the candidates.

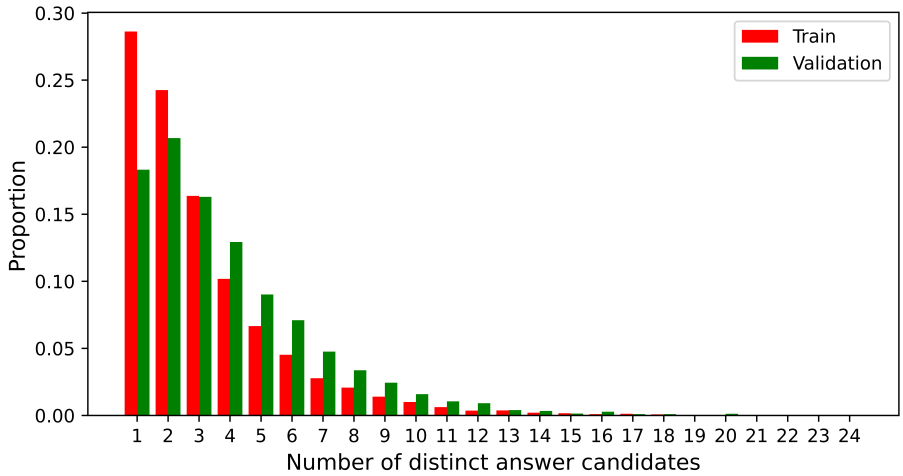


Fig. 5.1 Distribution of the number of distinct answer candidates for each question in training and validation splits in the OKVQA dataset. $K_{test} = 50$ documents are retrieved for each question. Occurrences of more than 24 answer candidates is not shown on graph since they are rare. There is disparity between the train and the validation distributions where it is much more likely for a training question to only have one or two answer candidates.

This issue is exacerbated by the fact that the training candidates have extremely high recall and hit rate. Table 5.13 shows that the training set ACRecall is 99.39, meaning that the answer candidates almost always include at least one ground truth answer. The answer given in in-context examples is the gold answer and while the gold answer might not always appear among a training question's answer candidates, the likelihood is very high. Thus, the LLM sees a pattern of only choosing an answer from answer candidates. Despite the prompt head stating 'The true answer may not be included in the candidates,' the trend in in-context examples do not indicate that. This impedes the LLM's ability to use its implicit knowledge to answer a question when non of the candidates are appropriate.

This shows the weakness in one of the main reasons to use a frozen LLM like GPT-3.5, which is to use its implicit knowledge source to complement explicit knowledge from documents. Three solutions are proposed to address this issue.

- The most simple solution is to use an alternative prompt 'Come up with an answer if none of the answer candidates are suitable.' We test three prompts as shown in Table 3.1. Results are shown in Table 5.20.

- Train InstructBLIP-RAVQA with reduced training data to intentionally worsen the ACRecall and Hit Rate of training questions. Results are shown in Section 5.4.5.

- Train InstructBLIP-RAVQA with alternative loss functions that punish instances where documents are ignored. This solution has limited success. Results are shown in Appendix A.

Table 5.13 Answer candidates are more likely to contain at least one ground truth answer as the maximum number of distinct candidates increase. Additional candidates become less accurate with more than 10 candidates. The overall quality of answer candidates is a lot higher than that in Prophet (Shao et al., 2023).

K_{AC}	Training Set		Validation Set		
	ACRecall	Hit Rate	ACRecall	Hit Rate	Prophet Hit Rate
1	94.77	90.65	66.31	61.86	53.04
2	98.28	95.77	80.00	75.61	
5	99.23	97.41	87.67	83.73	75.20
10	99.36	97.65	89.50	85.57	79.83
15	99.39	97.68	89.67	85.76	
max	99.39	97.68	89.71	85.78	

Despite aforementioned limitations, the overall quality of answer candidates is a lot higher than that in Prophet (Shao et al., 2023). Prophet relies on implicit knowledge from GPT-3 and do not use any explicit knowledge source. In our case, answer candidates come from external documents. They are a form of question-aware document summary. Thus, they are a lot more relevant than that in Prophet. Since Hit Rate is 8.5 points higher than that of Prophet with 5 answer candidates (Table 5.13), it is expected that our better answer candidates directly result in better few-shot performance.

Note that the number of answer candidates in Prophet is different from in this paper. In Prophet, the number of answer candidates is always the same for each question. However, in this report, this number varies between questions and a maximum cap is imposed by K_{AC}.

5.4.2 Image Caption Quality Evaluation

The image caption is an important component of in-context examples since it describes the context to each question. Since it is not possible to include image encoding directly to GPT-3.5, we need to use text-based vision. Three visual models are used to generate captions: Oscar+ (Li et al., 2020), BLIP2 (Li et al., 2023) and PromptCap (Hu et al., 2023). Originally, we wanted to use InstructBLIP to generate question-aware captions. However, preliminary tests showed that it did not work well. Details can be found in Appendix C.

Table 5.14 shows that Oscar+ captions are generally more detailed than that of BLIP2 despite BLIP2 achieving higher COCO Caption performance (Li et al., 2023, 2020). BLIP2's

Table 5.14 Examples of image captions by Oscar+, BLIP2 and PromptCap.

Question	What profession would you say this guy has?
Oscar+	a man is working on a motorcycle in front of a tent.
BLIP2	a man working on a motorcycle.
PromptCap	a man in blue overalls working on a motorcycle.
Question	The birds on the television derive their name from what country?
Oscar+	a cat sitting in front of a television watching birds
BLIP2	a cat sitting on a television
PromptCap	a cat sitting in front of a tv with a picture of geese on it.
Question	Where is this building located?
Oscar+	a large building with a clock tower on top of it.
BLIP2	a building with a clock tower.
PromptCap	a building with a clock tower in england.
Question	What does the blue p represent?
Oscar+	a black box sitting next to a brick wall.
BLIP2	a parking meter and a brick wall.
PromptCap	a parking meter with a blue p on it.
Question	What is the name of the beer?
Oscar+	a bottle of beer next to a plate of food
BLIP2	a beer and food
PromptCap	a bottle of kingfish beer and a plate of food.

short caption style likely aligns better with COCO's ground truth. BLIP2 also has better text-recognition ability (Table 5.5). BLIP2's better caption translates to better in-context performance. The baseline model that uses Oscar+ captions and performs worse than using BLIP2 captions (59.32 vs 59.55) in Table 5.15.

However, both Oscar+ and BLIP2 do not add enough relevant information that can help answer the question. In contrast, PromptCap provides captions that include details about what the question is asking about. As seen in Table 5.14, PromptCap describes the key object of interest in more detail. For example, when the question asks about birds on TV, PromptCap states that the birds are geese, whereas the other models do not include details about the birds. PromptCap also occasionally directly answers the question. This advantage is reflected in its better performance in in-context testing. It scores 1.2 better than BLIP2 (Table 5.15.)

5.4.3 Testing various in-context components

Table 5.15 Ablation on components for in-context prompting. One variable is changed at a time compared to the baseline. The baseline model has prompt head, confidence scores, $K_{AC} = 5$, $N = 5$, in-context similarity based on question embedding only. The LLM used is Flan-T5-XXL.

Variants	VQA Score
Baseline	59.32
(a) Without answer candidates or scores	32.50
(b) Without prompt head	59.24
(c) Without document confidence scores	57.17
(d) BLIP2 caption	59.55
(e) PromptCap caption	60.72

We first start with some ablation experiments to verify that each component of in-context examples are necessary. The baseline prompt is designed based on Prophet results that showed prompt head, confidence scores and image captions are integral parts of in-context examples. Performance degrades without them (Shao et al., 2023). We reach similar conclusions with Flan-T5-XXL in Table 5.15. A very large performance drop is observed without answer candidates (26.8%). A small performance drop is observed without prompt head (0.08%). A larger performance drop is observed without document confidence scores (2.2%).

Table 5.16 Selection of in-context examples based on distance between various types of features. The baseline model has prompt head, confidence scores, $K_{AC} = 5$, $N = 5$, in-context similarity based on question embedding only. Purely text-based features include the question and caption. Image embeddings come from the output of QFormer from InstructBLIP.

