5-2024

Using pre-trained models for vision-language understanding tasks

Rui CAO
Singapore Management University, ruicao.2020@phdcs.smu.edu.sg

Citation

CAO, Rui. Using pre-trained models for vision-language understanding tasks. (2024). 1-217.
Available at: https://ink.library.smu.edu.sg/etd_coll/595

Using Pre-trained Models for Vision-Language Understanding Tasks

RUI CAO

SINGAPORE MANAGEMENT UNIVERSITY
2024

Using Pre-trained Models for Vision-Language Understanding Tasks

by

Rui Cao

Submitted to School of Computing and Information Systems
in partial fulfillment of the requirements for
the Degree of Doctor of Philosophy in Computer Science

Proposal Committee:

Jing Jiang (Supervisor / Chair)
Professor of Computer Science
Singapore Management University

Chong-Wah Ngo
Professor of Computer Science
Singapore Management University

Wei Gao
Assistant Professor of Computer Science
Singapore Management University

Roy Ka-Wei Lee
Assistant Professor of Computer Science
Singapore University of Technology and Design

Singapore Management University
2024

I hereby declare that this PhD thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in this thesis.

This PhD thesis has also not been submitted for any degree in any university previously.

Cao Rui

Rui Cao

4 May 2024

Using Pre-trained Models for Vision-Language Understanding Tasks

Rui Cao

In recent years, remarkable progress has been made in *Artificial Intelligence* (AI), with an increasing focus on integrating AI systems into people's daily lives. In the context of our diverse world, research attention has shifted towards applying AI to multimodal understanding tasks. This thesis specifically addresses two key modalities, namely, vision and language, and explores *Vision-Language Understanding* (VLU).

In the past, addressing VLU tasks involved training distinct models from scratch using task-specific data. However, limited by the amount of training data, models may easily overfit the training data and fail to generalize. A recent breakthrough is the development of Pre-trained Models (PTMs), which are trained on extensive datasets to acquire universal representations. Leveraging these PTMs for VLU tasks has become a prevalent approach.

The use of PTMs for VLU tasks can be divided into two paradigms: (1) fine-tuning PTMs with downstream task data, and (2) zero-shot transfer or few-shot learning based on frozen PTMs. However, existing methods under these two paradigms suffer from a few limitations: direct fine-tuning of PTMs may overlook the unique characteristics of the downstream tasks; the zero-shot and few-shot performance of PTMs on some tasks may be poor; and complex VLU tasks may require multiple reasoning skills that a single PTM may not possess.

In the thesis, we aim to address the limitations above by optimizing the utilization of PTMs for VLU tasks. Our work can be organized based on whether we leverage fine-tuning or zero-shot / few-shot learning, and whether we adopt a single PTM or a composition of PTMs. When tuning a single PTM, we explore how to incorporate task-specific components to better cater to downstream tasks (Tuning-Single). For VLU tasks where frozen PTMs are not ideal solutions due to poor performance, we investigate using a single frozen PTM to facilitate sub-steps in these tasks (Frozen-

Single). On the other hand, we also study how to compose a set of tuned PTMs, each capable of a reasoning skill, to improve the performance on these tasks in the low-resource setting (Tuning-Composition). As VLU tasks may involve multiple skills and multiple reasoning steps, we consider a composition of frozen PTMs and assign reasoning tasks to proper frozen PTMs without requiring any adaptation (Frozen-Composition).

Specifically, in this thesis, we narrow down our scope to two VLU tasks, *Hateful Meme Detection* (HMD) and *Visual Question Answering* (VQA). HMD classifies a given multimodal meme as either hateful or not hateful, while VQA aims to answer questions related to a given image. The decision to focus on these two tasks stems from their importance in real-world applications. Furthermore, both tasks present non-trivial challenges that demand innovative solution approaches.

For the HMD task, most existing work has primarily focused on direct fine-tuning of PTMs, treating HMD as a general multimodal classification task and overlooking its unique characteristics. We address the limitation by integrating task-specific components with PTMs and tuning them end-to-end. We proposed **DisMultiHate**, which is based on a PTM but learns to disentangle representations of hate speech-related target entities in memes to enhance hateful content classification. Additionally, HMD often requires external background knowledge for meme comprehension, yet there are no dedicated knowledge bases constructed for this purpose. In light of this, we explore leveraging knowledge in *Pre-trained Language Models* (PT-LMs). We propose **PromptHate**, which prompts PT-LMs and utilizes their implicit knowledge for HMD. Since PT-LMs are inherently textual, PromptHate involves converting images into textual captions with a frozen pre-trained vision-language model (PT-VLM).

Though achieving good detection performance, PromptHate suffers from non-informative captions. Generic image descriptions may lack crucial details, such as race and gender information, vital for detecting hateful content. To address this, we proposed **Pro-Cap**, which leverages a frozen PT-VLM to complement PromptHate.

Specifically, we prompt a frozen PT-VLM with hateful content-related questions and use the answers as image captions (termed Pro-Cap), ensuring that the captions contain critical information for hateful content detection.

While these methods exhibit commendable performance, they heavily rely on extensive supervised learning, demanding large volumes of annotated data. This process is both costly and time-consuming. In response, we further introduce **Mod-HATE**, which harnesses the power of a composition of tuned PTMs, each of which possesses an essential reasoning capability for HMD. To the best of our knowledge, Mod-HATE represents a pioneering exploration of hateful meme detection tailored to the few-shot learning setting.

For VQA, we study it under the zero-shot transfer setting. Notably, previous zero-shot VQA models overlooked the explicit consideration of multi-step reasoning chains inherent in VQA. To address this oversight, We introduce a modularized zero-shot network that explicitly decomposes questions into sub-reasoning steps, converts sub-reasoning tasks to objectives suitable for PTMs, and assigns tasks to appropriate PTMs without adaptation.

Expanding our investigation, we delve into a specific VQA scenario known as knowledge-based VQA (K-VQA). In K-VQA, apart from an image, external knowledge is indispensable for answering the given questions. Recent approaches have utilized pre-trained large language models (LLMs) as both a knowledge source and a zero-shot QA model for K-VQA. However, these recent methods lack explicit demonstration of the knowledge needed to answer questions and thus lack interpretability. To rectify this deficiency, we propose **KGenVQA**, which first generates knowledge from a frozen LLM and subsequently leverages another frozen LLM for question answering with the incorporation of the generated knowledge.

Finally, we conclude the thesis with a summary of our contributions and a discussion of potential future directions regarding the application of PTMs to VLU.

Table of Contents

List of Figures

List of Tables

Acknowledgement

I am deeply grateful to my supervisor, Prof. Jing Jiang, for her unwavering support, invaluable guidance, and profound insights throughout this research journey. Her mentorship has been instrumental in shaping the direction of this thesis and my academic growth.

I extend my sincere appreciation to the members of my thesis committee, Prof. Wei Gao, Prof. Chong-Wah NGO, and Prof. Roy Ka-Wei Lee, for their constructive feedback and scholarly contributions, which have enriched the quality of this work.

I am indebted to Singapore Management University for providing the resources and conducive research environment essential for the completion of this thesis. I would like to thank the staff and faculty members for their assistance and support at every stage of this endeavor.

My heartfelt gratitude goes to my family for their unwavering love, encouragement, and understanding throughout this challenging yet rewarding journey. Their belief in me has been a constant source of strength and motivation. I am grateful to my friends and colleagues for their encouragement, stimulating discussions, and shared experiences, which have made this academic pursuit more fulfilling and enjoyable.

Lastly, I would like to express my deepest appreciation to all those individuals who, directly or indirectly, contributed to this thesis. Your support, whether moral, intellectual, or practical, has been invaluable and is deeply appreciated.

Chapter 1

Introduction

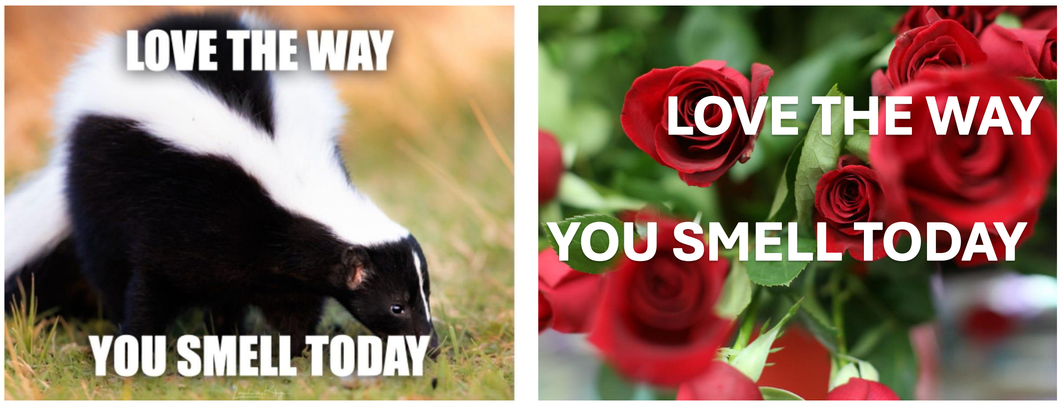
Disclaimer: *This thesis contains violence and discriminatory content that may be disturbing to some readers.*

1.1 Vision-Language Understanding

A modality refers to *the way in which something happens or is experienced* [12]. For instance, the connection between people and the physical world is established via *sensory modalities* such as vision and touch, which represent the primary channels of communication and sensation [12]. The complexity of human experience with the physical world underscores the need to comprehend multimodal signals from the outside world.

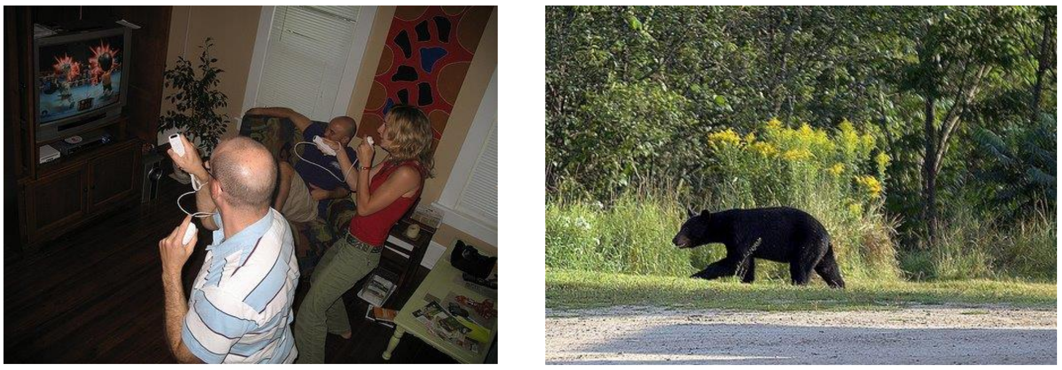
In recent years, Artificial Intelligence (AI) has witnessed significant progress, powering intelligent systems such as Chat-GPT [134] and enhancing human life through applications like face recognition and conversational assistants. However, to fully harness AI's potential in the context of our diverse world, which is inherently multimodal, it becomes imperative to explore models capable of multimodal understanding.

A prevalent example that demonstrates the importance of understanding multimodal data in our daily lives is the fusion of vision and language. There is a wide range of tasks and real-world applications that involve the integration of vision



(a) **(b)**

Figure 1.1: Examples of the hateful meme detection task. (a) is an example of *hateful* memes, while (b) is benign (i.e., *non-hateful*).



Question: *What is the man to the left of the woman holding?*
Answer: *Controller*
(a)

Question: *What California national park are these known to be seen?*
Answer: *Yosemite*
(b)

Figure 1.2: Examples of the visual question answering task.

and language modalities. Examples include the task of recommending products to consumers where the conversation of recommendation involves both images and texts (*Recommendations with Images and Texts* [148]), the task of providing a brief textual description of an image (*Image Captioning* [107]) and the task of searching for the most relevant video related to the semantics in a given sentence (*Video-Text Retrieval* [189]).

Because vision and language are two key modalities in the world we live in, in this thesis, we focus on the joint comprehension of these two modalities. We use the term *Vision-Language Understanding* (VLU) to refer to this type of comprehension task.

VLU poses significant challenges in real-world applications. Firstly, it requires not only the comprehension of individual modalities (vision and language) but also an understanding of their interactions. For example, *Multimodal Memes*[1], often comprising images with short texts, pose an interesting VLU setting when we consider the task of *Hateful Meme Detection* [85]. In hateful meme detection, meme texts (e.g., "love the way you smell today"), when considered individually, may seem harmless. However, when combined with different seemingly benign images (e.g., an image of a skunk or an image of roses), they can lead to completely different connotations, which make some to be considered hateful (e.g., Figure 1.1 (a)) while others are benign (e.g., Figure 1.1 (b))[2]. Another example of the challenge of understanding the interactions between vision and language in VLU tasks is in the context of *Visual Question Answering* [54] (**VQA**). A VQA system is expected to accurately answer questions posed about an image, as depicted in Figure 1.2. Usually with only unimodal information (either the image or the question), it is impossible to provide a correct answer. It is only when a VQA system is able to jointly understand both the visual and textual inputs and their relations can the system accurately answer a question. As we can see, both hateful meme detection and VQA highlight the importance of comprehending the joint interactions between vision and language modalities in VLU.

Secondly, VLU tasks are oftentimes inherently complex, requiring a diverse range of comprehension skills and often involving multiple reasoning steps. For instance, the task of hateful meme detection demands an understanding of hate speech definitions, visual metaphors in multimodal memes, and the underlying meanings of memes. Similarly, VQA necessitates skills such as object recognition, spatial reasoning, and relational reasoning. Consider the VQA question in Figure 1.2 (a) as an example. We can see that here multi-step reasoning is required to reach the final answer. Models need to initially locate the man in the image, then shift to the left

[1]In the rest of this thesis, to simplify the discussion, we will use *memes* to refer to *multimodal memes*.

[2]The examples are **for illustrative purpose only**. They are not real memes in hateful meme detection datasets, as showing real hateful memes at first may be distasteful.

side, locate a woman, and finally identify the item she is holding.

Thirdly, VLU tasks often require external knowledge not explicitly provided. For example, to determine if the memes in Figure 1.1 are hateful, one needs knowledge about the smells associated with skunks and roses. Similarly, addressing the question in Figure 1.2 (b) necessitates knowledge about animals in California's national parks.

In summary, VLU poses significant challenges when we apply AI techniques to address it. To address the three major challenges above, initially, researchers designed various model architectures and supervised approaches using task-specific data. To model the vision-language interactions, solutions have evolved from simple fusion strategies (e.g., concatenation and element-wise addition) [214, 6] to sophisticated multimodal fusion techniques (e.g., bilinear fusion) [45, 86] to generate more expressive joint vision-language representations. Meanwhile, to deal with complex VLU problems requiring multiple skills and multi-step reasoning, *Neural Module Networks* were proposed. This framework incorporates different modules designed for different reasoning tasks and the model trains these modules end-to-end with VLU task data [69, 91, 5]. Finally, in response to the need of external knowledge, previous work tried to retrieve and incorporate knowledge from existing knowledge bases to handle the tasks [178, 122, 130].

However, these initial approaches involved neural network models trained from scratch with task-specific data, which were prone to overfitting due to the scarcity of training data for each downstream task. They struggled to generalize to real-world applications. Recent research has shifted towards pre-training large models that can be easily transferred to different tasks. By training models on extensive datasets, pre-trained models (PTMs) acquire universal representations and show good performance when adapting to downstream tasks. In the following sections, we will first briefly introduce PTMs and then discuss how researchers have tried to adapt them to VLU tasks.

1.2 Pre-trained Models for Language Processing and Vision-Language Understanding

Supervised learning with task-specific data is limited by the training data size. Therefore, it leads to overfitting of models on training data and incapability of generalization [22, 7]. This overfitting issue prompts researchers to explore alternative approaches. A successful approach is model pre-training. By training models on large and diverse datasets spanning various domains, PTMs can learn universal representations, enabling them to generalize across different sub-tasks. In the rest of this section, I will give a brief review of recent progress in pre-trained language models and pre-trained vision-language models.

A significant amount of research on pre-training focuses on pre-training language models for natural language processing (NLP). Pre-trained language models (which we refer to as PT-LMs) are typically trained to perform the task of masked language modeling [169] (predicting masked words based on context), next sentence prediction [36] (determining if two sentences follow each other in the original document) and next token prediction [142] (predicting the next token given the previous input tokens). Because these pre-training tasks do not require annotated text, PT-LMs can be trained on plain texts easily accessible from a wide range of sources and particularly from the Web. In terms of their model architecture, modern PT-LMs are predominately based on the transformer architecture [175]. Under the general transformer architecture, there are three types of language models: encoder-only models (e.g., BERT [36] and RoBERTa [114]) can encode text and subsequently perform classification tasks. The encoder-only models cannot be used directly for text generation tasks; decoder-only models (e.g., GPT [15] and OPT [204]) exploit a decoder for both processing inputs and generating outputs. These models are strong at text generation, such as question answering and conversational chatting; finally, there are also encoder-decoder models (e.g., BART [95] and T5 [144]), which have an encoder for input encoding and a separate decoder for target output generation.

They are powerful at natural language understanding such as text summarization and machine translation. Although traditionally PT-LMs were trained on only unannotated texts, recently, researchers have found that fine-tuning these models with instructional data (i.e., pairs of texts where the first part is a human-written instruction and the second part is the expected output from a language model) can significantly enhance these models' capabilities.