Variants	VQA Score
Question only	59.32
(a) Question+Oscar caption	59.64
(b) Question+BLIP2 caption	59.45
(c) Question+InstructQFormer embedding	59.66
(d) Question+Oscar caption+InstructQFormer embedding	59.75

We compare various features used to select in-context examples. The baseline method uses question encoding only and adding additional features results in better performance (Table 5.16). Interestingly, joint question and BLIP2 caption is worse than joint question and Oscar+ caption, despite BLIP2 resulting in better in-context performance as shown in

Table 5.15. We hypothesize that Oscar+'s more detailed captions better capture the content of images, thus more similar combined question and image pairs are found.

The content of images can also be captured by direct image embedding. InstructQFormer takes the question and image to generate image embeddings, which we combine with question embedding. This strategy performs better than text-based features by a small margin. Lastly, we try to combine question embedding, Oscar captions and image embedding together. This has the best performance 59.75% for few-shot learning. Thus, this in-context examples selection strategy is used in subsequent experiments[1].

5.4.4 Comparison of various LLMs for in-context learning

Table 5.17 Comparison of various LLMs of similar sizes for in-context learning. In-context examples have prompt heads, confidence scores, $K_{AC} = 5$, $N = 5$ and in-context similarity based on question + Oscar caption + InstructQFormer embedding.

LLM	VQA Score
Flan-T5-XXL (11B)	59.73
LLAMA2-13B	55.65
LLAMA2-13B-chat	55.92
Vicuna-13B	41.15

We test various LLMs with 13 billion parameters. Table 5.17 shows that the best LLM tested is Flan-T5-XXL. LLAMA2 performs worse, despite being released the most recently with claims of being a 'suitable alternative for closed-source models'.

Table 5.18 shows that LLAMA2 models interestingly continue generating after giving the answer. Since in-context examples end with the gold answer, and start with context, LLAMA2 fails to recognize that it should stop generating after answering, but follows the in-context pattern to guess the next context. Thus, post-processing had to be applied to only extract the answer. Additionally, we observe that LLAMA2-chat frequently put a period or line break symbol `\n` after the answer, despite none of in-context examples having those symbols after answers. This is likely due to the dialogue-based data that was used for training. Turn-based chat data end with periods or line breaks so this behavior is retained in in-context learning. This is undesirable since it shows that LLAMA2-chat fails to learn the expected answer format shown in in-context examples. Again, post-processing had to be done to remove periods.

[1] PromptCap caption is not tested for FAISS in-context example selection due to time limit

Table 5.18 Example of answers generated by various LLMs. Question: What fruit is typically added to the top of cereal?

Model	Generation content
Flan-T5-XXL	banana
LLAMA2-13B	banana Context: a bowl of cere
LLAMA2-13B-chat	banana. Context: a plate of pasta.
Vicuna-13B	banana

5.4.5 Alternative training scheme

Earlier we mentioned an issue in the disparity between training and validation answer candidates. Training answer candidates have close to perfect accuracy, which do not represent the accuracy in validation candidates. This might lead the LLM to only choose from answer candidates, rather than generate alternative answers when none of the candidates are appropriate. An alternative training scheme is designed to combat this issue. The amount of training data for InstructBLIP-RAVQA is reduced in half, and training is done for only 1 epoch instead of until convergence. This model is used only for generating training answer candidates. Since less training data has been seen by the model, the accuracy of training answer candidates is expected to be lower.

Table 5.19 Training Set - reduced indicates that half of the original training data is used to train InstructBLIP-RAVQA. The reduced training model is used to only generate training answer candidates with $K_{test} = 50$. Answer candidates from reduced training is a lot closer to the accuracy of validation candidates.

K_{AC}	Training Set		Training Set - reduced		Validation Set	
	ACRecall	Hit Rate	ACRecall	Hit Rate	ACRecall	Hit Rate
1	94.77	90.65	70.57	66.56	66.31	61.86
2	98.28	95.77	82.67	78.82	80.00	75.61
5	99.23	97.41	90.54	87.07	87.67	83.73
10	99.36	97.65	92.70	89.31	89.50	85.57
15	99.39	97.68	93.07	89.65	89.67	85.76
max	99.39	97.68	93.25	89.83	89.71	85.78

Table 5.19 shows that the reduced training scheme indeed lowers the ACRecall and Hit Rate of training answer candidates. The accuracy of training answer candidates is still higher than that of the validation candidates, but the two distributions are now a lot closer. Figure 5.2 shows that reduced training causes answer candidates to be more diverse. There are fewer questions with only one or two distinct candidates. This should teach the frozen LLM

to choose an answer from multiple answer candidates. It should also show that the answer candidates are not perfect. The LLM should come up with an answer when answer candidates are not appropriate. In-context results are presented in the next section.

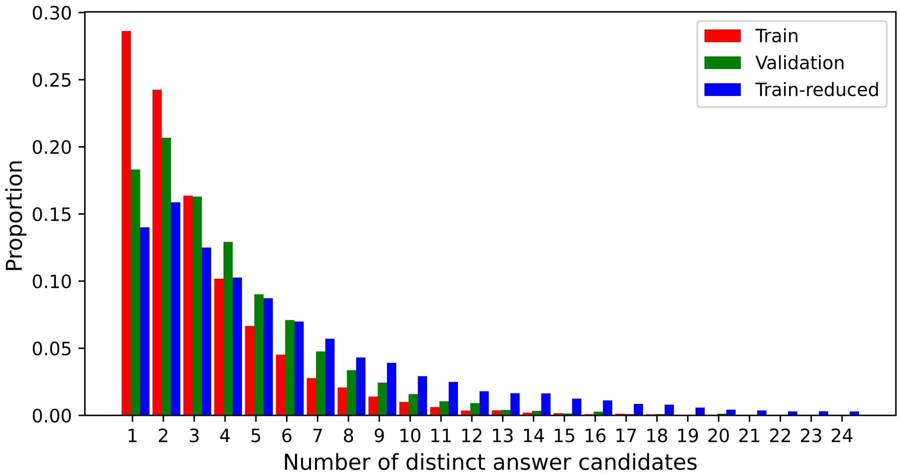


Fig. 5.2 Distribution of number of distinct answer candidates for each training question is a lot closer to that of validation question when less data is used to train a model to intentionally generate less accurate training answer candidates. $K_{test} = 50$ documents are retrieved for each question. Occurrences of more than 24 answer candidates is not shown on graph since they are rare.

5.4.6 GPT-3.5 performance

Finally, we test on GPT-3.5 to compare its in-context ability with that of Flan-T5-XXL. Table 5.20 shows that for the baseline setup, Flan-T5-XXL is the best performing model. This might seem surprising as GPT-3.5 is more than 10 times the size of Flan-T5-XXL and should have better in-context ability and more implicit knowledge. However, on closer look we can see that Flan-T5-XXL chooses from answer candidates 98% of the time. GPT-3.5 is much more likely to generate alternative answers. IKS is always lower than EKS showing that when models generate alternative answers from implicit knowledge, they are more likely to get the answer wrong. This is reasonable since easy questions can be answered by PromptCap captions, or by InstructBLIP-RAVQA. The in-context model can choose an answer for easy questions. The rest of the questions are difficult and sometimes not enough visual information is provided (Figure 5.3), thus EKS is low.

In Prophet (Shao et al., 2023), AFC is 89.9% for Text-Davinci-002, which is similar to 89.2% for Text-Davinci-003 in Table 5.20. Prophet's IKS is around 40% whereas IKS is

Table 5.20 Comparison of GPT-3.5 with Flan-T5-XXL. The baseline models use Prompt 1, answer candidates generated from InstructBLIP-RAVQA trained with full OKVQA train set, include confidence scores, $K_{AC} = 5$, $N = 5$, and use PromptCap captions. Reduced training uses less data to train an InstructBLIP-RAVQA model as described in Section 5.4.5.