Similar to PT-LMs, for VLU tasks, it also helps to pre-train models on vision-language data to increase generalizability and prevent overfitting. Therefore, in recent years, researchers have started developing pre-trained vision-language models (which we refer to as PT-VLMs). Aligned with PT-LMs, researchers have explored unsupervised or weakly supervised vision-language tasks, such as vision-language alignment prediction (predict whether an image and a sentence are paired) [117], masked vision-language modelling (predict a masked tokens in a sentence according to both contexts and image information) [117] and image-text contrastive learning (predict the semantic similarity between an image and a sentence) [98]. To delve deeper into sophisticated vision-language interactions, researchers [101, 117] have also explored complex but supervised VLU tasks such as image captioning and VQA during pre-training. In terms of pre-training data, while earlier work [101, 117, 168] used small-scale, clean and annotated datasets such as MS COCO captions [107] and Conceptual Captions [155], recent work [195, 99, 113, 111, 98, 143] all used web-scale data. For model architecture, PT-VLMs predominantly rely on transformer architecture [175]. They typically feature a vision-language encoding stage to capture cross-modal interactions. Encoder-based PT-VLMs [98, 143] focus on pre-training the encoder for expressive vision-language representations, excelling in understanding-based tasks such as image-text retrieval. On the other hand, encoder-decoder-based models [182, 29] demonstrate proficiency in visual-grounded text generation tasks like image captioning. Additionally, decoder-only PT-VLMs [111, 195] leverage large language models, performing the vision-language encoding and output decoding within a decoder, and excel in conversational applications involving both images and texts. Vision-language instructional data were also explored to acquire better generalization to unseen data [111, 195]

We have reviewed recent progress with PT-LMs and PT-VLMs. We now briefly review how PTMs are being used for downstream tasks. The ways PTMs are applied for downstream tasks have evolved and can be roughly divided into two paradigms. In the beginning, these models were fine-tuned on task-specific data [36, 101, 117]. With the scaling up of model sizes and pre-training data, PTMs have become increasingly capable of being applied directly to downstream tasks in a zero-shot or few-shot [2, 9, 96] manner, provided that tasks can be appropriately converted to objectives similar to pre-training tasks.

However, these methods for applying PTMs to VLU tasks suffer from a few limitations.

Firstly, direct fine-tuning of PTMs with task-specific VLU data regards the task as a general VLU task and overlooks the unique characteristics of the task. Take the task of hateful meme detection for instance. The comprehension of target entities is of vital importance to hateful content detection [34], but direct fine-tuning cannot give special treatment to the target entities embedded in the hateful memes.

Secondly, the zero-shot and few-shot performance of PTMs on some tasks may be poor, which prohibits the direct application of frozen PTMs. For example, PT-VLMs have near-random guess performance regarding the hateful meme detection task in the low-resource setting.

Thirdly, the complexity of VLU tasks requires multiple skills that a single PTM may not possess. Furthermore, they may require multi-step reasoning which most PTMs lack. For instance, the task of VQA requires the capability of object detection, spatial reasoning and relation reasoning as well as multiple reasoning steps before reaching the final answer.

1.3 Thesis Statement

To address the limitations identified earlier in applying PTMs for VLU tasks, in this thesis, we propose and evaluate a few new approaches. Our work can be organized based on whether we leverage fine-tuning or zero-shot/few-shot learning, and whether we adopt a single PTM or a composition of PTMs. We summarize the four categories of optimizing the utilization of PTMs to VLU tasks in our thesis as follows:

- **Tuning-Single**: It is the most natural way and has been extensively explored in the past. This thesis aims to explore methods to better integrate PTMs with task-specific components, thereby enhancing their effectiveness in addressing the challenges in VLU tasks. Formally we define the model for the task as $\mathcal{M}$ and $\mathcal{M} = f_\theta(\mathcal{M}_{\text{PTM}}, \mathcal{M}_{\text{tsp}})$, where $\mathcal{M}_{\text{PTM}}$ denotes the PTM, $\mathcal{M}_{\text{tsp}}$ denotes the task-specific part (note, $\mathcal{M}_{\text{tsp}}$ can be either neural networks or specifically designed method for tuning PTMs), and f_θ refers to the architectures connecting the PTM and the task-specific component. The parameters in $\mathcal{M}_{\text{tsp}}$ (if neural networks) and f_θ are learnt end-to-end with the update of $\mathcal{M}_{\text{PTM}}$. Take the task of video question answering for instance. Existing PT-VLMs trained with image-text data can only comprehend information within a frame, whereas incapable of capturing temporal information (e.g., the transition of an action) across frames. Thus, $\mathcal{M}_{\text{tsp}}$ can be specially designed for modelling the frame-to-frame temporal relations and be combined with $\mathcal{M}_{\text{PTM}}$ responsible extracting information within a frame. The design of $\mathcal{M}_{\text{tsp}}$ and f_θ to better cater $\mathcal{M}_{\text{PTM}}$ for a specific task is the main contribution in this category of using PTMs.

- **Frozen-Single**: With the increasing size of model architectures and limited accessibility to some models, using frozen PTMs becomes prevalent. However, frozen PTMs may not be directly applicable to some VLU tasks or direct applications of PTMs leads to poor performance. In the thesis, we study how to facilitate sub-steps in VLU tasks with frozen PTMs. Formally, the model

$\mathcal{M}$ can be represented with $\mathcal{M} = (\mathcal{M}^{\text{frozen}}_{\text{PTM}}, \mathcal{M}^{\text{tune}})$. One widely used strategy in this category is to use frozen CLIP [143] as a feature extractor to facilitate visual and textual representations in VLU tasks [127, 89]. The key and the contribution in this type is to identify the intermediate step that is in need of improvement and convert the step to pre-training objectives of a PTM.

- **Tuning-Composition**: VLU tasks may be complex and have poor zero-shot / few-shot performance. We seek to tune PTMs to acquire essential reasoning skills for such tasks and compose the set of tuned PTMs for these tasks in the low-resource setting. Specifically, we only have access to a few annotated example, $\mathcal{D}_{\text{train}}$ for the task. Assume K essential skills are identified for the task, we then tune PTMs to acquire these skills with auxiliary data and obtain a set of tuned models $\{\mathcal{P}^{\text{tune}}_k\}^K_{k=1}$. Based on $\mathcal{D}_{\text{train}}$, a composer is learnt to construct the model $\mathcal{M}$ for the task. One application of this strategy lies in a setting where users require assistance for various VLU tasks, such as navigation, multimodal recommendation and visual question answering. A plausible strategy is to have tuned models for each VLU task and use the few annotated examples from users (e.g., their instruction and desired feedback) to train a composer, which learns to compose tuned models confronted with different user instructions.

- **Frozen-Composition**: Different PTMs possess unique strengths and complex down-stream tasks may require various capabilities that a single PTM may not possess. This approach aims to make use of a composition of frozen PTMs and assign sub-reasoning tasks in complex VLU tasks to proper PTMs without requiring any adaption. Given a complicated task $\mathcal{T}$, we can decompose it into a sequence of sub-reasoning tasks: $\{t\}^{N_1}_{n=1}$, where N_1 is the number of sub-tasks and t_n is the n-th reasoning task. We then convert each t_n to objectives similar to pre-training goals of PTMs, assign t_n to a proper PTM, $\mathcal{P}^{\text{frozen}}_n$ and obtain a sequential outputs from PTMs. In other words, the set of composition of PTMs ($\{\mathcal{P}^{\text{frozen}}_n\}^{N_2}_{n=1}$ is responsible for identified sub-reasoning tasks (Note,

N_1 may not be equal to N_2). The sequential outputs will be organized to generate the task prediction $\mathcal{O}_{\text{task}}$. For instance, given a complex instruction asking for converting the object in a given image to the other style, it involves object recognition, style transfer and image generation. The three decomposed tasks can then be assigned to an object detector, a style transferring model and an image generator, respectively.

1.4 Task Definition

There is a wide range of tasks in need of VLU. This thesis narrows its focus to two specific VLU tasks, namely, *Hateful Meme Detection* [85] (**HMD**)[3] and *Visual Question Answering* [54] (**VQA**), and makes use of PTMs to facilitate the two tasks.

The decision to concentrate on these tasks stems from their direct relevance to real-world applications. Given memes, often comprising images with short texts, a HMD system is expected to distinguish between *hateful* and *non-hateful* memes [85]. The spread of online hateful memes may sow discord among individuals or communities online and potentially result in violent hate crimes. HMD plays an important role in combating online misbehavior and safeguarding individuals from harm. On the other hand, a VQA system is expected to accurately respond to questions posed about an image. Among its various applications, one of the most critical is to aid visually impaired individuals. Additionally, VQA systems have potential applications in autonomous driving, food recommendation, robot navigation, and numerous other domains.

Furthermore, both tasks present non-trivial challenges that demand innovative approaches. Firstly, both tasks requires comprehending vision-language interactions. With only the meme image or the meme text, it is infeasible to decide whether the multimodal meme is hateful or not, as discussed in Section 1.1. Similarly, in VQA, both the question and the image are indispensable for answering a VQA ques-

[3]Though not a well-adopted acronym, we use HMD as an abbreviation for hateful meme detection in the rest of the thesis.

tion. Secondly, external knowledge such as commonsense and cultural background knowledge, which may not be readily available in training data, necessitates for both tasks. Specifically, hateful memes are often associated with historical events and cultural backgrounds, while VQA questions are often asking about commonsense related to entities in images. Thirdly, HMD and VQA are complex, as multiple reasoning skills and multi-step reasoning are needed. HMD asks for the capabilities of comprehension of the definition of "*hateful*", understanding visual metaphors in memes, decoding the underlying meaning, etc. VQA requires several skills, such as object recognition, spatial reasoning and relational reasoning.

For these reasons, we believe these two tasks emphasize the importance as well as challenges of VLU tasks.

Though there have been explorations in applying PTMs to the two VLU tasks, they have several limitations. Regarding to using tuned PTMs, a prevalent method is to directly fine-tune PTMs with task-specific data. Nonetheless, direct fine-tuning of PTMs overlooks unique characteristics of different tasks (e.g., the recognition of victims of hateful memes in HMD) and treats them as general VLU tasks. Meanwhile, direct fine-tuning of PTMs on task-specific data requires large volumes of supervised data, which is impractical in some real-world settings. For example, the dynamic nature of hateful memes tied to evolving events poses significant hurdles in acquiring and annotating sufficient training examples.

On the other hand, researchers have tried utilizing frozen PTMs by converting downstream tasks into those similar to pre-training objectives of PTMs. However, in some cases, frozen PTMs cannot be applied directly due to poor performance. For instance, PT-VLMs have near-random performance on HMD. Besides, PTMs may not possess or excel in all essential skills for complex tasks (e.g., VQA asks for multiple reasoning skills) and multi-step reasoning in complex tasks is challenging to PTMs.

In this thesis, we present methods for optimizing the utilization of PTMs for the two VLU tasks.

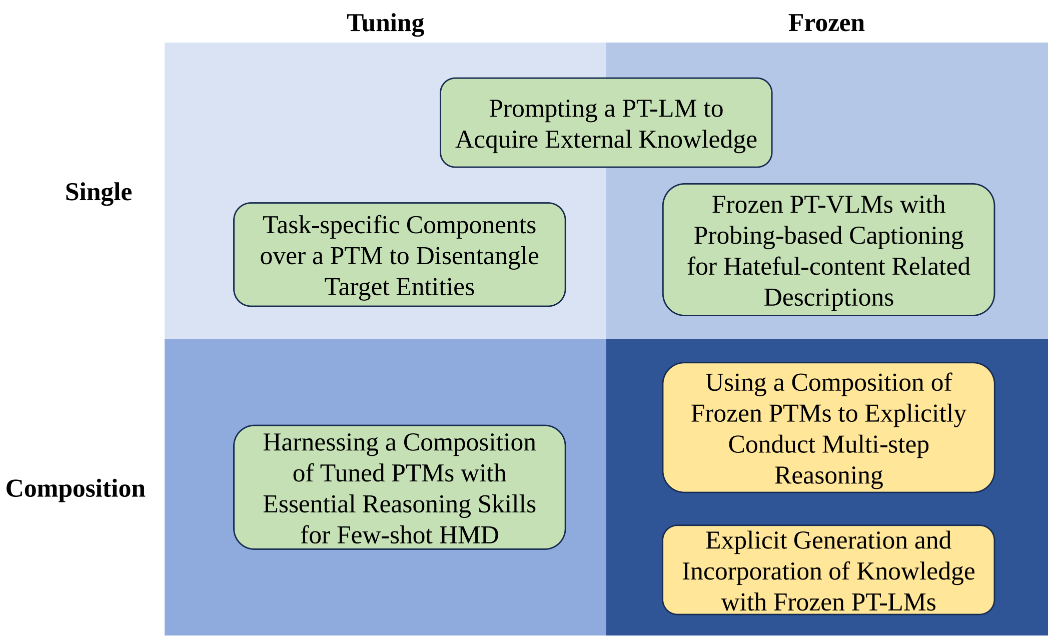

Tuning
Frozen
Single
Composition
Prompting a PT-LM to Acquire External Knowledge
Task-specific Components over a PTM to Disentangle Target Entities
Frozen PT-VLMs with Probing-based Captioning for Hateful-content Related Descriptions
Harnessing a Composition of Tuned PTMs with Essential Reasoning Skills for Few-shot HMD
Using a Composition of Frozen PTMs to Explicitly Conduct Multi-step Reasoning
Explicit Generation and Incorporation of Knowledge with Frozen PT-LMs

Figure 1.3: An overview of thesis contributions. We follow four strategies for utilizing PTMs: Tuning-Single, Frozen-Single, Tuning-Composition and Frozen-Composition, and design methods to better use PTMs to cater for HMD and VQA. The green blocks are for the HMD task, the yellow blocks for VQA.

1.5 Overview of the Thesis

In this thesis, we aim to build systems to address the two challenging VLU tasks introduced above, namely, HMD and VQA, with the exploitation of PTMs. To adapt PTMs to the tasks, we try four strategies, Tuning-Single, Frozen-Single, Tuning-Composition and Frozen-Composition, as discussed in Section 1.3. We made the following contributions to support the thesis statement (as illustrated in Figure 1.3):

Task-specific Components over a PTM to Disentangle Target Entities: Instead of direct fine-tuning PTMs for HMD, we design and incorporate a PTM into a novel framework that disentangles target entities in memes and tunes the PTM with task-specific components end-to-end. The method improves both the performance and interpretability. However, the method suffers from the limitation that HMD requires contextual background knowledge.

Prompting a PT-LM to Acquire External Knowledge: To address the lack of contextual background knowledge, we leverage a PT-LM as an implicit knowledge source via prompting. Since PT-LMs are inherently textual, the method involves a frozen PT-VLM for converting meme images into textual captions. Then, the multimodal classification task can be transformed into a masked word modeling task. We further add two demonstrations and prompt PT-LMs for the prediction. The proposed method achieves the state-of-the-art performance on two benchmarks, whereas, the method suffers from non-informative captions. Generic image descriptions may lack crucial details, such as race and gender information, vital for detecting hateful content.

Frozen PT-VLMs with Probing-based Captioning for Hateful-content Related Descriptions: To address the limitation of non-informative captions, we proposed a probing-based captioning approach to leverage a frozen PT-VLM to complement the previous method. Specifically, we prompt a frozen PT-VLM with hateful content-related questions and use the answers as image captions, ensuring that the captions contain critical information for hateful content detection. By inserting the generated captions into text-based meme detection models, significant improvements can be achieved. While our proposed methods above demonstrate strong detection performance, they demand large volumes of annotated data. The dynamic nature of hateful memes tied to evolving events, nevertheless, makes it impractical to annotate sufficient training examples.

Harnessing a Composition of Tuned PTMs with Essential Reasoning Skills for Few-shot HMD: In response, we study HMD in the low-resource setting, where only a few annotated examples are available. We propose a modularized networks, which harnesses the power of a composition of tuned PTMs, each of which possesses an essential reasoning capability for HMD. We use the few available annotated samples to train a composer, which assigns weights over PTMs based on their relevance to HMD. The composed modularized networks demonstrate superior few-shot performance, as well as computational efficiency during inference compared to traditional in-context learning.

Using a Composition of Frozen PTMs to Explicitly Conduct Multi-step Reasoning: The VQA task is complex as questions often require multi-step reasoning and most PTMs lack such capability. Secondly, the VQA task asks for different reasoning skills, while a single PTM may not possess all these skills. Meanwhile, existing zero-shot VQA models do not think about explicit reasoning chains inherent in VQA. Considering these factors above, we design a modularized zero-shot VQA model. It explicitly decomposes VQA questions into sub-reasoning tasks and assigns each sub-task to proper frozen PTMs. The good performance on two VQA benchmarks shows both the importance of question decomposition in VQA and the possibility of applying such a system to real-world, while the model is highly interpretable.

Explicit Generation and Incorporation of Knowledge with Frozen PT-LMs: Expanding our inquiry, we delved into a specific VQA scenario, K-VQA, where external knowledge apart from images is indispensable to answer questions. Recent zero-shot K-VQA models lack explicit demonstration of the knowledge used to answer questions and thus lack interpretability. We propose to explicitly generate and incorporate knowledge with frozen PT-LMs for K-VQA. The method improves the K-VQA performance as well as interpretability.

In the rest of the thesis, we will first provide a literature review of PTMs and of the two VLU tasks (HMD and VQA) in Chapter 2. We will elaborate on the origins of the two tasks from hate speech detection and text-based question answering, their extensions to multimodality and benchmarks used in the thesis. We then describe the model which disentangles target entities to help with improving HMD in Chapter 3. In Chapter 4, we show details of how to leverage language models as implicit knowledge bases to facilitate HMD. As this model suffers from non-informative captions, in Chapter 5, we introduce our probing-based captioning technique to complement the model. In Chapter 6, we present our work for HMD in the low-resource setting by embracing a composition of tuned PTMs.

Subsequently, we delve into our proposed models for VQA. In Chapter 7, we elaborate on how to design a modularized neural networks, decompose questions, assign PTMs to each sub-reasoning task and leverage frozen PTMs in a compositional manner. The proposed method to generate and incorporate knowledge with frozen

PT-LMs to facilitate K-VQA is presented in Chapter 8. Finally, in Chapter 9, we conclude the thesis and give a few future research directions regarding the two tasks.