Model	Type	VQA Score	AFC	EKS/IKS
Flan-T5-XXL	Baseline	61.16	0.9847	61.68/27.79
	Reduced training	61.16	0.9833	61.67/31.19
	Prompt 2	52.95	0.9558	54.51/20.18
	Prompt 3	61.25	0.9885	61.55/35.86
GPT-3.5-Turbo	Baseline	59.60	0.9457	62.14/15.33
	Reduced training	59.76	0.9259	63.36/14.81
	Prompt 2	59.18	0.9241	62.84/14.67
	Prompt 3	61.11	0.9558	63.09/18.12
Text-Davinci-003	Baseline	60.24	0.8922	63.71/31.36
	Reduced training	60.00	0.8783	63.87/31.04

31.36% in this paper, which is lower. This is expected since our answer candidates are more accurate and EKS is higher. As a result, we obtain better performance than Prophet with no ensembling (60.24% vs 57.91%).

In order to obtain higher in-context learning score, the best theoretical strategy would be to always choose from the answer candidates, since we know that answer candidates are very accurate (Table 5.13). Indeed, we see that Prompt 3 encourages choosing from candidates and results in higher AFC and VQA Score (61.25% vs 61.16% for Flan-T5-XXL, 61.11% vs 59.60% for GPT-3.5-Trubo). In contrast, Prompt 2 encourages alternative answers from implicit knowledge and results in lower AFC and VQA Score.

As an alternative to Prompt 2, we use reduced data to train an InstructBLIP-RAVQA model to generate less accurate answer candidates for in-context examples. This is less effective than using Prompt 2. We can see in Table 5.20 that AFC decreased marginally for Flan-T5-XXL (0.001), GPT-3.5-Turbo (0.020) and Text-Davinci-003 (0.014). Reduced training indeed created answer candidates that lead the LLMs to generate alternative answers. However, it is easier to implement, and more effective to use Prompt 2 instead. Prompt 2 results in decreased AFC by 0.029 for Flan-T5-XXL and 0.022 for GPT-3.5-Turbo.

Comparing Flan-T5-XXL, GPT-3.5-Turbo and Text-Davinci-003, we can see that they behave very differently in terms of AFC. When Prompt 2 is used for Flan-T5-XXL, its performance degrades drastically by 9%. It seems to understand the requirement for alternative answers. However, it also became worse at choosing from the answer candidates. The same issue does not occur for GPT-3.5-Turbo and Text-Davinci-003. EKS and IKS do not change drastically with different AFC. Overall, it seems that Text-Davinci-003 shows the best language understanding and reasoning ability. It has the highest EKS and IKS while understanding the requirement for alternative answers. Despite Flan-T5-XXL also having high EKS and IKS, it is unlikely to use its implicit knowledge.

Table 5.21 The effect of increasing the number of answer candidates K_{AC} on few-shot model performance. The baseline models use Prompt 1, include confidence scores, $K_{AC} = 5$, $N = 5$, and use PromptCap captions.

Model	K_{AC}	VQA Score
Flan-T5-XXL	3	61.69
	Baseline	61.16
	10	60.96
	15	61.05
GPT-3.5-Turbo	Baseline	59.60
	15	59.91

Table 5.21 shows the effect of including more answer candidates in in-context examples. For Flan-T5-XXL, when K_{AC} is increased, performance degrades. This shows that Flan-T5-XXL is not capable of making the right choice from more answer candidates. This also supports our hypothesis that better performance can be obtained when the model only chooses from the first few answer candidates. In contrast, GPT-3.5-Turbo yields improved performance with more answer candidates, showing its superior reasoning ability. It can distinguish between good and bad answer candidates and make appropriate choices.

Table 5.22 The effect of increasing the number of in-context examples N. The baseline models use Prompt 1 from Prophet, include confidence scores, $K_{AC} = 15$, $N = 5$, and use PromptCap captions.

Model	N	VQA Score
Flan-T5-XXL	Baseline	61.05
	10	61.18
	20	61.07
GPT-3.5-Turbo	Baseline	59.91
	20	60.03

Table 5.22 shows the effect of increasing number of in-context examples (shots) on model performance. For Flan-T5-XXL, few-shot performance is the best with 10 shots. This is due

to context window being 512 tokens during training (Chung et al., 2022). Thus, if the input sequence is longer than 512, Flan-T5-XXL might ignore the content in the middle of the input sequence. This phenomenon is explained in Liu et al. (2023b) where encoder-decoder models have difficulty accessing information in the middle of long sequences when the length of the sequence is longer than the model's pre-training context window. The input length is around 1000 tokens with 20 shots, meaning that Flan-T5-XXL can effectively make use of half of the input sequence. The input length is around 500 tokens with 10 shots, which fit Flan-T5-XXL's context window. This explains why 10 shots have the best in-context performance.

GPT-3.5-Turbo allows a maximum of 4096 tokens in the input. It is unclear how big its actual context window is, but it's likely to be much bigger than that of Flan-T5-XXL. Thus, there is no performance degrade with 20-shots for GPT-3.5-Turbo. At the same time, very little improvement is seen with more shots (improve by 0.12). This is similar to the result in Prophet (Shao et al., 2023) that showed improvement by 0.42 from 8 to 20 shots. We hypothesize that GPT-3.5 was able to learn the in-context examples' pattern and the expected answer format with a small number of shots. Thus, more shots do not result in big performance gain.

5.5 Conclusion

We present experiment results in this chapter. The following results are highlighted for visual-language fine-tuning and in-context learning. Visual-language fine-tuning results are as follows:

- LoRA is an effective tool to reduce the number of trainable parameters and training time.
- Flan-T5-XL shows better language generation ability than Flan-T5-Large and T5-Large.
- InstructBLIP (Dai et al., 2023) achieves the best captioning performance and OKVQA fine-tuned performance, followed by BLIP2 (Li et al., 2023) and ViT (Radford et al., 2021).
- Mapping network pre-training on captioning tasks is an essential step to bridge visual and language modalities.
- Text-based vision and visual prefix are complementary. Using both kind of visual representation together results in the best VQA performance.

Image	Prompt	Ground Truth Labels	Prediction	Knowledge Required
	Context: a black motorcycle parked in a parking lot. Question: What sport can you use this for? Candidates: race (0.848), motorcycle (0.83), ride (0.827), motocross (0.826), bike (0.805), Answer:	race, motocross, ride	motocross	Object detection External knowledge
	Context: a bathroom with a plant on the wall. Question: Name the type of plant this is? Candidates: fern (0.836), succulent (0.832), wall (0.828), house (0.82), vine (0.814), Answer:	vine, climb, look like some kind of ivy, ficus	fern	Object detection
	Context: a man with a backpack with a teddy bear in it. Question: What toy is this? Candidates: teddy bear (0.846), Answer:	stuffed animal, teddy bear	teddy bear	Object detection
	Context: a man holding a dog on his back. Question: Which part of this animal would be in use of it was playing the game that is played with the items the man is holding? Candidates: tail (0.783), mouth (0.755), arm (0.744), Answer:	mouth	mouth	Object detetion External knowledge
	Context: a bathroom with a toilet and a sink. Question: Who leaves a toilet like this? Candidates: man (0.85), person (0.836), human (0.805), clean (0.791), Answer:	man, men	person	Object detection External knowledge
	Context: a kitchen with a center affixed unit. Question: A center affixed unit like this one in a kitchen is called a what? Candidates: island (0.827), mixer (0.81), bowl (0.775), stove (0.769), Answer:	island	island	Object detection External knowledge
	Context: a busy city street with many people walking around. Question: Why might someone go to this place? Candidates: shop (0.792), crowded (0.761), advertising (0.742), night (0.74), Answer:	shop, nyc, business	shop	Object detection Common sense
	Context: a baseball player holding a bat. Question: What is that man doing with the bat? Candidates: bat (0.859), swing (0.843), hit (0.841), Answer:	swing, hit, try to hit hte ball	swing	Object detection
	Context: a group of people swimming in the ocean at a salt water beach. Question: Is this at a salt water beach or a lake? Candidates: salt (0.804), lake (0.796), both (0.782), Answer:	salt water beach, salt water, lake, beach	salt	Object detection
	Context: two hot dogs with onions and peppers. Question: What is the name of the items the hot dog are topped with? Candidates: relish (0.914), pickle (0.911), onion (0.881), condiment (0.867), Answer:	condiment, onion relish, vegetable, relish	onion	Object detection
	Context: a desk with four computers and a phone. Question: What is this desk used for? Candidates: work (0.867), computer (0.842), compute (0.824), office (0.816), Answer:	work, compute, office	work	Object detection Common sense
	Context: a passenger jet sitting on top of an airport tarmac. Question: What type of plane is that? Candidates: passenger (0.832), 747 (0.829), Answer:	commercial, passenger, quanta, md 80	passenger	Object detection
	Context: a display case filled with lots of different types of donuts in 2012. Question: In what year was this desert first introduced? Candidates: 1800 (0.774), 1950 (0.77), 1953 (0.757), 1914 (0.751), 1804 (0.748), Answer:	1847, 1860, 1934, 1900s	1804	Object detection External knowledge

Fig. 5.3 Examples of prompts for OKVQA validation questions. Prompt head and in-context prompts are not shown. Predictions are generated by Text-Davinci-003 with Baseline setup in Table 5.20. Knowledge required shows the author's subjective view of the necessary inforation required to answer each question.