Chapter 2

Related Work

In this chapter, we will give a more thorough review of related work. Specifically, we first present the research background of PTMs. Next, we go through related work about two tasks we focus on in this thesis, namely, HMD and VQA.

2.1 Pre-trained Models

Relying on a small amount of in-domain task-specific data limits model performance, causing models to easily overfit the training data, especially when the complexity of the models (measured by the number of parameters, for example) is increased. In the research field of computer vision, annotated datasets such as ImageNet are of large scale so that models trained on these datasets (e.g., ResNet [62], Faster R-CNN [146]) usually can work well on a wide range of settings. However, training datasets for downstream NLP tasks and VLU tasks are relatively smaller so model sizes and performance are limited. To address this issue, researchers proposed to pre-train models to facilitate downstream tasks. In the following sections, we will discuss PTMs briefly and then provide more details on PT-VLMs, because vision-language models are more closely associated with VLU tasks.

2.1.1 Pre-trained Language Models

PT-LMs aim to learn a model capable of natural language understanding. As limited by the annotated data, people adopted unsupervised learning and used a great amount of web data, such as Wikipedia and online comments for training. For instance, by randomly masking words in a sentence, models are trained to generate masked words according to context. Most existing PT-LMs are based on the Transformer architecture [175]. PT-LMs are usually classified into the following types according to their model styles:

Encoder-only PT-LMs: These models can access all input tokens and generate contextual representations for each word according to its context. The representative PT-LMs in this folder are BERT [36] and RoBERTa [114]. They are pre-trained with unsupervised tasks such as masked word prediction and next sentence prediction that force models to learn contextualization. These PT-LMs are good at inducing word or sentence representations and are usually used as text encoders and fine-tuned for downstream tasks such as sentiment classification and natural language inference.

Decoder-only PT-LMs: This category of PT-LMs is designed to predict next tokens based on previous tokens. The most representative models in this category are GPT-3 [15], OPT [204] and LLaMA [172]. Due to the nature of their pre-training tasks, such as next word prediction, these PT-LMs are good at generation tasks such as story generation.

Encoder-decoder PT-LMs: PT-LMs under this type are hybrids of the two kinds above. They can access all tokens during the encoding phase and the decoding phase, they predict the next token based on all previous tokens. The representative models are BART [95] and T5 [144]. These models are good at sequence-to-sequence tasks, such as machine translation and text summarization.

Besides the standard pre-training data from Wikipedia, books or web-data, recent studies exploited instruction data for model pre-training [31, 181]. Confronted with different tasks, different instructions will be provided as inputs to models together with the task inputs. These PT-LMs using instructions during pre-training show

Model	Text	Vision	Objectives
		PT-VLMs with In-domain Data	
VisualBERT [101]	BERT	Faster R-CNN	MLM, ITM
ViLBERT [117]	BERT	Faster R-CNN	MLM, MRC
LXMERT [168]	BERT	Faster R-CNN	MLM, ITM, MRC, MRFR, VQA
Uniter [210]	BERT	Faster R-CNN	MLM, ITM, WRA, MRFR, MRC
OSCAR [102]	BERT	Faster R-CNN	MLM, ITM
VL-T5 [29]	BART, T5	Faster R-CNN	MLM, ITM, VQA, VG, IC
ViLT [87]	ViT	Linear Projection	MLM, ITM
		PT-VLMs Incorporating Web Data	
FLAVA [159]	ViT	ViT	MLM, MMM, MIM, ITM, CL
VLMo [13]	BERT	ViT	MLM, ITM, CL
ALBEF [98]	BERT	ViT	MLM, ITM, CL
BLIP [99]	BERT	ViT	ITM, IC
Flamingo	Chinchilla	NFNets	CMLM
BLPT-2	OPT, FlanT5	ViT	ITM, CL, IC
		PT-VLMs Incorporating Instruction Data	
Mini-GPT-4	Vicuna	ViT	Ins.T
LLaVA	LLaMA	ViT	IC, Ins.T
mPLUG-OWL	LLaMA	ViT	IC, Ins.T
InstructBLIP	FlanT5, Vicuna	ViT	ITM, CL, IC, Ins.T

Table 2.1: A summary of representative PT-VLMs regarding their text encoder, visual encoder, pre-training objectives and pre-training data corpus. For pre-training objective, we denote *word-region alignment* as WRA, *visual grounding* as VG, *Image Captioning* as IC.

good performance when generalizing to unseen data, highlighting the importance of instruction tuning.

2.1.2 Pre-trained Vison-Language Models

Impressed by the great improvements made by PT-LMs, PT-VLMs were introduced to deal with VLU tasks. Mostly, they are based on existing PT-LMs and the high-level idea of pre-training is to bridge the gap between the text encoder and the vision encoder so that knowledge in PT-LMs can be distilled to multimodality.

To better exploit pre-training data, besides using supervised learning during pre-training, some unsupervised tasks are also used for PT-VLMs, similar to PT-LMs. For instance, *masked language modeling* (MLM) learns to predict the masked word according to both its contexts and the multimodal information (i.e., image information). *Image text matching* (ITM) can be used for unsupervised learning on web data. *Masked region classification* (MRC) and *masked region feature regression* (MRFR) are applied to images so that the model classifies the masked region or generates the features for the masked region according to both the image contexts and the language information. To align the image representations with textual representations, *multi-modal contrastive learning* (CL) has also been widely adopted as an unsupervised goal. We divide PT-VLMs into three types according to different data they used in model pre-training: *in-domain data*, which is mostly human annotated and clean, *web data*, which is of larger scale and relatively noisy, and *multimodal instruction data*, which provides different task instructions towards different tasks.

In-domain Data Pre-training

At first, vision-language models are pre-trained on in-domain data, where most data comes from manually annotated clean datasets such as MS COCO [107] and Visual Genome [88]. The in-domain are of relatively small scale and annotated more related to VLU tasks. Some web data, such as Conceptual Captions [155] and SBU Captions [135], which are of high quality and clean, can also be regarded as in-domain data. We summarize representative PT-VLMs under this category in the first block of Table 2.1. Based on the type of vision encoders they use, we divide PT-VLMs under this category to two types: region-based and end-to-end PT-VLMs. **Region-based PT-VLMs** pre-process image features with off-shelf visual encoders which detect objects first and extract features within each detected region. Then the extracted region-based image features will be fed into the model together with input texts. Some of PT-VLMs (e.g., ViLBERT [117], LXMERT [168]) adopt a two-stream architecture that visual features and textual features will be processed individually with some self-attention layers first and then conduct cross-attention. Some PT-VLMs (e.g., VisualBERT [29, 101], Uniter [210], OSCAR [102]. VL-T5 [29]), in contrast, adopt a single-stream model architecture that concatenates textual and visual features, regard concatenation as one sequence and process the concatenated sequence with a single transformer architecture. There is no general agreement about which architecture works better for PT-VLMs. However, this line of work needs pre-processing of images when using these PT-VLMs, which is laborious and slow. Besides, they are also limited by the vision backbones as the object detectors may generate duplicated detected regions of an object in an image, making the visual inputs noisy.

Patch-based PT-VLMs, on the other hand, are motivated by the patch-based image encoding in Computer Vision [40]. ViLT was the first to consider applying the new visual encoding in PT-VLMs. Instead of two-stages of detection and feature extraction, patch-based vision encoding [40] directly applies a pure transformer to sequences of image patches. It is much faster and more flexible than PT-VLMs using region-based features. Compared with region-based PT-VLMs, we notice that when using in-domain data for pre-training, fewer works leverage end-to-end vision encoding. A plausible reason is that in-domain data is of relatively smaller scale and the region-based visual representations contain more semantic information. It is easier to bridge the multimodal gap than when using the patch image features.

Web Data Pre-training

Inspired by CLIP [143], researchers have explored using larger scale but nosier web data for pre-training. Though CLIP is powerful at several vision-related tasks, it fails in VLU tasks as the modeling of multimodal interactions, by cosine similarity, is shallow. To understand more complex multimodal interactions which are essential in VLU tasks, based on CLIP, modules to catch complicated multimodal information are added. Besides, more VLU-related pre-training objectives are added. We summarize these PT-VLMs in the second block of Table 2.1.

Most PT-VLMs leveraging noisy web data adopt the contrastive learning strategy at first for bridging the multimodal gap. Specifically, the encoded image features and the encoded text features will be forced to be in the same semantic space. After contrastive learning, traditional unsupervised learning objectives, such as MLM and ITM are also incorporated to learn more expressive multimodal representations. PT-VLMs using BERT as the text encoder exploit pre-training objectives mentioned above and are in need of fine-tuning when adapting to downstream tasks. We denote this type of PT-VLMs as models pre-trained for *representation learning*. Some PT-VLMs incorporate large PT-LMs which are powerful at conditional text generation. These PT-VLMs also adopt cross-modal language modeling as a pre-training objective. We regard these PT-VLMs are conducting *encoder-decoder pre-training*. These PT-VLMs show good zero-shot capabilities in a few VLU tasks. We provide more details about the two types of PT-VLMs below:

Representation Learning: Though web data is of larger scale, they are much nosier. ALBEF [98] exploited momentum distillation to smooth the labels for contrastive learning. BLIP [99] further enlarged the pre-training corpus by *captioning and filtering*. It first fine-tuned a pre-trained image caption generator, generated synthetic captions and filtered noisy generated captions. The remaining generated captions will be used for further pre-training. FLAVA [159] on the other hand, targeted at a foundation model also capable of computer vision-related tasks so that it included MIM which asks for predicting masked image patches according to image contexts only. VLoM [13] introduced mixture-of-modality-experts to encode inputs from different modalities, making it more flexible to various downstream tasks.

Encoder-decoder Pre-training: PT-VLMs in this category contain larger PT-LMs (with more than a billion model parameters). They aim to not only build PT-VLMs good at multimodal representing, but also image conditional multimodal generation. To achieve this goal, these PT-VLMs contain an encoder for encoding multimodal contents. The capability of the encoder is to generate expressive multimodal representations. The decoder is to generate texts according to the output from the encoder.

To distill knowledge and language modeling capability in PT-LMs, their decoders are based on frozen large language models, such as Chinchilla [65], FlanT5 [31] and OPT [204]. The high-level idea of the pre-training objectives is to let the PT-LMs understand the multimodal outputs from the encoder.

To bridge the multimodal gap, Flamingo [2] proposed a gated Xattn-dense layers. Specifically, cross-attention layers are inserted between layers of frozen PT-LMs while the keys and values are derived from visual features. The queries are generated from text inputs. In this way, Flamingo can condition PT-LMs on vision information. During pre-training, it adopts multimodal conditional language modeling as the objective. Besides, it also manually constructed some in-context examples as pre-training data so that Flamingo can be used in both zero-shot and few-shot multimodal conditional text generation. BLIP-2 [100] added a Query-Transformer to bridge the multimodal gap. In the first pre-training stage, it adopts traditional pre-training objectives of PT-VLMs focused on representation learning (e.g., ITM, CL) so that the Query-Transformer can learn to extract the visual representation that is most informative of the text. In the second pre-training stage, BLIP-2 learns to conduct multimodal conditional text generation. BLIP-2 can be directly applied for zero-shot image captioning and zero-shot VQA.

Instructions Data in Pre-training

Inspired by the instruction tuning (Ins.T) of PT-LMs, researchers have considered adding multimodal instruction data into the pre-training of VLMs. The goal in the pre-training stage is to learn a visual representation that PT-LMs can comprehend and conduct instruction tuning so that PT-VLMs can follow instructions.

To convert images into comprehensible inputs to PT-LMs, usually, these PT-VLMs use a vision encoder (e.g., ViT [40]) to extract raw representations first and then leverage an alignment network to map the raw representations to the semantic space of PT-LMs. In Mini-GPT-4 [212], a simple linear projection layer is used as the alignment network to map the visual features from a vision encoder. The simple alignment network may not be capable of capturing all visual information so that the mapping may lose information. Meanwhile, they only adopted one kind of multimodal instruction data (i.e., giving a detailed image description), which was limited. LLaVA [111] considered a two-stage instruction tuning. Firstly, they learned the alignment networks. Next, multimodal instruction data was applied to tune the alignment network together with the language model. Besides, LLaVA also contributed to a large multimodal instruction dataset, containing more diverse instructions, such as multimodal conversation and complex multimodal reasoning. Based on the larger scale multimodal instruction dataset, mPLUG-OWL [195] further improved the alignment part by tuning the vision encoder together with the added mapping layers. InstrucBLIP [113] converted existing datasets for VLU tasks into instruction formats, used a more complicated alignment network, Query-Transformer, and conducted instruction tuning. It shows great performance on several held-out VLU datasets.

2.1.3 Applications of Pre-trained Vision-Language Models

Pre-trained with large data corpus, the goal of PTMs is to learn universal representations or solutions so that can facilitate downstream tasks. In this section, we review how to apply PTMs to downstream tasks. To narrow down the scope, we focus on the PT-VLMs in this section as the main topic in the thesis is about VLU. However, the strategies to applying PTMs are also consistent with the categories described below:

Tuning: During pre-training, models learn universal representations so that they can be easily generalized to downstream tasks via direct fine-tuning. For instance, PT-VLMs are able to generate expressive multimodal features considering both image and text information. They can be tuned end-to-end with a classifier or a text decoder for open-ended VQA [168, 117, 29, 210].

In some cases, the capability of PT-VLMs is not enough to perform a downstream task. Researchers thus design additional architectures and combine PT-VLMs with these components. Considering some questions in VQA require external knowledge beyond images, in [37], the authors leveraged both the PT-VLM, LXMERT [168] and a knowledge extractor. It used the VLU capability of LXMERT for feature extraction and conducted multimodal knowledge triplet extraction based on the features. CLIP [143] has been trained with a great amount of multimodal web data and demonstrates strong representation capabilities especially towards noisy data. MOMENTA [140] considered incorporating CLIP for HMD for both meme image and meme text encoding and designed unique architectures to facilitate meme detection.

With the PT-VLMs getting larger, people considered more efficient tuning for PT-VLMs, especially when there are only a few training examples. Instead of tuning all parameters, they add a few additional tunable parameters to adapt PT-VLMs to downstream. One line of works updates the prompts fed to PT-VLMs [208, 209]. The goal is to optimize prompts in the continuous space so that the input prompts can be optimal to leverage the learned contents in PT-VLMs. The other line of work considers parameter-efficient tuning, where either only tunes a few model parameters [161] or adds adaptors that contain a few parameters [164].

Frozen: With the scaling of model sizes, PT-VLMs are able to be used in a frozen manner. Flamingo [2] first increased PT-VLMs to billions of parameters. It demonstrated strong zero-shot and few-shot performance on several VLU tasks, such as VQA and image captioning. Recently, with more large scale open-source PT-LMs, researchers considered distilling knowledge in PT-LMs and incorporating frozen PT-LMs in VLMs and conducted pre-training. These PT-VLMs, such as BLIP-2 [100] and InstructBLIP [113] also perform well in zero-shot VLU tasks.

Compositions: Though PT-VLMs are good at some VLU tasks, they are unable to be directly applied to other VLU tasks. **Noted, we review methods by the end of 2022. At that time, no PT-VLMs could directly perform zero-shot VQA or zero-shot reference expression.** For instance, though CLIP is good at image text matching (i.e., selecting the piece of text that matches best with the given image), it cannot be directly applied to the task of VQA. To leverage its image text matching capability, TAP-C [161] converted questions into masked template, generated candidate answers of the masked templates only according to textual contexts by PT-LMs, filled candidate answers into the masked template to generate prompts and match the prompts with the image. In this way, the VQA task becomes the image-text retrieval problem so that can be solved by CLIP. However, CLIP alone cannot achieve the goal and PT-LMs (e.g., T5 [144]) are used. Some questions in VQA require external knowledge that cannot be obtained in images. PT-LMs are pre-trained with large text corpus so that can be used as knowledge bases [138] to facilitate these knowledge intensive questions. However, PT-LMs cannot understand visual information in images. Consider the fact, a few works first converted images into textual descriptions with an off-the-shelf image caption generator so that PT-LMs could understand and transform VQA into a text-based QA problem and leveraged the (implicit) knowledge in PT-LMs [193, 58, 70]. For the task of reference expression, considering the limitation of PT-VLMs regarding compositional reasoning and spatial reasoning [170], ReCLIP [163] applied CLIP only to extract noun chunks of the expression and solved the spatial reasoning by manual heuristics. In conclusion, with compositions of PTMs, we can solve tasks that cannot be directly solved by a single PTM or we can make up limitations of PTMs by applying proper PTMs to suitable decomposed tasks.

2.2 Hateful Meme Detection

The proliferation of social media has enabled users to share and spread ideas at a prodigious rate. While the information exchanges in social media platforms may improve an individual's sense of connection with real and virtual communities, these platforms are increasingly exploited for the propagation of toxic content such as hate speeches [43, 152]. As stated in Chapter 1, hateful contents are expressing hate towards an individual or a certain group. The proliferation of hateful content online will lead to disharmony in communities and may potentially result in hate crimes.

At first, most efforts and progress were made in the text-based hateful content detection, *automated hate speech detection*. We provide a brief summarization of related works about hate speech detection in Section 2.2.1. However, the other kind of hateful content, *multimodal hateful memes*, makes it harder for detection as it involves the comprehension of two modalities as well as their interactions. Meanwhile, uses could re-post or share these hateful memes in multiple conversations and contexts. Therefore, it is urgent but challenging for detecting hateful memes. We provide existing works to HMD in Section 2.2.2, which is one of the main focus of our proposal. In the last section of this chapter, Section 2.2.3, we elaborate benchmarks for HMD.

2.2.1 Automated Hate Speech Detection

To combat online hate speech, various solutions are proposed. Based on how they solve the problem, we categorize these methods into two types, *classic machine learning strategies* and *deep learning strategies*.