- Incorporating InstructBLIP visual encoders into RA-VQA (Lin and Byrne, 2022) boosts OKVQA performance to be 62.83%, exceeding the performance of all other retrieval based models. InstructBLIP-RAVQA achieves 4th highest performance on the OKVQA leaderboard, after models of much larger sizes.
- InstructBLIP-RAVQA achieves better document retrieval performance than RA-VQA.

In-context learning results:

- Question-aware captions, answer candidates with document confidence scores, and a prompt head are integral parts of in-context prompting of frozen LLMs.
- Pre-trained PromptCap (Hu et al., 2023) can generate image captions that cater to the specific question. It is more effective than Oscar+ (Li et al., 2020) and BLIP2 (Li et al., 2023) captions for in-context learning.
- The best way to select in-context examples is to use the distance between combined question, image caption and image encoding.
- Flan-T5-XXL shows better in-context performance than LLAMA2-13B (Touvron et al., 2023) and Vicuna-13B (Chiang et al., 2023).
- The best strategy to encourage frozen LLMs to generate answers from implicit knowledge is to give a prompt with the instruction to do so. Similarly, the best strategy to encourage frozen LLMs to choose an answer from external document-based candidates is to give a prompt to do so.
- Flan-T5-XXL shows better in-context performance than GPT-3.5 models because it is much more likely to choose from very accurate answer candidates. Answers from implicit knowledge are more likely to be wrong since those questions are more difficult.
- Text-Davinci-003 shows the best general language understanding and reasoning ability since it achieves the highest VQA Score for answers that come from candidates, and for answers that come from implicit knowledge.
- GPT-3.5-Turbo shows better reasoning ability than Flan-T5-XXL since it can make use of more answer candidates and more in-context examples, whereas Flan-T5-XXL shows worse performance with more candidates and in-context examples.
- Our in-context prompting strategy that makes use of question-aware captions, external document-based knowledge and LLM's implicit knowledge outperforms other in-context methods on the OKVQA dataset. Our best in-context performance 61.69%

does not use ensembling and outperforms the current best in-context model Prophet (61.10%) that uses 5-fold ensembling.

Chapter 6

Future Work and Conclusion

6.1 Future Work

There are many aspects of the methods and experiments used in this report that can be improved. We discuss retrieval-augmented visual-language fine-tuning and in-context learning improvements separately.

Retrieval-augmented visual-language fine-tuning improvements:

- Text-based vision used in this report and in RA-VQA (Lin and Byrne, 2022) include Oscar+ captions. In in-context learning, we showed that PromptCap (Hu et al., 2023) captions are superior. We can replace Oscar+ with PromptCap captions to fine-tune InstructBLIP-RAVQA. Since PromptCap captions are more relevant, we might see a small improvement in InstructBLIP-RAVQA performance.

- We observed that in RA-VQA, the trained answer-generator frequently ignore documents and generate the same answer candidates for different documents. This might be improved by changing the underlying training mechanism. Currently, the document retriever and answer generator are updated using separate losses (Section 3.1.3) meaning that the answer generator is not punished when it ignores documents. If we can add a loss term to update the answer generator directly so that it is discouraged to ignore documents, we might be able to get more answer candidates that come from the documents and improve the ACRecall score.

- We observed that when less data is used to train InstructBLIP-RAVQA for only 1 epoch, answer candidates are more diverse (Section 5.4.5). Furthermore, as training continues, PRRecall decreases from second epoch onward (not shown in this report) but the VQA Score increases. This shows that the document retriever and answer generator training processes are disjointed. Further hyperparameter tuning of learning rates might improve this so that PRRecall increases throughout training.

- In this report, InstructBLIP is used with Flan-T5-XL which achieves 58.15% fine-tuned on OKVQA. In Dai et al. (2023), InstructBLIP with Vicuna-7B achieves 62.1% fine-tuned on OKVQA. Thus, we can replace Flan-T5-XL with Vicuna-7B for better performance. However, we need to ensure enough memory can be allocated for this model size.

- We use the loss between generated content and ground truth labels to measure captioning performance. A better metric is BLEU@4 which is commonly used to measure captioning performance (Li et al., 2023, 2020; Wang et al., 2022a,b).

In-context learning improvements:

- We can use PromptCap captions encoding distance as the feature used for FAISS indexing. In this report, we only used Oscar+ or BLIP2 caption encoding distances.

- We can use ensembling for in-context prompting. Prophet used 5-fold ensembling to prompt Text-Davinci-002 with the same testing question 5 times with different ordering of in-context examples. Majority voting is used to decide on the final answer. VQA Score improved by 2.2 from 1-fold to 5-fold with 20 shots. Given more funding, we can try a similar approach.

- We found that Text-Davinci-003 showed the best reasoning ability despite being a legacy model. However, it is 10 times the price of GPT-3.5-Turbo. Thus, we cannot conduct experiments with more shots with it. Given more funding, we would use Text-Davinci-003 or GPT-4 instead of GPT-3.5-Turbo. Alternatively, we can use GPT-3.5-Instruct which will be released soon. We found that LLAMA2's dialogue fine-tuned model has limited in-context generation ability (Section 5.4.4.) Since GPT-3.5-Turbo is optimized for chat, it might have similar issues. GPT-3.5-Instruct is instruction fine-tuned and might have better in-context ability.

6.2 Reflection on the OKVQA dataset

This report evaluates model performance on the OKVQA dataset, which claims that image content is not sufficient to answer the questions. However, not all questions in OKVQA require outside knowledge. Some require text or object recognition with a sufficiently strong visual model. Some only need common sense that a decently sized general-purpose language would have (Figure 5.3). The best performance from this report without document is 58.91% which suggests most questions do not require external knowledge. Thus, we see state-of-the-art performance on OKVQA by very large visual-language models with powerful visual encoders and language models that have been trained on internet scale data. We also see in-context prompting methods under-perform state-of-the-art modals by a sizeable margin, due to the lack of good visual representation.

It is difficult to draw the line at when outside knowledge is required. Most people need to google to find out when McDonald's is founded, but do we really need to google to know that a frowning face means that a person is sad, or that broccoli and tomato are both vegetables? Intuitively, the more knowledgeable a language model is about human culture, history, geography and science, the less it requires outside knowledge. At the same time, outside knowledge understanding also requires strong general understanding and summarising ability. Thus, it seems that external knowledge understanding and implicit knowledge generation ability are correlated. Regardless of the source of knowledge, larger models with strong general language ability seem advantageous over smaller models for KBVQA. This is a daunting prospect for researchers like us with limited compute resources.

When using the OKVQA dataset, we need to consider what we are trying to achieve. Are we trying to get state-of-the-art performance on this particular dataset? If so, specific engineering is required to design a system that balances using external documents vs using common sense. Are we trying to create a retrieval system that can find relevant documents from a database? If so, OKVQA is a limited dataset to measure retrieval performance since many questions do not require external knowledge. Are we trying to create a small but powerful visual-language model that can achieve similar performance to models that are much larger? If so, a more general VQA dataset might be a better choice. In this project, OKVQA is a good dataset that allows us to explore all three of these questions.

6.3 Conclusion

In this report, we started with literature review of state-of-the-art models in VQA. Specifically, we found that there are three commonly used types of methods for KBVQA: large visual-language models, external retrieval-based systems and in-context learning. Each method used on its own has limitations, thus we design a system in two parts that combine all three methods.