Machine Learning Strategies: Traditional machine learning methods have been applied to detect hate speech in social media [23, 34, 27, 133, 183, 184]. Typically, these methods include an initial feature extraction phase, where features are extracted from the raw textual content. The most commonly extracted features include Term Frequency Inverse Document Frequency (TF-IDF) scores, Bag-of-Words vectors, and other linguistic attributes. Xiang et al. [187] also explored the latent semantic features extracted from tweets for hate speech detection by mining topics of the tweets. Beyond the textual content, some studies have also utilized other meta-information from the users' profiles and network structures (i.e., followers, mentioned, etc.) [23, 137, 160]. The extracted features are subsequently used as input for classifiers such as Logistic Regression, SVM, Random Forest, etc., to predict if the given tweet contains hate speeches.

Deep Learning Strategies: Deep learning methods have achieved notable performance in many classification tasks. Unlike traditional machine learning methods, deep learning methods are able to automatically learn latent representations of the input data to perform classification. Such deep learning approaches have also been applied to several NLP tasks, including text classification [52, 192]. The increasing popularity of deep learning approaches also sees a number of recent studies adopting these methods to detect hate speech in social media [19, 18, 7, 11, 38, 46, 55]. With the development of PT-LMs, the application of tuned PT-LMs [147, 10] and the zero-shot or few-shot application of frozen PT-LMs are used for hate speech detection [3, 28].

2.2.2 Automated Hateful Meme Detection

Though great progress made in automated hate speech detection, HMD is more challenging. Given a meme image with a short text on it, the system is required to predict whether it is hateful. The task involves the comprehension of two modalities and the complex reasoning between modalities. Fewer solutions, compared with text-based hate speech detection, have been proposed for HMD.

Fine-tuning of Pre-trained Vision-Language Models As a multimodal task, the most straightforward solution is to fine-tune PT-VLMs on HMD data to distill the capability of VLU in PT-VLMs [213, 211, 108, 128, 176]. To further enhance the capability of PT-VLMs, some works considered model ensemble, which fine-tuned multiple PT-VLMs with the HMD data and conducted majority voting during the inference time. It shows good performances compared with using a single PT-VLM [176, 128, 108]. Consider the nature of the HMD task, during fine-tuning, some works also included additional image tags (e.g., object labels, image entities and races of people) to facilitate the PT-VLMs [213, 108, 211].

Incorporation of PTMs Direct application of PT-VLMs regards HMD as a downstream VLU task but ignores its unique characteristics. Considering this aspect, some works design additional structures and incorporate them with PT-VLMs. For instance, unlike other VLU tasks, the image and the meme text are sometimes not

Datasets	Train	Val.	Test
FHM	3,050 hateful, 5,450 non-hateful	250 hateful, 250 non-hateful	500 hateful, 500 non-hateful
MAMI	5,000 hateful, 5,000 non-hateful	—	500 hateful, 500 non-hateful
MultiOFF	187 offensive, 258 non-offensive	59 offensive, 90 non-offensive	59 offensive, 90 non-offensive
HarM	1,064 harmful, 1,949 harmless	61 harmful, 116 harmless	124 harmful, 230 harmless

Table 2.2: Statistical summary of datasets for HMD.

correlated. HMD often calls for external background knowledge beyond the image. Besides, memes are noisy. Based on these considerations, MOMENTA [140] leveraged CLIP [143] to deal with the noise in data. However, CLIP only has shallow multimodal interactions. MOMENTA further added several intra-modality attention components to perform multi-level multimodal interactions. Hate-CLIPper [89] continued using CLIP for text and image representations but considered more complicated and expressive multimodal fusion.

Besides the works mentioned above focusing on correcting classifying hateful memes and non-hateful memes, there are also a few works concentrating on explaining hateful memes. In [124], the authors considered pointing out the vulnerable targets of hateful memes and provided fine-grained annotated data where each hateful meme was annotated with its hateful target. Similarly, in [140], harmful memes are annotated with their targets and the authors proposed a neural network for both detecting harmful memes and their targets. HatReD [64] provided the underlying hateful contextual reasons, which further facilitated the understanding of hateful memes. The authors in [63] used a post-hoc gradient-based explanation method to analyze how PT-VLMs conduct HMD, further pointing out the importance of explainability of models and the importance of eliminating bias in PT-VLMs.

2.2.3 Benchmarks

The task of HMD is a relatively new research area and has been formally formulated in [85]. Besides, annotating hateful memes is complicated and may require linguistic knowledge. As a consequence, there are few datasets for HMD. In this section, we briefly introduce widely used benchmarks for HMD and discuss about relevant datasets used to validate models' generalization to anti-social (e.g., harmful or offensive) multimodal meme detection. We use these datasets to validate our proposed models for HMD. We provide statistics of datasets in Table 2.2.

Hateful Memes Only Though several datasets have been proposed for text-based hate speech detection [183, 184, 34], the first formal HMD dataset was proposed in [85]. The *Facebook Hateful Meme* (**HFM**) dataset is a synthetic dataset that contains synthetic memes with added confounders such that unimodal information is insufficient for detection and deep multimodal reasoning is required. The FHM dataset contains hateful memes targeting various vulnerable groups in categories including *Religion*, *Race*, *Gender*, *Nationality*, and *Disability*. Besides FHM which contains hateful memes targeting at diverse vulnerable groups, the other hateful meme dataset, *Multimedia Automatic Misogyny Identification* (**MAMI**) dataset focuses on a particular type of hateful memes, namely, those targeting women. Performance on MAMI therefore reflects the capability of HMD methods for female victims. Different from FHM, MAMI contains real memes from online social media platforms and websites, making the data noisier.

Datasets for Generalization Due to the difficulty of hateful meme annotations, there are few datasets for HMD. When considering a broader scope, there are also relevant datasets for anti-social meme detection. We consider the datasets for either offensive or harmful memes to be relevant datasets as hateful memes are always offensive and harmful. These datasets can be used to evaluate models' generalization to more diverse anti-social memes. *Multimodal Offensive Meme* dataset (**MultiOFF**) considered both hateful memes and other related anti-social memes about trolling or cyberbullying, which are annotated as offensive. Memes on MultiOFF are related to the 2016 U.S. Presidential Election and are collected from online social platforms such as Reddit, Facebook, Twitter and Instagram. The goal of the dataset is to validate whether models can distinguish offensive memes from non-offensive ones. To force models focusing on both modalities, models can only give correct predictions after understanding both modalities and the underlying meaning of the whole meme. The other relevant dataset is *Harmful Meme Detection* dataset (**HarM**), which contains real-world memes about COVID-19. It annotates memes into harmful, partially harmful and harmless. To unify it with other datasets which are binary classification datasets, we merge harmful and partially harmful into one label, harmful.

2.3 Visual Question Answering

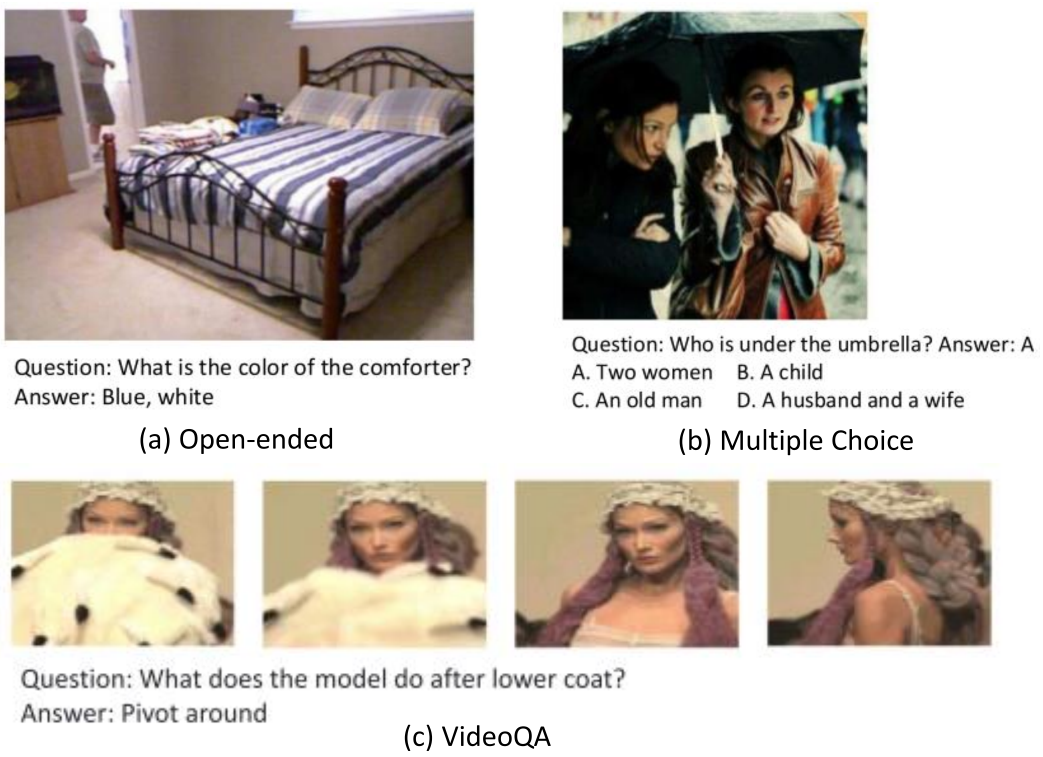
The capability of AI systems to answer questions in different scenarios has received a lot of attention in recent years. Acquiring such capability, AI systems can be applicable to various situations. For instance, systems able to answer text-based questions can facilitate child education and search engines. When extending the question answering (QA) capability to the multimodal situation, the AI systems can further assist visually impaired persons and be applicable to online multimodal product recommendations. In the past few years, the research field of QA has attracted a lot of attention and witnessed great progress. In the following sections, we will first give a review of research about question answering. Specifically, we first briefly introduce text-based QA and shift to multimodal QA, image-based VQA and video-based VQA. In the second section, we dive into recent proposed solutions to the image-based VQA, which is the main focus of the proposal. If not stated otherwise, we use VQA for image-based VQA in the section and the last section. Finally, we discuss widely used benchmarks for model evaluation of the VQA task.

2.3.1 Question Answering and Extension to Multimodality

Question answering has been an important topic in the research field as it has many application scenarios. Initially, people focused on unimodal QA, text-based QA.

Text-based Question Answering The goal of text-based QA is that QA systems automatically provide a natural language answer to a question asked by human beings in natural language. To answer the question, the QA process can be divided into three steps: 1) question comprehension, 2) retrieving and extracting relevant information to the question and 3) answer generation [61, 94]. According to where the QA systems' relevant information comes from, text-based QA can be further divided into three types: Machine Reading Comprehension (MRC), Knowledge-based QA (KBQA) and Open-domain QA (OpenQA). MRC provides pieces of passages and the relevant information has been included in these passages so that there is no step for information retrieval. The systems are expected to extract a span as the answer from the provided contexts. Instead, both KBQA and OpenQA call for information retrieval first to retrieve relevant information to aid in answering questions. For KBQA, the information comes from structured knowledge bases while OpenQA needs to retrieve relevant unstructured texts from local files or from the web.

Extension to Multimodality: Images and Videos As mentioned previously, we live in a multimodal world so real-world applications call for the capability of VLU. QA is an important capability that people expect AI systems should have. Naturally, people would like to see QA systems in multimodal settings. In the research aspect, the text-based QA extends to multimodality, such as image-based QA (ImageQA) [6, 54] and video-based QA (VideoQA) [104, 93]. For the task of ImageQA, the source of information comes from both the text modality (i.e., question) and the visual modality (i.e., image). Given an image and the corresponding question, the systems are expected to provide an answer to the question. Similar to text-based QA, some ImageQA questions are difficult so that models need to retrieve external knowledge either from structured knowledge bases [179] or unstructured text corpus [122, 154] Compared with ImageQA, VideoQA is more difficult to solve. Its visual modality extends from one image to a long sequence of images and its questions can be designed to examine the content in one frame or multiple frames. It requires not only the visual comprehension within each image but also the temporal relationship among multiple frames that have certain connections. We provide examples of the two VQA tasks in Figure 2.1.



Question: What is the color of the comforter?
Answer: Blue, white

Question: Who is under the umbrella? Answer: A
A. Two women B. A child
C. An old man D. A husband and a wife

Question: What does the model do after lower coat?
Answer: Pivot around

Figure 2.1: Examples from ImageQA and VideoQA. The two examples in the first row are from ImageQA where (a) is for open-ended QA while (b) for multiple choice QA. The second row is an example from VideoQA.

2.3.2 Deep Learning Solutions to VQA

In this section, we provide a more detailed review of the ImageQA task. If not stated otherwise, we use VQA to represent the ImageQA task. We review related works of VQA regarding how they obtain the VQA models. Specifically, at the beginning, researchers designed model architectures and trained them from scratch with VQA data. With the development of PT-VLMs, fine-tuning or incorporating PT-VLMs becomes a trend for VQA. With the scaling of model sizes, PT-VLMs demonstrate zero-shot capabilities so that a few studies focus on zero-shot VQA. Besides direct application of PT-VLMs, another line of work also tries compositions of PTMs for VQA.

Train from Scratch Traditionally, people follow a two-stream framework to deal with information from two modalities. Generally, the models contain 1) a text encoder, 2) an image encoder, 3) a fusion component and 4) an answer decoder.

The text encoder and the image encoder extract textual representations from the question and visual representations from the image independently first and conduct multimodal fusion to generate a fused representation considering the multimodal interactions. The fused multimodal representation will be used for answer prediction. The previous VQA models were mostly trained from scratch with VQA data with the image encoder frozen.

For the image encoding, most works exploited an off-the-shelf pre-trained feature extractor (e.g., GoogleNet [167], VGGNet [158] and ResNet [62]) to convert an image into either a vector or grid-based features [121, 6, 48, 188, 45, 14]. However, these features may not capture the semantic meanings of an image as one vector representation may lose detailed information and grid-based features separate objects into different regions ignoring the object-level semantics. With the development of object detection models, people started using object detection models, such as Faster R-CNN [146] for generating object-level representation to increase visual semantics [4, 72, 86]. The image representations are pre-processed and the visual encoders are frozen. For textual representation, people exploited LSTM [166] as the question is a sequence of words [6, 45, 188, 121]. Also, motivated by the good performance of convolution neural networks (CNN) in computer vision community, researchers also considered applying CNN for text feature extraction in VQA [191, 119]. With the development of PT-LMs, such as BERT [36], people also use PT-LMs as the text encoder [123, 51]. Usually, the text encoder will be tuned end-to-end with the whole VQA model.

Besides the encoders, a very important component in VQA models is the multimodal fusion component, which models the multimodal interactions. In the past years, various solutions have been proposed in VQA focusing on generating more expressive multimodal fused features. At first, people considered shallow multimodal interactions that conducted direct concatenation, element-wise addition or element-wise multiplication between textual and visual representations [6]. To guide modelsto focus on more relevant areas in images according to the question, an attention mechanism was applied. Initially, people used one-hop attention and attention in the visual channel (i.e., text-to-image attention) [4]. Later, people explored multi-hop attention [191] and co-attention in both channels [132, 129]. some works explored the bilinear fusion mechanism which allows every position of representations from two modalities to interact and tried different strategies to apply it to VQA [86, 45, 14]. Recently, inspired by the power of self-attention and cross-attention in transformers [175], a few have tried to use the attention mechanism in transformers for multimodal interactions [72, 49]. For more details about multimodal fusion techniques, please refer to the survey for more details [202].

For answer decoding, different VQA tasks will have different types of decoders. For multiple choice type of VQA, the task will be regarded as a classification task so that the multimodal inputs (the text representation will be extracted from both the question and candidate answers) will be fed to a classifier for prediction. For open-ended VQA, some of them regard it as a classification task so that a classifier will be applied over the multimodal representation [6, 14, 45]. Meanwhile, a few works regard open-ended VQA as a generation task so that a text decoder will be applied over the multimodal features [121, 48]

Fine-tuning Pre-trained Vision-Language Models The training data of each VLU task is limited so that models are easily overfitting to training data. To solve the issue, people adopted the pre-training strategies in PT-LMs and proposed several PT-VLMs.

The pre-training strategies varied in three stages. Firstly, most PT-VLMs adopt clean data for pre-training. Some of the data are manually annotated and some are constructed with self-supervision [168, 117, 101, 102]. Motivated by CLIP [143] which used a great amount of web data and trained models with contrastive objectives, several PT-VLMs also adopted a similar pipeline [98, 99]. The works mentioned above have focused on representation learning (i.e., learning good multimodal representations) and need fine-tuning when applying to downstream VLU tasks. With the larger and larger PT-LMs, researchers have tried to distill their knowledge into

Dataset	**# Ques.**	**# Img.**
VQA-v1.0 [6]	614,163	204,721
VQA-v2.0 [54]	1,228,326	204,721
GQA [73]	22,000,000	113,000
Visual7W [214]	327,939	47,300
FVQA [179]	5,826	2,190
KB-VQA [178]	2,402	700
OKVQA [122]	14,055	14,031
A-OKVQA [154]	24,903	23,692
VCR [199]	290,000	99,904

Table 2.3: Statistical distributions of the VQA datasets. The first block contains general VQA datasets and the second block contains knowledge-intensive VQA datasets.

multimodal settings. Their main pre-training goal is to bridge the multimodal gap so that the visual inputs can be understood by PT-LMs and the PT-LMs are usually frozen so that the pre-training is parameter-efficient. These models can be directly applied to downstream VLU tasks and also have superior performance when fine-tuned for VLU tasks. Considering that PT-VLMs are getting larger, instead of tuning all parameters, some works are conducting parameter-efficient tuning like prompt tuning [208, 209], inserting adapters [164] or updating partial parameters [161].

Zero-shot VQA When PT-VLMs are not large enough so that they cannot do zero-shot VQA directly, people designed networks with compositional PTMs to perform zero-shot VQA. Most of them convert images into textual descriptions, converted VQA into text-based QA and leveraged PT-LMs for answer generation [161, 58, 70, 171]. PICa [161] convert the images into textual descriptions with an off-the-shelf image caption generator [127] and optionally added image tags. However, the image captions may not always capture essential cues for answering questions. PNP-VQA [171] uses gradient-based explanation tool to highlight regions related to the question according to an image-text-matching model [99]. PromptCap [70], on the other hand, fine-tune PT-VLMs to generate question-relevant image descriptions. Img2LLM [70] considered automatically generating demonstrations when prompting PT-LMs.

2.3.3 Benchmarks

In this section, we discuss benchmarks for VQA. According to the information source, we divide VQA into two types: 1) *general VQA* where all information can be obtained within the image and 2) *knowledge-intensive VQA* where external knowledge beyond the image is needed. We summarize statistics of datasets in Table 2.3.

General VQA According to answer types, we further divide general VQA into two types: 1) open-ended VQA (as shown in Figure 2.1 (a)) and 2) multiple choice (MCR) VQA (as shown in Figure 2.1 (b)).

For open-ended VQA, the answers are open-ended words or phrases. At first, the task was formulated as a generation problem in the DAQUAR [120] and COCO-QA dataset [145]. Later, most works re-formulated VQA into a classification task. Given a pre-defined vocabulary, the models are expected to select an answer from the vocabulary [54, 6, 73, 79]. These datasets are designed to test different capabilities of VQA models. For instance, the VQA-v1.0 [6] and VQA-v2.0 [54] datasets include questions of diverse semantics as the questions are manually annotated. The questions are of short reasoning chains (most of them require a single step reasoning) and aim to test general VQA capabilities such as object localization, attribute recognition and relational reasoning. The CLEVR dataset [79], on the other hand, is synthetically generated. It aims to test the capability of compositional reasoning in VQA models so that most questions involve more than one step reasoning. The images contain simple objects with few attributes, which eliminates the difficulty in vision perception and are usually used as an analysis dataset for evaluating the compositional reasoning capability of VQA models. Besides, it is annotated with explicit reasoning chains to reach the final answer. Similarly, the GQA dataset [73] also contains synthetic questions in need of multi-step reasoning. However, the GQA dataset contains real-world images so that the good performance on GQA reflects how good the model's compositional reasoning capability is towards real-world scenes. Besides these datasets, there are also datasets that are splitted from datasets mentioned above but with specific goals. For instance, VQA-CP [1] origins from [54] while the images and questions are specially selected so that models relying on linguistic bias will fail. GQA-OOD [82] is proposed to test model performance on long-tail answers so highlighting the importance of mitigating models' bias towards frequent answers over rare answers. One of the widely used VQA datasets is Visual7W [214], whereas currently most datasets are about open-ended VQA.

Knowledge-intensive VQA In some cases, answering VQA questions requires external knowledge beyond images. These questions are called knowledge-intensive VQA questions. There are also datasets proposed for knowledge-intensive VQA [178, 179, 122, 154, 199]. At first, the knowledge-intensive questions are constructed based on external knowledge bases [178, 179], such as DBpedia [8] and ConceptNet [110]. In other words, knowledge bases can provide all needed external information for these questions. Recently, people proposed open-domain knowledge-intensive VQA datasets [122, 154, 199] so that a few knowledge bases cannot cover all needed knowledge for questions. Questions are designed to examine a wide range of world knowledge, such as visual knowledge, commonsense knowledge about human social behavior and physical knowledge, which are closer to real-world knowledge-intensive VQA situations.

Chapter 3

Task-specific Components over a PTM to Disentangle Target Entities in Memes

A system for HMD is expected to distinguish between *hateful* and *non-hateful* memes. As stated in Section 2.2.3, HMD extends hateful content detection from the text-only modality (i.e., hate speech) to multimodality (vision-language) and is more challenging. It requires both the comprehension of vision and language modality and the complex reasoning between two modalities.

With the development of PT-VLMs, fine-tuning PT-VLMs for HMD has been a straightforward method to bridge the multimodal gap. Therefore, direct fine-tuning of PT-VLMs has been adopted in most existing works [213, 211, 108, 128, 176]. They treat HMD as a general downstream task in VLU and ignore the unique characteristics of HMD. Nevertheless, these existing methods have limited explainability and cannot reason the context embedded in the hateful memes.

Following the definition of hate speech in [34], hateful memes should be defined as *memes that are used to express hatred towards a targeted group or are intended to be derogatory, to humiliate, or to insult the members of the group*. In the extreme case, a meme may be offensive (e.g., containing slurs) but not hateful as it is not targeting

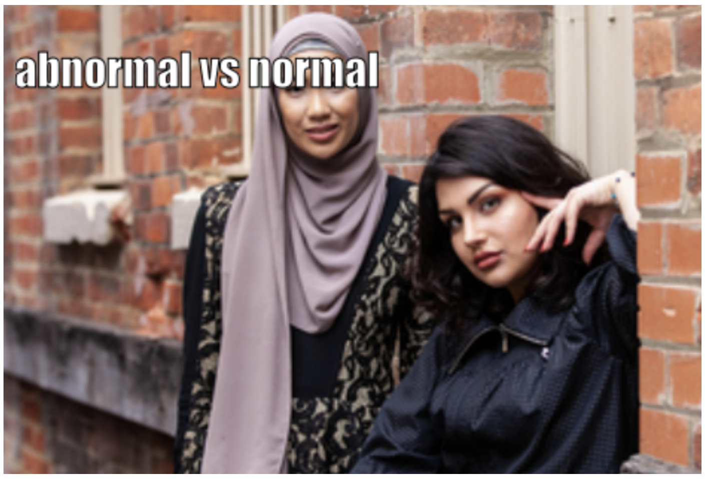


Figure 3.1: Example of hateful meme.

at any individuals or groups. Therefore, understanding the targets of memes is crucial for HMD. For the example in Figure 3.1, the target entity of the hateful content would be both gender and religion (i.e., female Muslim). Existing multimodal hateful meme classification models are unable to capture such target entity contextual information. Instead of direct fine-tuning PTMs, we tried to incorporate them into our designed model which was aware of the specific characteristics of HMD and leveraged the capability of PTMs.

This chapter aims to address the research gaps by proposing a novel framework, DisMultiHate, which learns and disentangles the representations of hate speech-related target entities, such as race and gender, in memes to improve the hateful content classification. Our framework includes a novel self-supervising training task that enables us to extract the target entities using disentangled latent representations. The disentangled representations serve as contextual information to improve hateful meme classification.

DisMultiHate aligns with the **Tuning-Single** strategy of utilizing PTMs in Section 1.3. The model incorporates a PTM, $\mathcal{M}_{\text{PTM}}$ and a task-specific component, $\mathcal{M}_{\text{tsp}}$, learning to disentangle target entities in memes, as well as architectures f_θ, bridging the two parts. The parameters in $\mathcal{M}_{\text{tsp}}$ and f_θ are learnt end-to-end with the tuning of $\mathcal{M}_{\text{PTM}}$. Our contributions lie in identifying the unique characteristic of HMD (mining the target entities of memes), designing the task-specific component

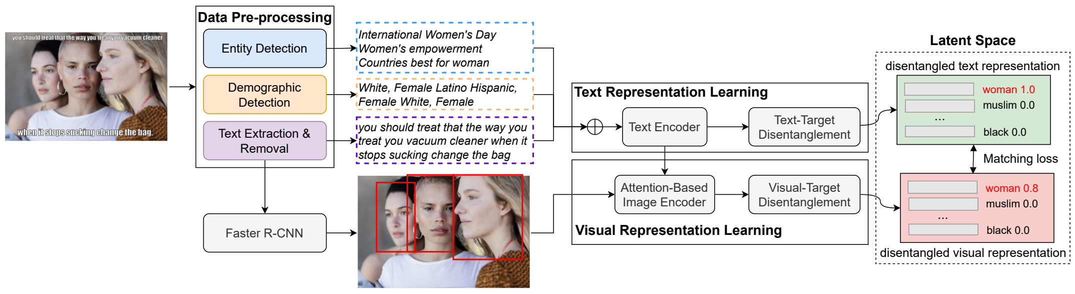

you should treat that the way you treat your vacuum cleaner
when it stops sucking change the bag
Data Pre-processing
Entity Detection
Demographic Detection
Text Extraction & Removal
Faster R-CNN
International Women's Day
Women's empowerment
Countries best for woman
White, Female Latino Hispanic, Female White, Female
you should treat that the way you treat you vacuum cleaner when it stops sucking change the bag
Text Representation Learning
Text Encoder
Text-Target Disentanglement
Attention-Based Image Encoder
Visual-Target Disentanglement
Visual Representation Learning
Latent Space
disentangled text representation
woman 1.0
muslim 0.0
...
black 0.0
Matching loss
woman 0.8
muslim 0.0
...
black 0.0
disentangled visual representation

Figure 3.2: Architecture of our DisMultiHate model.

to cater for the characteristic and improving the application of PTMs to HMD.

In the rest of the chapter, we first introduce the high-level idea of the proposed model, DisMultiHate and dive into model details in Section 3.1. We conduct experiments to validate the effectiveness of DisMultiHate on two benchmarks (at the time of submission) and provide experiment results in Section 3.2. Finally, we present the conclusion of the chapter in Section 3.3.

3.1 DisMultiHate Model

Figure 3.2 illustrates the architectural framework of our proposed DisMultiHate model. Broadly, DisMultiHate consists of three main modules: (a) *data pre-processing*, (b) *text representation learning*, and (c) *visual representation learning*. We first provide a formal task definition for HMD. The details of the data pre-processing module are discussed in Section 3.1.2. As mentioned above, understanding the target entities of memes is essential for HMD so that the goal of representation learning is to learn both visual and textual representations aware of the meme target. In other words, we can disentangle targets from both representations. Therefore, the goal of the text representation learning module is to learn a disentangled latent representation of the combined textual information output from the data pre-processing module. To better capture the semantics in texts, we used a pre-trained BERT [36] and tuned it end-to-end with other model components so that it can generate target-aware visual representations. The details of the text representation learning module will be discussed in Section 3.1.3. The visual representation learning module aims to learn a disentangled latent representation based on the meme's image. The details of the visual representation learning module will be discussed in Section 3.1.4. A core element in the two learning modules is the process of disentangling the target information from the text and visual representations. Specifically, the latent text and visual representations are projected into a disentangled latent space $\mathcal{D}$, where each latent unit of the disentangled representation represents a probability for a certain category of hate (i.e., religion, gender and race, etc.). However, there is no direct supervision about the meme target (at the time of submission).

For a multimodal meme, disentangled targets from the textual modality and the visual modality should be consistent with each other. Therefore, we introduce a self-supervised matching loss, which constrains the disentangled visual representation to be similar to the disentangled text representation. Finally, the learned text and visual representations will be fed into a regression layer (the binary classification task can also be a regression task to predict hateful scores) to predict the likelihood of the meme being hateful. The classification process will be discussed in Section 3.1.5.

3.1.1 Problem Definition

We define the problem of hateful memes multimodal classification as follows: Given an image $\mathcal{I}$ and a piece of text $\mathcal{O}$ consisting of a sequence of words, a classification model will predict the label of the multimodal meme (*hateful* or *non-hateful*). This binary classification task can also be regarded as a regression task, where a model predicts a confidence score $y \in \mathbb{R}$ ranging from zero to one, indicating the likelihood of the meme being hateful. Specifically, the meme would be regarded as hateful if the predicted score is above a threshold λ; otherwise, the meme is predicted to be non-hateful.

3.1.2 Data Pre-processing

To improve our proposed method's reproducibility, we also provide an overview of the data pre-processing steps applied on the meme datasets. Specifically, the following steps were taken to pre-process the memes in the datasets:

- **Image resizing**: The datasets provided memes in all sizes. We resized the images proportionally to a minimum of 140 pixels and a maximum of 850 pixels. This ensures consistency of the visual input into our proposed model and baselines.
- **Text extraction and removal**: We extract and remove the text in the memes using open-source Python packages EasyOCR[1] and MMEditing[2]. The texts are removed from the memes to facilitate better entity and demographic detection.
- **Entity detection**: To augment the memes with relevant external information, we leverage Google Vision Web Entity Detection API[3] to detect and caption the entities in the cleaned image. The detected entities provide contextual information on the memes.
- **Demographic detection**: Often, hate speeches are targeted at groups based on demographic information such as race and gender, and such information serves as important contextual information in hateful meme classification. To augment the memes with demographic information, we utilized the FairFace classifier [81] to detect and classify the faces in the images, then mapped the label back to the person's bounding box with the largest overlapped area with the face. Note that the demographic information is only extracted when the meme contains human entities.

The pre-processed datasets serve as input for the training of our proposed model discussed in the next section.

3.1.3 Text Representation Learning

This module is designed to learn a disentangled latent representation of the textual information extracted from a meme. The input of the module is the concatenation

[1]https://github.com/JaidedAI/EasyOCR
[2]https://github.com/open-mmlab/mmediting
[3]https://cloud.google.com/vision/docs/detecting-web

of the text information output of the data pre-processing step. Specifically, we concatenated the text extracted from the meme, detected entities, and demographic information. Formally, we denote the concatenated text information as $\mathbf{O} = \{\mathbf{o}_j\}_{j=1}^{M}$, where $\mathbf{o}_j \in \mathbb{R}^{|\mathcal{V}|}$ is the one-hot vector representation for the j-th word in the text's word sequence, M is the length of the text, and $\mathcal{V}$ is the vocabulary.

Text Encoder. The concatenated text information $\mathbf{O}$ is first fed into a text encoder to generate latent text representations. Since the input text involves words from various domains such as religion, politics, or military, a powerful text encoder is required to capture the semantics in textual information. Bidirectional Encoder Representations from Transformers (BERT) [36], which has demonstrated its superiority in various NLP tasks, is an ideal text encoder for our task. We initialize the BERT with pre-trained weights and fine-tune it so that it generates textual representations aware of target entities. Using the BERT text encoder, we generate the textual representations as follows:

$$[\mathbf{s}, \mathbf{C}] = \text{BERT}([\mathbf{w}_{\text{[CLS]}}, \mathbf{O}]), \tag{3.1}$$

where $\mathbf{w}_{\text{[CLS]}} \in \mathbb{R}^{|\mathcal{V}|}$ denotes the one-hot representations for the "[CLS]" token, $[\cdot, \cdot]$ is the concatenation operation and $\mathbf{C} = \{\mathbf{c}_j\}_{j=1}^{M}$ is the set of textual representations, and $\mathbf{c}_j \in \mathbb{R}^u$ is the representation for the j-th word in the input text $\mathbf{O}$. Similar to [36], we utilize the representation of the "[CLS]" token as the sentence representation $\mathbf{s} \in \mathbb{R}^u$.

Text-target disentanglement. The latent text representation generated using the BERT encoder captures rich information on the meme's semantics. However, for our hateful meme classification task, we are interested in contextual information present in the latent text representation, specifically the targets of the hateful content. For example, in Figure 3.2, we aim to identify "gender" as the target entity of the hateful message. Therefore, we need to design a mechanism to disentangle the target information in the latent text representation s.

We first transform the the sentence representation s into a latent space using a linear projection layer:

$$\mathrm{s_p} = \mathbf{W}_{\mathrm{s}}\mathbf{s} + \mathbf{b}_{\mathrm{s}}, \tag{3.2}$$

where $\mathbf{W}_{\mathrm{s}} \in \mathbb{R}^{|\mathcal{D}| \times u}$ and $\mathbf{b}_{\mathrm{s}} \in \mathbb{R}^{|\mathcal{D}|}$ are parameters to be learnt.

The goal of latent space disentanglement is to minimize the overlap of information among latent units in the vector. There are many methods that perform latent space disentanglement using regularization terms to minimize the mutual information between latent units [24, 16, 149]. For our task, we aim to disentangle the projected text representation such that each unit in the latent vector represents a type of hateful meme targets. Noted that we assume that each meme is likely to concentrate on a certain category of hate (i.e., religion, gender, race, etc.) in most cases. To achieve this objective, we adopt a similar approach to [118], where we maximize the likelihood of the target present in the latent text representation while minimizing the likelihood for the absent targets. Such a discontinuous $\arg\max$ operation can be fulfilled by applying *Straight-Through Gumbel-Softmax* (STGS) function [76] over $\mathrm{s_p}$. Specifically, a continuous vector $\mathbf{z} \in \mathbb{R}^{|\mathcal{D}|}$ is first sampled from the Gumbel-Softmax distribution based on $\mathrm{s_p}$:

$$u_k \sim \text{Uniform}(0, 1), \tag{3.3}$$

$$g_k = -\text{log}(-\text{log}(u_k)), \tag{3.4}$$

$$z_k = \frac{\text{exp}(\text{log}(s_{\mathrm{p}}^k) + g_k)/\tau}{\sum_{k=1}^{||\mathcal{D}||} \text{exp}(\text{log}(s_{\mathrm{p}})^k + g_k)/\tau}, \tag{3.5}$$

where s_{p}^k is the k-th element in $\mathrm{s_p}$. In the forward pass, the STGS function then transforms the continuous vector sampled from the Gumbel-Softmax distribution into a one-hot vector [76]:

$$l_{\mathrm{s}}^k = \begin{cases} 1 & k = \arg\min_{m} z_m \\ 0 & else \end{cases} \tag{3.6}$$

Finally, the exploitation of STGS to generate the disentangled text representation can be simplified as:

$$\mathrm{l_s} = \text{Gumbel-Softmax}(\mathrm{s_p}), \tag{3.7}$$

where $\mathrm{l_s} = \{s_{\mathrm{p}}^k\}_{k=1}^{|\mathcal{D}|}$ as generated by Equation 3.3. For example, in Figure 3.2, the disentangled text representation would be a one-hot latent vector where 1 is assigned to the latent unit that represents 'woman' (i.e., target). The disentangled text representation $\mathrm{l_s}$ will be used in the self-supervised matching with the visual disentangled latent representation in section 3.1.4.