The first part of the system is a fine-tuned visual-language model with external document retrieval. We adopt the method in RA-VQA (Lin and Byrne, 2022) with the addition of a visual encoder for better visual understanding. We tested various visual encoders and found

InstructBLIP to be the best performing. Our InstructBLIP-RAVQA model achieves 62.83% OKVQA performance which is better than all other retrieval based methods.

The second part of the system uses in-context learning to balance explicit knowledge and implicit knowledge. We use fine-tuned InstructBLIP-RAVQA to generate document-wise answer candidates which are given to a different frozen LLM for the final answer. The frozen LLM is expected to select an answer from candidates, or generate an alternative answer from its implicit knowledge when none of the answers are appropriate. Our in-context method achieves 61.69% which outperforms all other models using in-context prompting for OKVQA.

References

Aghajanyan, A., Yu, L., Conneau, A., Hsu, W.-N., Hambardzumyan, K., Zhang, S., Roller, S., Goyal, N., Levy, O., and Zettlemoyer, L. (2023). Scaling laws for generative mixed-modal language models.

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., Ring, R., Rutherford, E., Cabi, S., Han, T., Gong, Z., Samangooei, S., Monteiro, M., Menick, J. L., Borgeaud, S., Brock, A., Nematzadeh, A., Sharifzadeh, S., Bińkowski, M. a., Barreira, R., Vinyals, O., Zisserman, A., and Simonyan, K. (2022). Flamingo: a visual language model for few-shot learning. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 23716–23736. Curran Associates, Inc.

Bao, H., Wang, W., Dong, L., Liu, Q., Mohammed, O. K., Aggarwal, K., Som, S., and Wei, F. (2022). Vlmo: Unified vision-language pre-training with mixture-of-modality-experts.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-end object detection with transformers.

Changpinyo, S., Sharma, P., Ding, N., and Soricut, R. (2021). Conceptual 12M: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *CVPR*.

Chen, X., Djolonga, J., Padlewski, P., Mustafa, B., Changpinyo, S., Wu, J., Ruiz, C. R., Goodman, S., Wang, X., Tay, Y., Shakeri, S., Dehghani, M., Salz, D., Lucic, M., Tschannen, M., Nagrani, A., Hu, H., Joshi, M., Pang, B., Montgomery, C., Pietrzyk, P., Ritter, M., Piergiovanni, A., Minderer, M., Pavetic, F., Waters, A., Li, G., Alabdulmohsin, I., Beyer, L., Amelot, J., Lee, K., Steiner, A. P., Li, Y., Keysers, D., Arnab, A., Xu, Y., Rong, K., Kolesnikov, A., Seyedhosseini, M., Angelova, A., Zhai, X., Houlsby, N., and Soricut, R. (2023a). Pali-x: On scaling up a multilingual vision and language model.

Chen, X., Wang, X., Changpinyo, S., Piergiovanni, A., Padlewski, P., Salz, D., Goodman, S., Grycner, A., Mustafa, B., Beyer, L., Kolesnikov, A., Puigcerver, J., Ding, N., Rong, K., Akbari, H., Mishra, G., Xue, L., Thapliyal, A., Bradbury, J., Kuo, W., Seyedhosseini, M., Jia, C., Ayan, B. K., Riquelme, C., Steiner, A., Angelova, A., Zhai, X., Houlsby, N., and Soricut, R. (2023b). Pali: A jointly-scaled multilingual language-image model.

Chen, Y.-C., Li, L., Yu, L., El Kholy, A., Ahmed, F., Gan, Z., Cheng, Y., and Liu, J. (2020). Uniter: Universal image-text representation learning. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX*, page 104–120, Berlin, Heidelberg. Springer-Verlag.

Chiang, W.-L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J. E., Stoica, I., and Xing, E. P. (2023). Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Cho, J., Lei, J., Tan, H., and Bansal, M. (2021). Unifying vision-and-language tasks via text generation. In *ICML*.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. (2022). Palm: Scaling language modeling with pathways.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H., Petrov, S., Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., and Wei, J. (2022). Scaling instruction-finetuned language models.

Dai, W., Li, J., Li, D., Tiong, A. M. H., Zhao, J., Wang, W., Li, B., Fung, P., and Hoi, S. (2023). Instructblip: Towards general-purpose vision-language models with instruction tuning.

Danilevsky, M., Qian, K., Aharonov, R., Katsis, Y., Kawas, B., and Sen, P. (2020). A survey of the state of explainable AI for natural language processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 447–459, Suzhou, China. Association for Computational Linguistics.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale.

Driess, D., Xia, F., Sajjadi, M. S. M., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q., Yu, T., Huang, W., Chebotar, Y., Sermanet, P., Duckworth, D., Levine, S., Vanhoucke, V., Hausman, K., Toussaint, M., Greff, K., Zeng, A., Mordatch, I., and Florence, P. (2023). Palm-e: An embodied multimodal language model. In *arXiv preprint arXiv:2303.03378*.

Eichenberg, C., Black, S., Weinbach, S., Parcalabescu, L., and Frank, A. (2022). Magma – multimodal augmentation of generative models through adapter-based finetuning.

Gan, Z., Chen, Y.-C., Li, L., Zhu, C., Cheng, Y., and Liu, J. (2020). Large-scale adversarial training for vision-and-language representation learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Gao, F., Ping, Q., Thattai, G., Reganti, A. N., Wu, Y. N., and Natarajan, P. (2022). A thousand words are worth more than a picture: Natural language-centric outside-knowledge visual question answering. *CoRR*, abs/2201.05299.

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

Gui, L., Wang, B., Huang, Q., Hauptmann, A., Bisk, Y., and Gao, J. (2022). KAT: A knowledge augmented transformer for vision-and-language. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 956–968, Seattle, United States. Association for Computational Linguistics.

Guo, D., Xu, C., and Tao, D. (2023). Bilinear graph networks for visual question answering. *IEEE Transactions on Neural Networks and Learning Systems*, 34(2):1023–1034.

Hu, E. J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2022). LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Hu, Y., Hua, H., Yang, Z., Shi, W., Smith, N. A., and Luo, J. (2023). Promptcap: Prompt-guided task-aware image captioning.

Hudson, D. A. and Manning, C. D. (2019). 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 (CVPR)*.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12).

Jia, C., Yang, Y., Xia, Y., Chen, Y.-T., Parekh, Z., Pham, H., Le, Q. V., Sung, Y., Li, Z., and Duerig, T. (2021). Scaling up visual and vision-language representation learning with noisy text supervision.

Jiang, H., Misra, I., Rohrbach, M., Learned-Miller, E., and Chen, X. (2020). In defense of grid features for visual question answering. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10264–10273.

Johnson, J., Douze, M., and Jegou, H. (2021). Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(03):535–547.

Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Zitnick, C. L., and Girshick, R. (2017). Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. (2020). Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Li, J., Li, D., Savarese, S., and Hoi, S. (2023). Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models.

Li, L., Gan, Z., Cheng, Y., and Liu, J. (2019). Relation-aware graph attention network for visual question answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Li, W., Gao, C., Niu, G., Xiao, X., Liu, H., Liu, J., Wu, H., and Wang, H. (2021). UNIMO: Towards unified-modal understanding and generation via cross-modal contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2592–2607, Online. Association for Computational Linguistics.

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. (2020). Oscar: Object- semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, page 121–137.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing.

Lin, W. and Byrne, B. (2022). Retrieval augmented visual question answering with outside knowledge. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11238–11254, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Lin, Y., Xie, Y., Chen, D., Xu, Y., Zhu, C., and Yuan, L. (2022). Revive: Regional visual representation matters in knowledge-based visual question answering. *arXiv preprint arXiv:2206.01201*.

Liu, F., Emerson, G., and Collier, N. (2023a). Visual spatial reasoning.

Liu, H. and Singh, P. (2004). Conceptnet — a practical commonsense reasoning tool-kit. *BT Technology Journal*, 22:211–226.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. (2023b). Lost in the middle: How language models use long contexts.