Via learning a disentangled text representation in the projected latent space, we update the text representation s through the back-propagation process, thus fine-tuning s to be more representative of the target information. We then use the updated s for hateful meme classification, as discussed in Section 3.1.5.

3.1.4 Visual Representation Learning

After learning the text representation, we focus on learning the disentangled latent representation in the visual modality. The input of this module is the image features pre-processed using Faster R-CNN [146]. Formally, we define the set of image features as $\mathbf{V} = \{\mathbf{v}_i\}_{i=1}^N$, where $\mathbf{v}_i \in \mathbb{R}^d$ is the feature for the i-th detected region using Faster R-CNN [146] and N is the number of detected region.

Attention-Based Image Encoder To enable better interaction between visual and textual modality, we adopted the multi-head attention proposed in [175] to learn an attended latent visual representation of the meme. Specifically, we leverage the textual representations **C** generated by BERT text encoder as guidance to attend the feature map **V** and generate the attended visual representation using the attention-based image encoder. The attended visual representation is computed as follow:

$$\mathbf{F}_t = \text{softmax}(\frac{\mathbf{W}_{\text{q},t}\mathbf{C}(\mathbf{W}_{\text{k},t}\mathbf{V})^T}{\sqrt{q}})\mathbf{W}_{\text{v},\mathbf{t}}\mathbf{V} \tag{3.8}$$

$$\tilde{\mathbf{F}} = \text{Concate}(\{\mathbf{F}_t\}_{t=1}^{q}) \tag{3.9}$$

$$\mathbf{G} = \mathbf{W}_{\text{f},1}(\text{Relu}(\mathbf{W}_{\text{f},2}\tilde{\mathbf{F}}) + \mathbf{b}_{\text{f},2}) + \mathbf{b}_{\text{f},1} \tag{3.10}$$

$$\tilde{\mathbf{V}} = \tilde{\mathbf{F}} + \mathbf{G} \tag{3.11}$$

$$\mathbf{v}_{\text{att}} = \sum_{m=1}^{M} \tilde{\mathbf{v}}_m, \tag{3.12}$$

where q denotes the number of times the dot-attention is computed in the multi-head attention. Specifically, the t-th attended image feature $\mathbf{F}_t \in \mathbb{R}^{\frac{u}{q}\times M}$ is generated as shown in Equation 3.8, where $\mathbf{W}_{\text{c},t} \in \mathbb{R}^{\frac{u}{q}\times u}$, $\mathbf{W}_{\text{k},t} \in \mathbb{R}^{\frac{u}{q}\times d}$ and $\mathbf{W}_{\text{v},t} \in \mathbb{R}^{\frac{u}{q}\times d}$ are parameters involved in the $t-th$ computation. The attended image features in different aspects are concatenated in row and generate $\tilde{\mathbf{F}} \in \mathbb{R}^{u\times M}$. Similar to [175], a residual connection is applied to the attended image features as illustrated in Equation 3.10 and 3.11. Finally, weighted average pool over the attended image features $\tilde{\mathbf{V}}$ results in the attended latent visual representation $\mathbf{v}_{\text{att}}$.

Visual-Target Disentanglement Although the attended visual representation is generated with an attention mechanism, there is no explicit guidance or supervision signal that forces the model to focus on the image regions that are more relevant to the contextual information (i.e., target entities of hateful memes). For instance, in Figure 3.2, the visual representation should ideally focus on the image region with the three women and to be aware that the focused region infers the "gender" as the target of the hateful meme. To make the visual representation more "target-aware", we design a latent space matching mechanism, which aims to disentangle the target information from the visual representation and constraint the disentangled visual representation to be consistent with the disentangled latent text representation.

Similar to the text-target disentanglement, we first project the visual representation $\mathbf{v}_{\text{att}}$ into the latent space with a linear projection layer:

$$\mathbf{v}_{\text{p}} = \mathbf{W}_{\text{a}}\mathbf{v}_{\text{att}} + \mathbf{b}_{\text{a}}, \tag{3.13}$$

where $\mathbf{W}_{\text{a}} \in \mathbb{R}^{|\mathcal{D}| \times u}$ and $\mathbf{b}_{\text{a}} \in \mathbb{R}^{|\mathcal{D}|}$ are parameters to be learnt.

To disentangle the target information in the visual representation, we introduce a matching loss that constraints $\mathbf{v}_{\text{p}}$ to be similar to the disentangled text representation $\mathbf{l}_{\text{s}}$:

$$\mathcal{L}_{\text{Match}} = \sum_{k=1}^{|\mathcal{D}|} l_{\text{s}}^{k} log(v_{\text{p}}^{k}) + (1 - l_{\text{s}}^{k}) log(1 - v_{\text{p}}^{k}), \tag{3.14}$$

where v_{p}^{k} is the k-th unit in the visual latent representation $\mathbf{v}_{\text{p}}$. The matching loss serves as a supervision signal to disentangle the target information in the latent visual representation. The underlying intuition is that the disentangled text representation $\mathbf{l}_{\text{s}}$ is trained to disentangle the target in the textual information, and constraining the disentangled visual representation to be similar to the disentangled text representation would enable the target latent unit to be maximized in disentangled visual representation. For example, in Figure 3.2, the matching loss will take guidance from the disentangled text representation and enforce the disentangled visual representation to maximize the latent unit representing "gender".

Similarly to the text representation learning, the visual representation $\mathbf{v}_{\text{att}}$ would be updated through the back-propagation process, fine-tuning $\mathbf{v}_{\text{att}}$ to be more representative of the target information. The updated $\mathbf{v}_{\text{att}}$ would be used for the hateful meme classification discussed in Section 3.1.5.

3.1.5 Classification

To perform hateful meme classification, we leverage a regression layer to generate a *hateful score*. Specifically, if the hateful score is above a threshold, it will be regarded as hateful, otherwise non-hate. By learning the disentangled textual and visual representation and minimizing the matching loss between them in the latent space, the sentence representation s and the attended image feature $\mathbf{v}_{\text{att}}$ are both fine-tuned to be more representative of the target information. Finally, we concatenate s and $\mathbf{v}_{\text{att}}$, and feed the concatenated representation to a regression layer for the score prediction:

$$y = \sigma(\mathbf{w}_{\text{r}}^T[\mathbf{s}, \mathbf{v}_{\text{att}}] + b_{\text{r}}), \tag{3.15}$$

where $\mathbf{w}_{\text{r}} \in \mathbb{R}^{2u}$ and $b_{\text{r}} \in \mathbb{R}$ are parameters to be learnt. Following [85], we set the threshold λ as 0.5: if the score y is above the threshold, it will be predicted as a hateful meme, otherwise, non-hate.

Loss Function. We optimize the following loss when training our model using mini-batch gradient descent:

$$\mathcal{L}_{\boldsymbol{\theta}} = \mathcal{L}_{\boldsymbol{\theta},\text{Predict}} + \mu\mathcal{L}_{\boldsymbol{\theta},\text{Match}}, \tag{3.16}$$

where $\boldsymbol{\theta}$ denotes parameters of the model, $\mathcal{L}_{\boldsymbol{\theta},\text{Match}}$ is the matching loss in the disentangled latent space, computed from the sum of matching loss over all training samples (see Equation 3.14); and $\mathcal{L}_{\boldsymbol{\theta},\text{Predict}}$ is the loss from prediction and μ is the hyper-parameter balancing the relative importance of both loss types. The prediction loss is defined as:

$$\mathcal{L}_{\boldsymbol{\theta},\text{Predict}} = \sum_{s=1}^{|\mathcal{T}|} \hat{y}_s log(y_s) + (1 - \hat{y}_s) log(1 - y_s), \tag{3.17}$$

where $\mathcal{T}$ is the training set and $\hat{y}_s$ is the ground-truth label and y_s is the predicted score of the s-th training sample.

3.2 Experiment Results

In this section, we will first describe the settings of experiments conducted to evaluate our DisMultiHate model. Next, we discuss the experiment results and evaluate how DisMultiHate fares against other state-of-the-art baselines. We also conduct case studies to qualitatively analyze DisMultiHate's ability to identify the targets of hateful memes. Finally, we perform error analysis and discuss the limitations of the

Dataset	train	validation	test
FHM	hateful (3050), non-hateful (5450)	hateful (250), non-hateful (250)	hateful (500), non-hateful (500)
MultiOFF	offensive (187), non-offensive (258)	offensive (58), non-offensive (91)	offensive (58), non-offensive (91)

Table 3.1: Distributions of FHM and MultiOFF datasets

DisMultiHate model.

3.2.1 Evaluation Setting

Dataset. We train and evaluate our proposed model on two popular and publicly-available hateful datasets: *Facebook hateful memes (FHM* and *MultiOff*. Table 3.1 shows the distributions of the datasets.

Evaluation Metrics. We adopt the evaluation metrics proposed in the hateful meme dataset papers [85, 165]. Specifically, for the evaluation on FHM dataset [85], we use Area Under the Receiver Operating Characteristic curve (AUROC) and accuracy score as the evaluation metrics. Suryawanshi et al. [165] had only reported the F1, precision, and recall on the hateful class when they proposed the MultiOFF dataset [165]. However, we noted that due to class imbalance in hate speech classification, most existing studies [44, 153] have preferred to use weight metrics to evaluate the classification task. Thus, we adopt weighted F1, weighted precision, and weighted recall as the evaluation metrics for the MultiOFF dataset.

Baselines. We benchmark DisMultiHate against the state-of-the-art multimodal methods that were evaluated on the FHM and MultiOFF datasets. We have also included an unimodal baseline for comparison. Specifically, we applied the pre-trained BERT [36] to extract text features from the meme's text and feed the extracted text features to a fully connected layer for classification.

Model	Acc.	AUROC
BERT (unimodal)	58.3	64.7
ViLBERT	62.2	71.1
VisualBERT	62.1	70.6
ViLBERT-CC	61.4	70.1
VisualBERT-COCO	65.1	74.0
ERNIE-VIL	69.0	78.7
UNITER	68.6	78.0
VILLNA	71.2	78.5
VL-BERT	71.4	78.8
DisMultiHate (w/o disentangle)	73.6	81.4
DisMultiHate	**75.8**	**82.8**

Table 3.2: Experimental results on FHM dataset.

3.2.2 Experimental Results

For FHM dataset, we compare with the four best performing multimodal models reported in the original dataset paper [85], namely: **ViLBERT** [117], **ViLBERT-CC** (i.e., ViLBERT pre-trained on Conceptual Captions [155]), **VisualBERT** [101], and **VisualBERT-COCO** (i.e., VisualBERT pre-trained on COCO [26]). There are many interesting solutions proposed by the Facebook hateful memes classification challenge participants [213, 211, 108, 205, 128, 207]. For our evaluation, we benchmark against the methods explored by the top-performing participant[4]. Specifically, we benchmark against the methods proposed in Zhu's exploration [213]: **ERNIE-Vil** [197], **UNITER** [210], **VILLNA** [47], and **VL-BERT** [162]. We have reproduced the model using the code[5] released in [213]. We also adopt the same data augmentation method proposed in [213] to enhance the models. Specifically, all the reproduced models are augmented with entity tags retrieved using Google Vision Web Entity Detection, and **VL-BERT** is also further enhanced with demographic information extracted using FairFace [81].

For MultiOFF dataset, we compare with the multimodal methods reported in the dataset paper [165], namely: **StackedLSTM+VGG16**, **BiLSTM+VGG16**, and **CNNText+VGG16**. For these multimodal baselines, the researchers first extract the

[4]https://ai.facebook.com/blog/hateful-memes-challenge-winners/
[5]https://github.com/HimariO/HatefulMemesChallenge

Model	F1	Precision	Recall
BERT (unimodal)	56.4	56.1	57.7
StackedLSTM+VGG16	46.3	37.3	61.1
BiLSTM+VGG16	48.0	48.6	58.4
CNNText+VGG16	46.3	37.3	61.1
ERNIE-VIL	53.1	54.3	63.7
UNITER	58.1	57.8	58.4
VILLNA	57.3	57.1	57.6
VL-BERT	58.9	59.5	58.5
DisMultiHate (w/o disentangle)	60.8	61.4	62.7
DisMultiHate	**64.6**	**64.5**	**65.1**

Table 3.3: Experimental results on MultiOFF dataset.

image features using VGG16 [158] pre-trained on the ImageNetdataset, and extract text features using various text encoders (e.g., BiLSTM). Subsequently, the extracted image and text features are concatenated before feeding into a classifier for hateful meme classification. As the MultiOFF dataset is relatively new, few studies have benchmarked on this dataset. Therefore, as additional baselines, we reproduced the methods proposed by Zhu [213] on the MultiOFF dataset.

We have also included a variant of the DisMultiHate, which performed the hateful meme classification without disentangling the target information. Interestingly, we observe that even without the target disentanglement module, DisMultiHate had outperformed the baselines, demonstrating the strength of our data pre-processing approach on augmenting the model with entity and demographic information. DisMultiHate without target disentanglement has also outperformed the VL-BERT model, which was also augmented with entity and demographics. A possible reason for the performance could be due to DisMultiHate's ability to learn better textual and visual representations for hateful meme classification. Specifically, the visual representation was attended with the textual information, thereby enhancing the visual features with some form of contextual information. Nevertheless, we noted that the performance of target disentanglement further improves the classification results, suggesting the importance of target information in the hateful meme classification task.

Table 3.2 and 3.3 show the experimental results on the FHM and MultiOFF datasets respectively. In both tables, the highest figures are highlighted in **bold**. We observed that DisMultiHate outperformed the state-of-the-art baselines in both datasets. DisMultiHate has significantly outperformed the baselines proposed in the original dataset papers. For instance, DisMultiHate has outperformed VisualBERT-COCO by more than 10% (Acc) on the FHM dataset and outperformed BiLSTM+VGG16 by more than 16% (F1) on the MultiOFF dataset. DisMultiHate has also outperformed the best baseline by 4% (Acc) and 5% (F1) on the FHM and MultiOFF, respectively. We noted that the multimodal methods had outperformed the BERT unimodal baselines in the FHM dataset. However, for the experiment on MultiOFF dataset, we observe that the BERT unimodal baseline is able to achieve competitive performance and outperformed the multimodal baselines proposed in the dataset paper [165]. A possible explanation could be the caption and text information in the MultiOFF memes already contain hateful content. Thus, the unimodal baseline using textual features is able to achieve good performance.

Ablation Study. We also conduct an ablation study to examine the usefulness of entity and demographic information augmented in our DisMultiHate method. Table 3.4 and 3.5 show the results of the ablation study on FHM and MultiOFF datasets, respectively. We observed that DisMultiHate model augmented with both entity and demographic information had yielded the best performance.

More interestingly, for the FHM dataset, we observed that without augmenting demographic information yields better performance than without augmenting entity information. However, a different observation is made for DisMultiHate performance on the MultiOFF dataset. Specifically, DisMultiHate not augmented with entity information yields better performance than without augmenting demographic information. The ablation study highlights that the model is highly amenable and adapts to different datasets with varying characteristics.

Model	Acc.	AUROC
DisMultiHate (w/o Entity)	60.6	68.2
DisMultiHate (w/o Demo)	72.8	80.8
DisMultiHate (w/o Entity+Demo)	62.0	70.3
DisMultiHate	75.8	82.8

Table 3.4: Ablation study on FHM dataset.

Model	F1	Precision	Recall
DisMultiHate (w/o Entity)	62.0	64.0	63.8
DisMultiHate (w/o Demo)	60.5	61.0	61.1
DisMultiHate (w/o Entity+Demo)	62.0	62.4	63.1
DisMultiHate	64.6	64.5	65.1

Table 3.5: Ablation study on MultiOFF dataset.

3.2.3 Case Study

The ability to disentangle target information in memes is a core contribution in our DisMultiHate model, and we aim to evaluate this aspect of the model qualitatively. Working towards this evaluation goal, we design an experiment to retrieve relevant memes for a given target query. Specifically, for a given text query (e.g., "woman"), we first generate its disentangled latent text representation, $\mathbf{l}_\mathrm{q}$, using the process described in Section 3.1.3. Next, we compute the cosine similarity between $\mathbf{l}_\mathrm{q}$ and the disentangled latent visual representation $\mathbf{v}_p$ of all memes in the FHM dataset. Finally, we retrieved the top k memes with the highest similarity scores with the given target query. Intuitively, if DisMultiHate model is able to disentangle the target information in memes, we should be able to infer the query target from the retrieved memes qualitatively. For example, given the text query "woman", we should expect the top k retrieved memes to include woman-related memes.

Table 3.6 shows the retrieved memes for a given target. Specifically, we retrieve the two most relevant hateful and non-hateful memes for the given target query. We can intuitively infer that the retrieved memes are relevant to the given query. For example, the retrieved memes for the query "Muslim Woman" are observed to contain Muslim women in hajib. Interestingly, for the query "Black Man", we observe that the second meme is retrieved even though the image is in black and

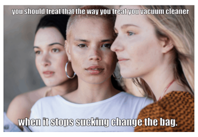
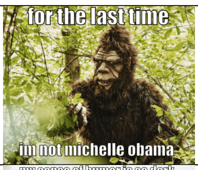
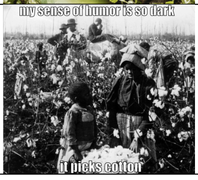
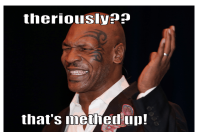
Target	**Hateful**		**Non-Hateful**	
Woman	 you should treat that the way you treat you vacuum cleaner when it stops sucking change the bag.			beauty shot of girlfriends focusing on different thoughts and ideas
Black Man				

Table 3.6: Retrieved memes from FHM dataset for a given target. The headers indicate the correctly predicted labels of the retrieved memes.