Lu, J., Batra, D., Parikh, D., and Lee, S. (2019). *ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks*, pages 13–23. Curran Associates Inc., Red Hook, NY, USA.

Luo, M., Zeng, Y., Banerjee, P., and Baral, C. (2021). Weakly-supervised visual-retriever-reader for knowledge-based question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6417–6431, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Malinowski, M. and Fritz, M. (2014). A multi-world approach to question answering about real-world scenes based on uncertain input. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 27*, pages 1682–1690. Curran Associates, Inc.

Marino, K., Chen, X., Parikh, D., Gupta, A., and Rohrbach, M. (2020a). Krisp: Integrating implicit and symbolic knowledge for open-domain knowledge-based vqa.

Marino, K., Chen, X., Parikh, D., Gupta, A., and Rohrbach, M. (2020b). Krisp: Integrating implicit and symbolic knowledge for open-domain knowledge-based vqa.

Marino, K., Rastegari, M., Farhadi, A., and Mottaghi, R. (2019). Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mishra, A., Shekhar, S., Singh, A. K., and Chakraborty, A. (2019). Ocr-vqa: Visual question answering by reading text in images. In *ICDAR*.

Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., and Wu, X. (2023). Unifying large language models and knowledge graphs: A roadmap.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. (2021). Learning transferable visual models from natural language supervision.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Schwenk, D., Khandelwal, A., Clark, C., Marino, K., and Mottaghi, R. (2022). A-okvqa: A benchmark for visual question answering using world knowledge. *arXiv*.

Shao, Z., Yu, Z., Wang, M., and Yu, J. (2023). Prompting large language models with answer heuristics for knowledge-based visual question answering. In *Computer Vision and Pattern Recognition (CVPR)*, pages 14974–14983.

Sharma, P., Ding, N., Goodman, S., and Soricut, R. (2018). Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia. Association for Computational Linguistics.

Singh, A., Natarajan, V., Shah, M., Jiang, Y., Chen, X., Batra, D., Parikh, D., and Rohrbach, M. (2019). Towards vqa models that can read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Smith, S., Patwary, M., Norick, B., LeGresley, P., Rajbhandari, S., Casper, J., Liu, Z., Prabhumoye, S., Zerveas, G., Korthikanti, V., Zhang, E., Child, R., Aminabadi, R. Y., Bernauer, J., Song, X., Shoeybi, M., He, Y., Houston, M., Tiwary, S., and Catanzaro, B. (2022). Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model.

Tapaswi, M., Zhu, Y., Stiefelhagen, R., Torralba, A., Urtasun, R., and Fidler, S. (2016). Movieqa: Understanding stories in movies through question-answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Thoppilan, R., Freitas, D. D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., Li, Y., Lee, H., Zheng, H. S., Ghafouri, A., Menegali, M., Huang, Y., Krikun, M., Lepikhin, D., Qin, J., Chen, D., Xu, Y., Chen, Z., Roberts, A., Bosma, M., Zhao, V., Zhou, Y., Chang, C.-C., Krivokon, I., Rusch, W., Pickett, M., Srinivasan, P., Man, L., Meier-Hellstern, K., Morris, M. R., Doshi, T., Santos, R. D., Duke, T., Soraker, J., Zevenbergen, B., Prabhakaran, V., Diaz, M., Hutchinson, B., Olson, K., Molina, A., Hoffman-John, E., Lee, J., Aroyo, L., Rajakumar, R., Butryna, A., Lamm, M., Kuzmina, V., Fenton, J., Cohen, A., Bernstein, R., Kurzweil, R., Aguera-Arcas, B., Cui, C., Croak, M., Chi, E., and Le, Q. (2022). Lamda: Language models for dialog applications.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. (2023). Llama 2: Open foundation and fine-tuned chat models.

Tsimpoukelli, M., Menick, J., Cabi, S., Eslami, S. M. A., Vinyals, O., and Hill, F. (2021). Multimodal few-shot learning with frozen language models.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Vrandečić, D. and Krötzsch, M. (2014). Wikidata: A free collaborative knowledge base. *Communications of the ACM*, 57:78–85.

Wang, J., Yang, Z., Hu, X., Li, L., Lin, K., Gan, Z., Liu, Z., Liu, C., and Wang, L. (2022a). Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*.

Wang, P., Wu, Q., Shen, C., Dick, A., and van den Hengel, A. (2017). Explicit knowledge-based reasoning for visual question answering. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 1290–1296.

Wang, P., Wu, Q., Shen, C., Dick, A., and van den Hengel, A. (2018). Fvqa: Fact-based visual question answering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(10):2413–2427.

Wang, P., Yang, A., Men, R., Lin, J., Bai, S., Li, Z., Ma, J., Zhou, C., Zhou, J., and Yang, H. (2022b). Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. *CoRR*, abs/2202.03052.

Wang, W., Bao, H., Dong, L., Bjorck, J., Peng, Z., Liu, Q., Aggarwal, K., Mohammed, O. K., Singhal, S., Som, S., and Wei, F. (2022c). Image as a foreign language: Beit pretraining for all vision and vision-language tasks.

Wang, Z., Yu, J., Yu, A. W., Dai, Z., Tsvetkov, Y., and Cao, Y. (2022d). Simvlm: Simple visual language model pretraining with weak supervision.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. (2022). Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

Wei, J., Wei, J., Tay, Y., Tran, D., Webson, A., Lu, Y., Chen, X., Liu, H., Huang, D., Zhou, D., and Ma, T. (2023). Larger language models do in-context learning differently.

Wu, J., Lu, J., Sabharwal, A., and Mottaghi, R. (2021). Multi-modal answer validation for knowledge-based VQA. *CoRR*, abs/2103.12248.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., and Luo, P. (2021). Segformer: Simple and efficient design for semantic segmentation with transformers. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 12077–12090. Curran Associates, Inc.

Yang, Z., Gan, Z., Wang, J., Hu, X., Lu, Y., Liu, Z., and Wang, L. (2022). An empirical study of gpt-3 for few-shot knowledge-based vqa. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(3):3081–3089.

Young, P., Lai, A., Hodosh, M., and Hockenmaier, J. (2014). From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Yu, J., Wang, Z., Vasudevan, V., Yeung, L., Seyedhosseini, M., and Wu, Y. (2022). Coca: Contrastive captioners are image-text foundation models.

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

Yu, Z., Yu, J., Xiang, C., Fan, J., and Tao, D. (2018). Beyond bilinear: Generalized multimodal factorized high-order pooling for visual question answering. *IEEE Transactions on Neural Networks and Learning Systems*, 29(12):5947–5959.

Zhai, X., Kolesnikov, A., Houlsby, N., and Beyer, L. (2022a). Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12104–12113.

Zhai, X., Kolesnikov, A., Houlsby, N., and Beyer, L. (2022b). Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12104–12113.

Zhang, P., Li, X., Hu, X., Yang, J., Zhang, L., Wang, L., Choi, Y., and Gao, J. (2021). Vinvl: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5579–5588.

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

Appendix A

Alternative DPR Loss

In Section 5.4.1, we discussed an issue with document-wise answer candidates; they are frequently identical for different documents. Since the answer generator only sees one document at a time, and gradient update only happens after a batch of questions, the same model weights generate multiple identical answers for different documents. One hypothesis to explain this phenomenon is that the model is trained to ignores all documents and uses its implicit knowledge instead to answer certain questions. This is not an issue when it comes to the OKVQA fine-tuned model performance. However, since the predictions are used as answer candidates for few-shot prompting, the lack of prediction diversity means very few distinct answer candidate can be used. This betrays the purpose of using document retrieval as external knowledge from documents is not utilized.

In this chapter, two alternative training losses are tested in an attempt to address this issue. Document retrieval loss is changed to punish retrieval of documents that get ignored by answer generator. Although a small improvement is observed in VQA Score by fine-tuned InstructBLIP-RAVQA models, in-context performance degrades. More variety in validation answer candidates is observed, but not in training answer candidates. Thus, this alternative training regime is unsuccessful. The methods and results are presented for reference.