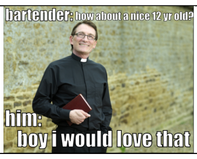
Meme		 bartender: how about a nice 12 yr old? him: boy i would love that		no, that's not his daughter.. that's his wife! yet the world is silent...
Actual Label	Non-Hateful	Non-Hateful	Hateful	Hateful
Pred. Label	Hateful	Hateful	Non-Hateful	Non-Hateful
Dis. Target	Woman	Catholics	Black Man	Muslim Woman

Table 3.7: Error analysis of wrongly classified memes from FHM dataset. Dis. target is for disentangled target.

white, and it is not obvious that there are African Americans in the image. However, DisMultiHate is still able to disentangle the "Black Man" target in the meme by using relevant textual information such as "*dark*" and "*pick cotton*" to infer contextual information on the slavery of African American. A similar observation is observed for the "Woman" target query, where the second meme does not contain any image of a woman but an ape. However, the second meme is also relevant to the target query as DisMultiHate disentangle the "Woman" target in the meme by using relevant textual information such as "*Michelle Obama*" to infer contextual information on insulting the individual's physical appearance(i.e., a woman) with a picture of an ape. In summary, the case studies presented in Table 3.6 has demonstrated DisMultiHate's ability to disentangle the target in memes using a combination of textual and visual information captured in the memes. Similar observations were also made for other potential hate speech target queries (e.g., Hispanic, Asian, transgender, etc.).

3.2.4 Error Analysis

Besides analyzing DisMultiHate quantitatively performance over the state-of-the-art baselines, we are also interested in examining the classification errors of DisMultiHate. Table 3.7 illustrates four selected examples of DisMultiHate's wrongly classified memes. For example, DisMultiHate has classified the first meme to be hateful when the actual label of the meme is non-hateful. A possible reason for this error could be the mention of the keyword "black" and the disentangled target being "woman", which misled the model to make a wrong prediction.

Our error analysis also reveals some issues with the FHM dataset. For instance, the second meme is annotated as non-hateful in the dataset. However, upon closer examination of the meme, we could infer some form of discrimination towards the Catholics and Priest, and our DisMultiHate has predicted the meme to be hateful. Another issue of the FHM dataset is the potential noise in the dataset. For example, DisMultiHate has wrongly classified the meme as non-hateful when the content is obviously communicating otherwise. We have checked the FHM dataset and found similar memes (i.e., a meme with a running black man) annotated as non-hateful.

DisMultiHate has also wrongly predicted the last meme to be non-hateful as none of the textual keyword, or image features provided the context information that it is hateful. Some form of advance reasoning would be required to understand the hateful context presented in this meme. We could explore adding advanced reasoning modules to classify such memes that require deeper reasoning for future work.

3.3 Conclusion

In this chapter, we proposed a novel framework, DisMultiHate, which learns and disentangles the representations of hate speech-related target entities, such as race and gender, in memes to improve the hateful content classification. We evaluated DisMultiHate on two publicly available datasets, and our extensive experiments have shown that DisMultiHate outperformed the state-of-the-art baselines. We have conducted case studies to empirically demonstrated DisMultiHate's ability to disentangle target information in the memes. We have also performed error analysis and discussed some of the limitations of the DisMultiHate model. We will incorporate a more advanced reasoning module in the model for future works and test the model on more hateful meme datasets. Through applying DisMultiHate to disentangle the target in hateful memes, we also hope to raise awareness of the vulnerable groups targeted in hate speeches in real-world datasets.

Chapter 4

Prompting a PT-LM to Acquire External Knowledge

As summarized in the previous chapter, various solutions were proposed for HMD. For instance, some studies have adopted pre-trained visual language models such as VilBERT [117] and VisualBERT [101] and fine-tune these models with the hateful meme classification task [108, 213, 211, 128, 176] (as illustrated in Figure 4.1(a)). Some tried to incorporate PTMs into task-specific HMD models [140, 89, 92]. Nevertheless, existing approaches still have limitations as understanding hateful memes may require additional contextual background knowledge. Consider the hateful meme example in Figure 4.1. The background knowledge that the pig is considered unclean by Muslims and is a sin to consume, is required to infer that the meme is hateful.

Recent studies have attempted to prompt PT-LMs and yield good performance for uni-modal NLP [150, 15, 151, 50]. Nevertheless, few works have attempted to prompt PT-LMs for multimodal tasks [194, 200, 56]. [193] has explored prompting GPT-3 model [15] for the visual question & answering task. However, the approach has limitations as large models such as GPT-3 are expensive to tune. In this chapter, we aim to address the research gaps and proposes a novel framework to leverage the implicit and unstructured knowledge in PT-LMs [173, 138] to improve hateful

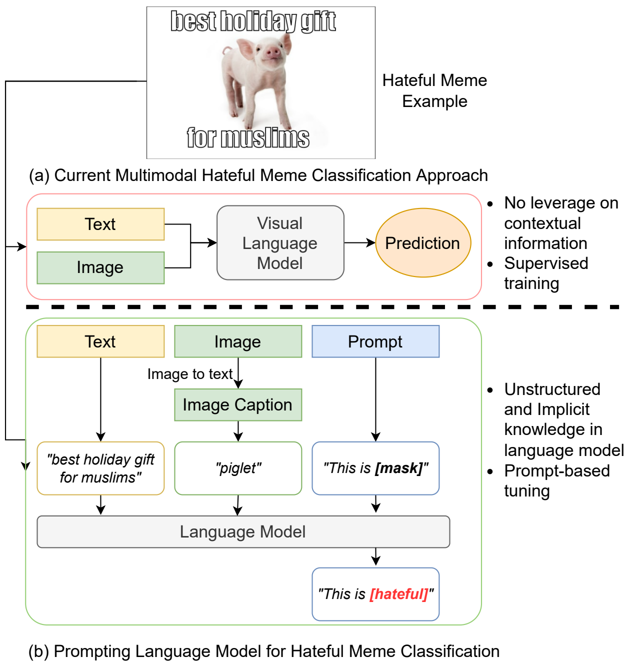

best holiday gift
for muslims
Hateful Meme Example
(a) Current Multimodal Hateful Meme Classification Approach
Text
Image
Visual Language Model
Prediction
No leverage on contextual information
Supervised training
Text
Image
Prompt
Image to text
Image Caption
"best holiday gift for muslims"
"piglet"
"This is [mask]"
Language Model
"This is [hateful]"
Unstructured and Implicit knowledge in language model
Prompt-based tuning
(b) Prompting Language Model for Hateful Meme Classification

Figure 4.1: Comparison between (a) fine-tuning visual language model approach and (b) prompt-based approach.

meme classification. Figure 4.1 illustrates the comparison between the existing multimodal hateful meme classification approach and our proposed prompt-based approach. Specifically, in our prompt-based approach, we first convert images into textual descriptions that a PT-LM can understand and design specific prompts to adapt and leverage the implicit knowledge in the PT-LM. Subsequently, given a meme and a prompt, the prompt-tuned PT-LM generates a textual output, indicating the predicted label of the input meme. The underlying intuition for the prompt-based approach is that PT-LMs will tap into the implicit and unstructured knowledge in their large-scale pre-training data to generate the continuation of the prompt, i.e., from "*this is* _" to "*this is* ***hateful***.".

The strategy of using PTMs in the proposed model is the hybrids of **Tuning-Single** and **Frozen-Single** as discussed in Section 1.3. Instead of directing tuning of a PT-LM, we leverage a PT-LM ($\mathcal{M}_{\text{PTM}}$) with a prompting mechanism ($\mathcal{M}_{\text{tsp}}$) for its stored implicit knowledge. Compared with direct fine-tuning, prompting adds no additional parameters, while is more similar to the pre-training objective (i.e., masked word prediction) of PT-LMs. Besides, to make images comprehensible to

PT-LMs, we further exploit a frozen PT-VLM to generate textual descriptions of images. Our contributions are: 1) identify a task-specific characteristic (i.e., need of external knowledge in HMD); 2) utilize a task-specific strategy of leveraging PT-LMs (i.e., prompting instead of direct fine-tuning); and 3) identify an intermediate step in need of help and convert it to acceptable goals of PTMs (i.e., use a frozen PTM for converting images to textual descriptions).

In the rest of the chapter, we first introduce the details of the proposed model in Section 4.1 and the experimental results in Section 4.2. To further understand the advantages and limitations of the proposed model, we provide a few visualization cases in Section 4.3. In the last part of the chapter, Section 4.4, we provide a conclusion of the work.

4.1 PromptHate Model

In order to prompt PT-LMs for HMD, we need to convert HMD into a masked language modeling task and converting images into textual descriptions. Therefore, firstly, we provide a formal definition about using masked language modeling for HMD in Section 4.1.1. Next, we describe how to conduct the image conversion in Section 4.1.2. In the rest section, we elaborate the work flow of PromptHate.

4.1.1 Problem Definition

We define the problem of multimodal hateful memes classification as follows: Given a meme with image $\mathcal{I}$ and text $\mathcal{O}$, a classification model will predict the label of the multimodal meme (*hateful* or *non-hateful*). Traditionally, this binary classification task requires models to predict a probability vector $\mathbf{y} \in \mathbb{R}^2$ over the two classes. Specifically, y_0 denotes the predicted probability that the meme is non-hateful while y_1 is for the probability that the meme is hateful. If $y_1 > y_0$, the meme is predicted as hateful, otherwise, non-hateful. In theframework, we transform the hateful meme classification task into a Masked Language Modelling (MLM) problem. Specifically, a PLM is prompted to replace the `[MASK]` token that represents the label of the meme (e.g., hateful or non-hateful). We discuss the prompting details in Section 4.1.3.

4.1.2 Image Captioning

To prompt PLMs for multimodal hateful meme classification, we first need to covert the meme's image into an acceptable textual input for PLMs. A common approach to extract the image's semantics and represent it with textual description is via image captioning [193, 56]. We first extract the text in the memes using open-source Python packages EasyOCR[1], followed by in-painting with MMEditing[2] to remove the text. We then apply a pre-trained image captioning model, ClipCap [127]. ClipCap is able to generate good quality captions for low-resolution web images. The generated captions tend to describe the dominant objects or events in the meme's image and we use these captions as inputs into the PromptHate model.

Besides captioning the image, we also leveraged Google Vision Web Entity Detection API[3] and pre-trained FairFace classifier [81] to extract the entities in the memes and the demographic information if the meme contains a person. The extracted entities and demographic information are used as supplementary information that will be combined with the image captions as input to the PLMs. Note that although the extracted supplementary information may capture key information about the meme, the contextual background knowledge is still absent in the image caption and supplementary information. For instance, with the utilization of entity information, we may identify a pig in the meme and extract the term "*Muslim*" from the meme text. However, the contextual knowledge that Muslims do not eat pork is absent in the supplementary information.

Figure 4.2 illustrates the architectural framework of our proposed PromptHate model. A key step in the PromptHate is the construction of a prompt, which consists of a positive demonstration (i.e., normal meme), a negative demonstration

[1]https://github.com/JaidedAI/EasyOCR
[2]https://github.com/open-mmlab/mmediting
[3]https://cloud.google.com/vision/docs/detecting-web

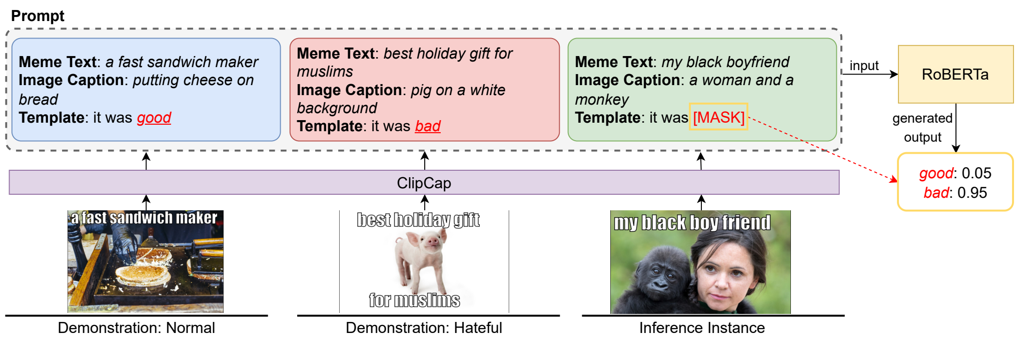


Figure 4.2: Overview of PromptHate Framework.

(i.e., hateful meme), and an inference instance (i.e., meme to be predicted). We first convert the three memes into meme texts and image descriptions using the data pre-processing steps described in Section 4.1.2. Subsequently, we construct templates, which are natural sentences that include label words for the individual memes. For instance, a normal demonstration meme will have a template "*the meme is **good***", while the hateful demonstration meme uses the template "*the meme is **bad***". The label word in template for the inference instance is replaced with a **[MASK]**, which the PLM (i.e., RoBERTa) is tasked to complete the sentence with "***good***" or "***bad***". Subsequent sections provide the technical details on prompting for the multimodal hateful meme classification task.

4.1.3 Prompting Hateful Meme

To guide the PLM in inferring the label word, we also provide *positive* and *negative* demonstrations to the PLM. The positive demonstration $\mathcal{S}^{\text{pos}}$ is generated as: $\mathcal{S}_1^{\text{pos}}\,\texttt{[SEP]}\,\mathcal{S}_2^{\text{pos}}\,\texttt{[SEP]}\,\mathcal{T}(\mathcal{W}_{\text{pos}})$, where $\mathcal{S}_1^{\text{pos}}$ and $\mathcal{S}_2^{\text{pos}}$ are meme texts and image descriptions respectively, `[SEP]` is the separation token in the language model $\mathcal{L}$, and $\mathcal{T}(\mathcal{W}_{\text{pos}})$ generates the positive label word $\mathcal{W}_{\text{pos}}$ into a sentence (e.g., "*this is **good***"). Similar approach is used for the generation of negative demonstration $\mathcal{S}^{\text{neg}}$ and inference instance $\mathcal{S}^{\text{infer}}$ by replacing $\mathcal{W}_{\text{pos}}$ with $\mathcal{W}_{\text{neg}}$ and `[MASK]`, respectively. Inspired by Gao et al. [50], we concatenate the demonstrations with the inference instance:

$$\mathcal{S} = \texttt{[START]}\mathcal{S}^{\text{infer}}\texttt{[SEP]}\mathcal{S}^{\text{pos}}\texttt{[SEP]}\mathcal{S}^{\text{neg}}\texttt{[END]} \tag{4.1}$$

where, $\mathcal{S}$ serves as the prompt fed into $\mathcal{L}$, and `[START]` and `[END]` are start and end tokens in $\mathcal{L}$.

4.1.4 Templates and Label Words

Recent studies have explored designing better prompts by developing automatic template generation and label word selection methods [50]. As PromptHate is the first study that adopted prompting for hateful meme classification, we adopt a simpler approach of prompting with manually defined label words and templates.

Labels are required to be mapped into individual words for prompt-based models. As shown in Figure 4.2, *good* is used as the label word for the positive class (non-hateful), while *bad* for the negative class (hateful). We have also analysed sets of other label words. The comparison of using different label words is discussed in Section 4.2.

The template in prompts can be viewed as the function $\mathcal{T}$, which maps the label word into a sentence. In PromptHate, we manually define the function $\mathcal{T}(\texttt{[WORD]}) \rightarrow$ *It was* `[WORD]`*.*. Specifically, if $\mathcal{T}$ receives $\mathcal{W}_{\text{pos}}$ as input, the output sentence should be "***It was*** $\boldsymbol{\mathcal{W}_{pos}}$***.***". Conversely, if $\mathcal{T}$ receives $\mathcal{W}_{\text{neg}}$ as input, the output sentence should be "***It was*** $\boldsymbol{\mathcal{W}_{neg}}$***.***".

4.1.5 Model Training and Prediction

For training, we feed the prompt $\mathcal{S}$ into $\mathcal{L}$ and obtain the probability of the masked word, $\mathbf{y} \in \mathbb{R}^2$ over label words:

$$y_0 = \text{P}(\texttt{[MASK]} = \mathcal{W}_{\text{pos}}|\mathcal{S}), \tag{4.2}$$

$$y_1 = \text{P}(\texttt{[MASK]} = \mathcal{W}_{\text{neg}}|\mathcal{S}). \tag{4.3}$$

The training loss is based on cross-entropy loss with the ground-truth label $\hat{\mathbf{y}}$:

$$\text{Loss} = y_0 log(\hat{y_0}) + y_1 log(\hat{y_1}), \tag{4.4}$$

and the loss will be used for updating parameters $\boldsymbol{\theta}$ in $\mathcal{L}$. Differing from standard fine-tuning PLMs by adding a task-specific classification head, prompt-based tuning does not have additional parameters beyond those in the PLMs, and the MLM task does not deviate from PLM's pre-training objectives.

For model prediction, we obtain the probability of the masked word over label words in the same manner. If $y_1 > y_0$, the meme will be predicted as hateful, otherwise, non-hateful.

4.1.6 Multi-Query prompthate-ensemble

Demonstrations in the prompt provide additional cues for the inference instance. Existing works carefully select demonstrations which are similar to the inference instance [193, 50]. Nevertheless, memes that are similar in visual or textual modality may be targeting different protected characteristics (e.g., race, religion, gender, etc.), and understanding the target in the hateful meme is critical to the classification task [92]. To address this concern, we adopt a multi-query prompthate-ensemble strategy to predict the inference instance using multiple pairs of demonstrations. Specifically, when we adopt a M-query ensemble, an inference instance will be predicted using M pairs of demonstrations.