A.1 Method

In Section 3.1.3, we described how RA-VQA (Lin and Byrne, 2022) uses two parts in the training loss to ensure the quality of document retrieved and answer generated. The term $\sum_{k=1}^{K} \log p_{\phi}(s_k^*|x, z_k)$ is used to update the weights of the answer generator only, while the terms $\sum_{k \in P^+(x,S)} \log p_{\theta}(z_k|x) - \sum_{k \in P^-(x,S)} \log p_{\theta}(z_k|x)$ only update the weights of the document retriever. This method focuses on the quality of document and the quality of answers generated separately, but neglects how the generator makes use of a document. The loss terms do not directly connect the retriever and generator. The two models are connected indirectly through on-the-fly predictions and pseudo relevance labels.

There is no trivial modification that can force the auto-regressive answer generator to generate answers from documents and prevent documents from being ignored. Thus, an alternative is explored where the retriever works with the generator's tendency to ignore certain documents. The retriever is discouraged from retrieving documents that are ignored by the answer generator. This could in turn let the answer generator produce answers that come from documents. Two additional loss conditions are added.

$$\{k \,:\, y_k = s^* \wedge H(z_k, S) = 0\} \tag{A.1}$$

$$\{k \,:\, y_k \neq s_k^* \wedge H(z_k, S) = 1\} \tag{A.2}$$

The original P^- term only includes one condition $P^-(x,S) = \{k \,:\, y_k \neq s^* \wedge H(z_k,S) = 0\}$. The intuition behind adding these two additional terms is that the original P^- does not include the case where a correct answer is generated, but the document doesn't contain any answer. Or the case where a wrong answer is generated, but the document contains the right answer. So the instances where the answer generator ignores the document and generates an answer that's irrelevant to the document are allowed to happen without punishment. By punishing documents that get ignored, the retriever can ideally only retrieve documents that do not get ignored by the answer generator. This might cause degrade in PRRecall and VQA Score as the documents that actually contain the right answer might be not retrieved. However, if the generator would ignore the correct document anyways with the original loss condition (Table 5.12), it does not matter that wrong documents are retrieved with the new loss. The new loss term might improve answer candidate diversity, which is one of the main issues with in-context learning. More than half of in-context examples contain only 1 or 2 distinct answer candidates (Table 5.1).

Thus new document loss can include one or two additional terms:

$$P^-_{add1}(x,S) = \left\{ k \,:\, \begin{array}{c} (y_k \neq s^* \wedge H(z_k,S) = 0) \\ \vee \left(y_k \neq s_k^* \wedge H(z_k,S) = 1\right) \end{array} \right\} \tag{A.3}$$

$$P^-_{add2}(x,S) = \left\{ k \,:\, \begin{array}{c} (y_k \neq s^* \wedge H(z_k,S) = 0) \\ \vee (y_k = s^* \wedge H(z_k,S) = 0) \\ \vee \left(y_k \neq s_k^* \wedge H(z_k,S) = 1\right) \end{array} \right\} \tag{A.4}$$

The first option P^-_{add1} additionally punish the instances where a correct document is retrieved, but ignored. The second option P^-_{add2} is more strict where any instances that is not a correct document with corresponding answer from the document in punished. Experiments are conducted with both options with InstructBLIP-RAVQA using the same framework as in Section 5.3.

A.2 Results

Table A.1 Two models are trained with additional loss conditions according to Eqn A.3, A.4 with $K_{train} = 5$. They achieve slightly better performance with $K_{test} = 5$ than InstructBLIP-RAVQA with the original loss condition from Lin and Byrne (2022).

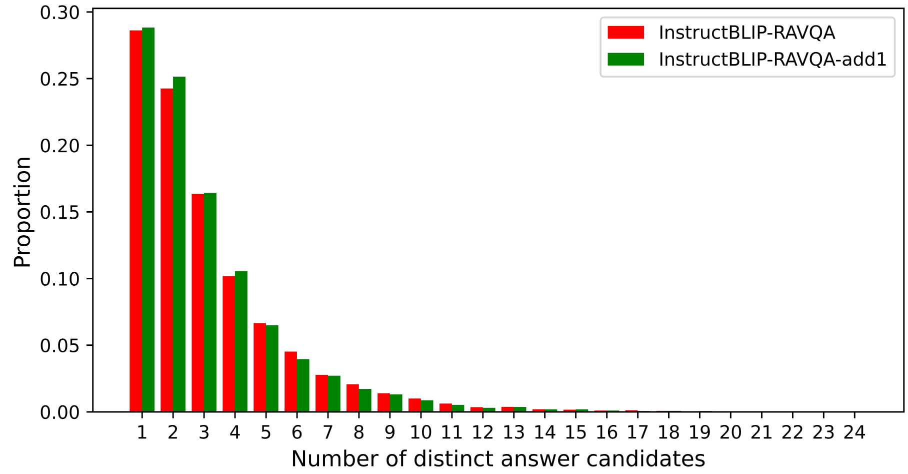
Model	PRRecall		VQA Score	
	$K_{test} = 5$	$K_{test} = 50$	$K_{test} = 5$	$K_{test} = 50$
InstructBLIP-RAVQA	84.78	97.05	62.51	62.83
InstructBLIP-RAVQA-add1	82.80	96.63	62.83	62.41
InstructBLIP-RAVQA-add2	83.83	96.85	62.88	62.94

Table A.1 shows that alternative loss functions achieve better OKVQA performance than the original loss by 0.3 points even though improved OKVQA performance is not the main goal of using alternative loss. P^{-}_{add2} achieves improved performance when K_{test} increases to 50, similar to the baseline InstructBLIP-RAVQA. P^{-}_{add1} has worse performance in VQA Score and PRRecall when $K_{test} = 50$. Despite this, P^{-}_{add1} has better answer candidates than P^{-}_{add2}, which improves from InstructBLIP-RAVQA. As shown in Table A.2, both P^{-}_{add1} and P^{-}_{add2} have better answer candidates in terms of ACRecall and Hit Rate than the baseline. This achieves one of the objects of using alternative loss, which is to improve answer candidate quality. Since P^{-}_{add1} has the best answer candidates for OKVQA's validation questions, we generate answer candidates for training questions using frozen P^{-}_{add1} model to verify if it achieves the objective of getting more distinct answer candidates.

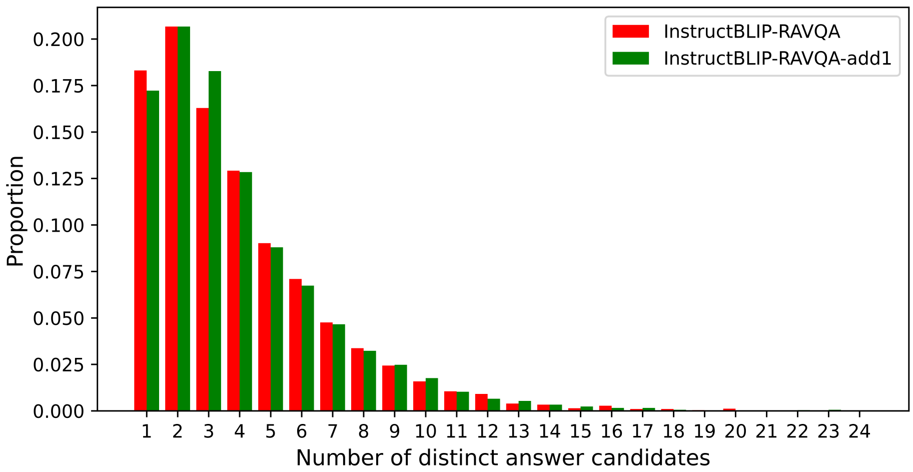
Table A.2 Alternative loss models' answer candidate quality is evaluated for OKVQA's validation questions when $K_{test} = 50$

Model	$K_{AC} = 5$		$K_{AC} = max$	
	ACRecall	Hit Rate	ACRecall	Hit Rate
InstructBLIP-RAVQA	87.67	83.73	89.71	85.78
InstructBLIP-RAVQA-add1	88.80	84.75	90.31	86.34
InstructBLIP-RAVQA-add2	88.09	83.95	90.19	86.10

Figure A.1 shows the distribution of number of distinct answer candidates for InstructBLIP-RAVQA-add1. We can see in validation questions, the goal of more distinct answer candidates



(a) Training questions



(b) Validation questions

Fig. A.1 Distribution of number of distinct answer candidates for OKVQA training and validation questions. InstructBLIP-RAVQA-add1 is used with $K_{test} = 50$ in comparison with InstructBLIP-RAVQA baseline

is achieved to a limited extent. The number of questions with 1 distinct answer candidates reduces. That with 3 distinct answer candidates increases. However, for training questions, answer candidates become more concentrated in low number of answer candidates, which is the opposite of the desired behavior. In earlier analysis, we note that having very few answer candidates in training questions is undesirable since training questions make up in-context examples. The model learns to only choose the final answer from the first or second answer candidate, rather than use its implicit knowledge when necessary. Thus, InstructBLIP-RAVQA-add1's answer candidate distribution diverge even more between training and validation questions, compared to baseline. We can expect the in-context performance to be worse. Indeed, Table A.3 shows worsened in-context performance. Thus, we can conclude that alternative loss functions do not help obtain more diverse answer candidates in order to boost in-context performamce.

There are several reasons why the alternative loss function does not improve answer diversity. Firstly, the source of lack of answer diversity is the answer generator selectively ignoring certain documents. However, the alternative loss function only updates the weights of the DPR, in hopes of retrieving documents that do not get ignored, rather than changing the answer generator directly. Secondly, training DPR requires hyperparameter tuning which is not done in this report. Hyperparameters from RA-VQA are used directly. We observe that throughout InstructBLIP-RAVQA joint training, the PRRecall peaks at epoch 1, and decreases gradually afterwards. The OKVQA accuracy increases throughout training. Thus, an inappropriate learning rate might have been used for DPR. The learning rate is not changed for the alternative loss training regime. However, additional loss terms make loss bigger. It might be better to reduce the learning rate for DPR to accommodate larger loss.

Table A.3 Comparison of in-context performance for additional loss training regime and baseline model. In-context models have prompt head, confidence scores, $K_{AC} = 5$, $N = 5$, in-context similarity based on combined question + Oscar caption + BLIP2 image embedding.

Model	VQA Score
InstructBLIP-RAVQA	59.73
InstructBLIP-RAVQA-add1	58.89

Appendix B

Additional Experiments with ViT Prefix

Table B.1 ViT prefix performance on captioning task with different MLP layers and prompt types. Short prompt is "A photo of"

LLM	Visual Prefix	Number of MLP layers	Prompt Type	Loss
Flan-T5-Large	ViT	2	No prompt	2.986
		1	No prompt	3.017
			Short prompt	2.911

We conducted an additional experiment to investigate the effect of adding a short prompt "a photo of" to Flan-T5-Large for pre-training mapping network on Conceptual Captions. As expected, captioning loss is lower when a prompt is added. This is due to ’Flan’ models being instruction fine-tuned in the pre-training process. Additionally, we find that using 2 MLP layers result in lower captioning loss. Increased capacity in the MLP likely reshaped the visual encoding in a way that was better understood by the language model. We do not use 2 MLP layers in experiments using BLIP2 or InstructBLIP even though it might result in better performance. This is because the original BLIP2 and InstructBLIP papers only use one MLP layer. We follow the architectures proposed in the original papers so that our result is comparable to theirs.

In the main report, we do not focus on T5-Large since it under performs Flan-T5-Large consistently. In Table B.2, we see that indeed T5-Large achieves 2% lower VQA Score than Flan-T5-Large for all experiments. We see that by adding ViT prefix, FrDPR performance increased for both LLMs. The increase in T5-Large is larger than that in Flan-T5-Large (1.6% vs 0.13%). This is reasonable since it becomes more difficult to get incremental gains in performance as the baseline performance gets better.

Table B.2 FrDPR with ViT prefix is trained end-to-end with various prefix length and number of layers for mapping framework MLP. All experiments train MLP from scratch, without pre-training on captioning task.

Model	Prefix Length	Number of MLP layers	T5-Large	Flan-T5-Large
FrDPR	NA	NA	51.63	54.23
FrDPR-ViT	10	2	52.19	54.36
	32	2	51.53	54.11
		1	52.09	54.22

We expected when prefix length increases, the visual prefix contains more visual information which will result in better performance. However, as prefix length increases from 10 to 32, performance do not increase. This is likely due to the lack of pre-training. The mapping network is trained from scratch with the OKVQA task. From the main report (Table 5.6), we see that pre-training is an essential step to let the LLM understand visual prefix prior to OKVQA fine-tuning. Thus, with pre-training, we might be able to observe improved performance with longer prefix.

Appendix C

InstructBLIP is not capable of generating question-aware captions

Table C.1 Test combinations of inputs to InstructBLIP's Q-Former and LLM. Text are encoded before passing into Q-Former or LLM. Four input types are tested with question input either into the Q-Former or LLM, or both.

	Type 1	Type 2	Type 3	Type 4
Input to Q-Former	A photo of	<question >	<question>	<question> Provide a caption for the image that will help answer the question
Input to LLM	A photo of	Provide a caption for the image that will help answer the question	Taking the previous question into account, describe this image	<question> Provide a caption for the image that will help answer the question

Table C.2 Response of InstructBLIP to 4 types of input configurations.

Question	Type 1	Type 2	Type 3	Type 4
What sport can you use this for?	a motorcycle parked on a street near a building	motorcycle	motorcycle	racing
Name the type of plant this is	a bathroom with a large wall of plants	plant	bathroom	fern
What toy is this?	a man with a teddy bear on his back	dog on man	dog	back

Initially, we were planning on using InstructBLIP to generate instruction-aware captions. Since InstructQFormer can take instructions to generate instruction-aware encoding, it is not far fetched to assume that we can phrase the instruction such that it requires the model to generate a caption that include details about a question. However, we tried putting instruction such as "Provide a caption for the image that will help answer the question" as input to the QFormer, or to the LLM. We found that as long as the question appear, either in QFormer, or the LLM, the model always answers the question rather than provide a caption. Thus, we cannot use InstructBLIP to generate question-aware captions.

This can be explained by the training data used for InstructBLIP, which mainly consists of VQA and captioning tasks. There are also some reading comprehension, conversation and classification tasks (Dai et al., 2023). A list of instruction templates used during training is given in the paper. Those instructions do not contain captioning task using VQA datasets. Thus, InstructBLIP have never been trained to do instruction-aware captioning, which explains why it can either only caption or answer the question with the inputs given in Table C.2.

As an alternative, we use PromptCap (Hu et al., 2023), which is a model that can actually perform question-aware captioning. Further details are given in Section 3.3.2.

Appendix D

Model Hyperparameters

Table D.1 Hyperparameters for Q-only and question + text-based vision OKVQA fine-tuning

LLM	T5-Large	Flan-T5-Large	Flan-T5-XL
Batch size		2	
Gradient accumulation steps		16	
Start learning rate		6e-5	
Learning rate scheduler		Linear decay at 2e-8/step	
Precision		fp32	
Epoch		6	
LoRA r	NA	NA	8
LoRA α	NA	NA	32
LoRA dropout	NA	NA	0.1

Table D.2 Hyperparameters for pre-training on Conceptual Captions

LLM	Flan-T5-Large	Flan-T5-XL
Batch size	64	
Gradient accumulation steps	2	
Start learning rate	3e-4	
Learning rate scheduler	Constant	
Precision	fp32	
Epoch	10	
LoRA r	NA	8
LoRA α	NA	32
LoRA dropout	NA	0.1

Table D.3 Hyperparameters for fine-tuning visual-language models on OKVQA without documents.

LLM	Flan-T5-Large	Flan-T5-XL
Batch size	2	
Gradient accumulation steps	16	
Start learning rate	1e-4	
Learning rate scheduler	Linear decay at 2e-8/step	
Precision	fp32	
Epoch	6	
LoRA r	NA	8
LoRA α	NA	32
LoRA dropout	NA	0.1

Table D.4 Hyperparameters for fine-tuning InstructBLIP-Flan-T5-XL-FrDPR on OKVQA.

LLM	Flan-T5-XL
Batch size	1
Gradient accumulation steps	32
Retriever learning rate	1e-5
LLM learning rate	6e-5
Precision	bf16
Epoch	4
LoRA r	8
LoRA α	32
LoRA dropout	0.1