The multi-query prompthate-ensemble will result in a set of prediction scores for the inference instance: $\{\mathbf{y}_m\}_{m=1}^{M}$, where $\mathbf{y}_m \in \mathbb{R}^2$ is the predicted scores with the m-th pair of demonstration. The final prediction will be the average over all predicted scores:

$$\mathbf{y}_{\text{final}} = \frac{1}{M} \sum_{m=1}^{M} \mathbf{y}_m. \tag{4.5}$$

Datasets	Train		Test	
	#Hate.	#Non-hate.	#Hate.	#Non-hate.
FHM	3,050	5,450	250	250
HarM	1,064	1,949	124	230

Table 4.1: Statistical summary of FHM and HarM.

4.2 Experiment Results

In this section, we first provide a brief introduction to the datasets and evaluation setting. Next, we present a set of experiments conducted to evaluate PromptHate's hateful meme classification performance. We also conduct studies to understand the effects of various prompt settings, and discuss the limitations of our model via error case studies.

4.2.1 Evaluation Settings

The HarM dataset was constructed with real COVID-19-related memes collected from Twitter. The memes are labeled with three classes: *very harmful*, *partially harmful*, and *harmless*. We combine the *very harmful* and *partially harmful memes* into hateful memes and regard harmless memes as non-hateful memes. The good performance on the HarM dataset also implies the generalization of the PromptHate to other anti-social memes besides hateful ones.

Evaluation Metrics. We adopt the evaluation metrics commonly used in existing hateful meme classification studies [85, 213, 211, 128, 176]: Area Under the Receiver Operating Characteristic curve (AUROC) and Accuracy (Acc). In order to report more reliable results, we measure the average performance of models under **ten** random seeds. All models use the same set of random seeds.

Baselines. We benchmark PromptHate against the state-of-the-art hateful meme classification models. Specifically, we compare with two types of baselines models: (a) uni-modal models that only use information from one modality (i.e., the meme text or the meme image); (b) multimodal models.

For uni-modal baselines, we consider a text-only model by fine-tuning pre-trained

BERT on the meme text for classification (**Text BERT**). We also apply an image-only model, which processes the meme image using Faster R-CNN [146] with ResNet-152 [62] before feeding the image representation into a classifer for hateful meme classification (**Image-Region**).

For multimodal baselines, we compare with the multimodal methods benchmarked in the original FHM dataset paper [85], namely: **Late Fusion**, **Concat BERT**, **MMBT-Region** [84], **ViLBERT CC** [117], **Visual BERT COCO** [101]. We also compare to the state-of-the-art hateful meme classification methods [4]: **CLIP BERT**, **MOMENTA** [140] and **DisMultiHate** [92]. CLIP BERT and MOMENTA are models leveraging image features generated by the CLIP model [143]. CLIP is pre-trained with web data, thus it is able to generalize well to HMD where images and texts are noisy. CLIP BERT uses CLIP as the visual encoder and BERT as the text encoder and feed the concatenation of features to a classifier for prediction. MOMENTA considers the global and local information in two modalities by modeling the deep multi-modal interactions. DisMultiHate disentangles target information from the meme to improve the hateful content classification.

As PromptHate prompts RoBERTa [114] for hateful meme classification, we also benchmark PromptHate against fine-tuning RoBERTa (**FT-RoBERTa**). Specifically, we concatenate the meme text and image descriptions as input to fine-tune RoBERTa, and the output representation is fed into a MLP layer for classification.

4.2.2 Experiment Results

Table 4.2 and 4.3 show the experimental results on FHM and HarM datasets, respectively. The standard deviations ($\pm$) of the ten runs are also reported, and the best results are **bold**. PromptHate outperforms the state-of-the-art baselines in both datasets. We have also computed the statistical differences between PromptHate and the best-performing baseline (i.e., DisMultiHate on FHM and FT-RoBERTa on HarM), and PromptHate's improvement over the baseline is found to be sta-

[4]Note that we use the code published by the author and re-run the model for ten rounds with different random seeds.

Model	AUC.	Acc.
Text BERT	$66.10_{\pm 0.55}$	$57.12_{\pm 0.49}$
Image-Region	$56.69_{\pm 1.05}$	$52.34_{\pm 1.39}$
Late Fusion	$66.34_{\pm 1.54}$	$59.14_{\pm 0.91}$
Concat BERT	$66.53_{\pm 0.75}$	$60.80_{\pm 0.98}$
MMBT-Region	$72.86_{\pm 0.64}$	$65.06_{\pm 1.76}$
Visual BERT COCO	$68.71_{\pm 1.02}$	$61.48_{\pm 1.19}$
ViLBERT CC	$73.05_{\pm 0.62}$	$64.70_{\pm 1.12}$
CLIP BERT	$66.97_{\pm 0.34}$	$58.28_{\pm 0.63}$
MOMENTA	$69.17_{\pm 4.71}$	$61.34_{\pm 4.89}$
DisMultiHate	$79.89_{\pm 1.71}$	$71.26_{\pm 1.66}$
FT-RoBERTa	$76.32_{\pm 6.45}$	$67.72_{\pm 6.20}$
PromptHate	**81.45**$_{\pm 0.74}$	**72.98**$_{\pm 1.09}$

Table 4.2: Experimental results of models on FHM.

Model	AUC.	Acc.
Text BERT	$81.39_{\pm 0.91}$	$75.68_{\pm 1.59}$
Image-Region	$76.46_{\pm 0.47}$	$73.05_{\pm 1.80}$
Late Fusion	$83.17_{\pm 1.25}$	$77.57_{\pm 0.96}$
Concat BERT	$83.21_{\pm 1.37}$	$77.82_{\pm 1.09}$
MMBT-Region	$85.48_{\pm 0.75}$	$79.83_{\pm 2.00}$
Visual BERT COCO	$80.46_{\pm 1.04}$	$75.31_{\pm 1.44}$
ViLBERT CC	$84.11_{\pm 0.88}$	$78.70_{\pm 1.17}$
CLIP BERT	$82.63_{\pm 1.20}$	$76.66_{\pm 1.02}$
MOMENTA	$86.32_{\pm 3.83}$	$80.48_{\pm 1.95}$
DisMultiHate	$86.39_{\pm 1.17}$	$81.24_{\pm 1.04}$
FT-RoBERTa	$89.26_{\pm 1.04}$	$82.32_{\pm 1.60}$
PromptHate	**90.96**$_{\pm 0.62}$	**84.47**$_{\pm 1.75}$

Table 4.3: Experimental results of models on HarM.

tistically significant (*p-value* < 0.05). Consistent with the existing studies, the multimodal approaches outperformed the unimodal baselines. More interestingly, we noted PromptHate's improvements over the multimodal baselines that fine-tuned PLMs and FT-RoBERTa, demonstrating the strength of the prompting approach for HMD . Specifically, the performance comparison of FT-RoBERTa and PromptHate suggests that the prompting approach can better leverage the implicit knowledge embedded in the PLM by adopting a masked language modeling training objective for the hateful meme classification.

We also observe differences in PromptHate's performance on the FHM and HarM datasets; the model yields better performance on HarM. Similar observations are made for the other models. We postulate that the performance differences are

Model	AUC.	Acc.
DisMultiHate	**79.89**$_{\pm 1.71}$	**71.26**$_{\pm 1.66}$
PromptHate-RB	79.17$_{\pm 0.67}$	70.56$_{\pm 0.73}$
DisMultiHate-BL	79.97$_{\pm 1.19}$	71.62$_{\pm 1.15}$
DisMultiHate-RL	78.56$_{\pm 0.94}$	71.10$_{\pm 1.58}$
PromptHate	**81.45**$_{\pm 0.74}$	**72.98**$_{\pm 1.09}$

Table 4.4: Experimental results of models with different sizes on FHM.

Model	AUC.	Acc.
DisMultiHate	86.39$_{\pm 1.17}$	81.24$_{\pm 1.04}$
PromptHate-RB	**89.20**$_{\pm 0.72}$	**83.70**$_{\pm 1.99}$
DisMultiHate-BL	85.38$_{\pm 1.13}$	80.71$_{\pm 1.45}$
DisMultiHate-RL	88.39$_{\pm 0.74}$	82.18$_{\pm 1.13}$
PromptHate	**90.96**$_{\pm 0.62}$	**84.47**$_{\pm 1.75}$

Table 4.5: Experimental results of models with different sizes on HarM.

likely due to the difficulty of the dataset. FHM contains hateful memes on multiple topics, while HarM mainly contains COVID-19-related hateful memes. Therefore, the models would have to be able to generalize better to perform well on the FHM dataset. We also highlight the high standard deviation in FT-RoBERTa's performance on FHM, suggesting FT-RoBERTa's instability and difficulty in generalizing well on the dataset.

As RoBERTa-large is regarded as a general LM for prompting [50, 151, 150], PromptHate with RoBERTa-large is three times in the scale compared with BERT-base related baselines. To further valid the effectiveness of prompting approach in HMD, we conduct the following experiments: 1) we replace the RoBERTa-large with RoBERTa-base in PromptHate (PromptHate-RB); 2) we replace the BERT-base in the baseline models with either RoBERTa-large (**-RL**) or BERT-large (**-BL**). Specifically, we choose the most powerful baseline, DisMultiHate, for analysis. Experimental results on FHM and HarM are summarized in Table 4.4 and Table 4.5 respectively, where each block includes models of similar sizes.

Unsurprisingly, replacing the RoBERTa-large with RoBERTa-base worsens PromptHate performance. However, we do observe that PromptHate-RB still outperforms DisMultiHate on the HarM dataset. On the FHM dataset, PromptHate-RB has performed slightly worse than DisMultiHate but depicted higher stability

Setting	FHM		HarM	
	AUC.	Acc.	AUC.	Acc.
PromptHate	81.45	72.98	90.96	84.47
w/o MLM	76.32	67.72	89.26	82.32
w/o Demo.	80.37	71.76	90.38	84.35

Table 4.6: Ablation study of PromptHate.

regarding the standard deviation. Interestingly, replacing the text encoder of DisMultiHate with a larger pre-trained LM does not outperform PromptHate on both datasets. From the experimental results, we observe that model size plays a critical role in PromptHate performance. Nevertheless, the experimental results have also demonstrated the effectiveness of our proposed prompting approach over state-of-the-art baselines.

4.2.3 Ablation Study

Table 4.6 shows the ablation analysis of PromptHate. We notice removing the MLM training objective decreases PromptHate's performance significantly. The MLM training objective is designed to align with the PLMs' training objectives. This plays a significant role in enabling PromptHate to better utilize the embedding implicit knowledge in the PLMs for hateful meme classification. Interestingly, we observe that PromptHate can perform well even without the demonstrations. Nevertheless, the effects of demonstrations in prompt-based model remains an open research topic, which requires further studies [125].

4.2.4 Prompt Engineering

Designing good prompts is essential to prompt-based models. In this section, we discuss how varying the prompts affect PromptHate's performance in hateful meme classification.

Engineering Label Words

Label words are individual words representing the labels used in prompt generation. We investigate the effects of replacing the prompts in PromptHate with different sets of label words. Specifically, we replace the label words in the prompt's positive and negative demonstrations in our experiments. Table 4.7 presents the results. For example, in the first row in Table 4.7, we use "*It was **normal***" for positive demonstrations (i.e., non-hateful memes), and "*It was **hate***" for negative demonstration (i.e., hateful memes). Intuitively we aim to examine how the label word's semantics affect hateful meme classification. For a more extensive investigation, we conduct the experiments on full training and few-shot setting, i.e., using only 10% of training instances.

Table 4.7 shows that different prompts can lead to substantial differences in performances. Specifically, label words aligned to the semantic classes are able to achieve better performance compared to the reverse mapping (i.e., the last row of each setting). Interestingly, the differences between the semantic class-aligned prompts and the reverse mapping are more significant in the few-shot setting. A possible reason could be in the few-shot setting, the PromptHate relies more on the label words' semantics to extract implicit knowledge for hateful meme classification. Thus, the label words with the aligned semantic class will provide better context in the prompt to improve hateful meme classification when there are insufficient observations in training instances. Conversely, when PromptHate is trained with enough instances, the representations of the label words are updated to be closer to the hateful meme classification task.

Prompt with Hateful Target Information

Existing studies have found that modeling target information (i.g., the victim of the hateful content) can help improve hateful meme classification [92]. Therefore, we explore the effect of explicitly including the target information in prompts.

The FHM and HarM datasets are annotated with target information. For our ex-

Setting	**Label Words**		**FHM**	
	Pos.	Neg.	AUC	Acc
full	Normal	Hate	81.21	71.74
	Benign	Offensive	81.58	72.70
	Good	Bad	81.45	72.98
	Hate	Normal	80.51	72.22
Few-Shot	Normal	Hate	69.21	63.88
	Benign	Offensive	68.91	63.68
	Good	Bad	69.30	63.76
	Hate	Normal	62.17	57.56

Table 4.7: PromptHate with various label words.

Model	**FHM**		**HarM**	
	AUC.	Acc.	AUC.	Acc.
w/o Target	81.45	72.98	90.96	84.47
w Target	81.10	71.44	89.00	82.97

Table 4.8: PromptHate without and with target.

perimental design, we change the prompt template: from "*It was [MASK].*" to "*It was [LABEL_MASK] targeting at [TARGET_MASK].*". For example, if it is a hateful meme targeting nationality, the template will be "*It was **bad** targeting at **nationality**.*" If the meme is non-hateful, the **[TARGET_MASK]** will be replaced with **nobody**. During model training, we model the loss from prediction of **[LABEL_MASK]** in the inference instance.

Table 4.8 shows the results of the PromptHate performance with and without target information. We observe marginal differences in performance after modelling target information in prompts. A possible reason may be that learning to extract targets in memes adds auxiliary burden to the model. To better utilize target information, a more sophisticated strategy may be needed than the current simple approach.

In Table 4.9, we visualize PromptHate's prediction results on sample FHM memes. Incorrect predictions are labelled in red while the pie chart presents the distributions of the predicted target (i.e., **[TARGET_MASK]**) per meme. PromptHate with target information is observed to correctly predict the targets in the hateful meme even when it incorrectly classifies the memes (e.g third meme targeting religion). The right-most meme contains a racial slur 'Kenyan skidmark' and seems to have been annotated wrongly as non-hateful. Interestingly, PromptHate with target information indicates it as hateful and targeting race.

The target distributions can improve PromptHate's interpretability. However, the incorrect class prediction also highlights the difficulty of hateful meme classification. The task may require more than target comprehension to achieve good performance.

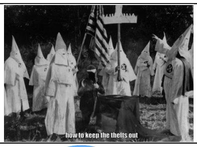
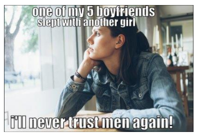
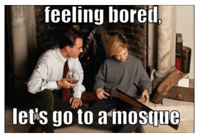
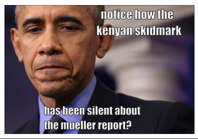
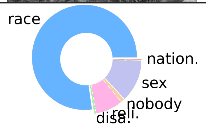
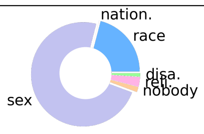
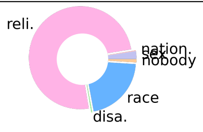
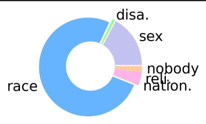
Meme		 one of my 5 boyfriends slept with another girl / i'll never trust men again!		 notice how the kenyan skidmark / has been silent about the mueller report?
Target Distributions				
w Target.	Hateful	Hateful	Non-Hateful	Hateful
w/o Target	Non-hateful	Non-hateful	Hateful	Non-hateful
Ground Truth	Hateful (race)	Hateful (sex)	Hateful (religion)	Non-Hateful?

Table 4.9: Example predictions of PromptHate with and without target information. Incorrect prediction in red. Ground truth for the right-most meme is questionable.

.

4.3 Qualitative Analysis

Besides analyzing PromptHate's quantitative performance, we also examine its classification errors. Table 4.10 illustrates three selected PromptHate's incorrect predictions.

From the examples, we notice that the captions generally describe the contents of images. However, it may ignore some essential attributes for HMD. For instance, the captions are unable to capture important information such as "*Jesus*". This missing information is supplemented by the augmented image tags (i.e., the entities and demographic of memes). Nevertheless, we also observed that even after augmentation with additional descriptions for the images, PromptHate still makes incorrect predictions for these memes.

There could be multiple reasons for the incorrect predictions. Firstly, the pre-

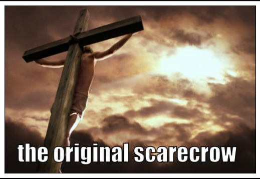
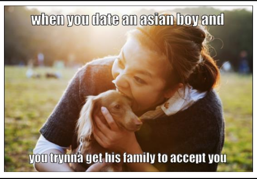
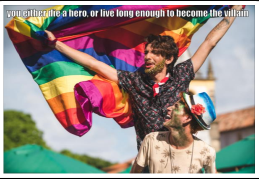
Meme		 when you date an asian boy and you trynna get his family to accept you	
Ground Truth	Hateful (religion)	Hateful (race)	Non-Hateful
Prediction	Non-hateful	Non-hateful	Hateful
Meme text	the original scare-crow	when you date an asian boy and you trynna get his family to accept you	you either die a hero, or live long enough to become the villain
Captions	builder crucified on the cross	portrait of a young woman with her pet dog	a man dressed as a rainbow holding a flag and dancing in the crowd
Entity	Crucifix Life, Resurrection of Jesus, Spiritual death, jesus died in the cross	none	Rainbow flag, Flag bisexual, Transgender flags, Bisexuality
Demographics	None	Black female	Latino Hispanic Male

Table 4.10: Error analysis of wrongly predicted memes. Incorrect prediction in red

sented information may still lack adequate context. For instance, in the second meme, the "biting" or "eating" action is missing from the captions and the addition image description. Thus, PromptHate lacks the context that the meme is ridiculing Asians' "dog-eating" behaviour, and is hateful. Secondly, there could be biases learned by the model. For instance, PromptHate may predict the right-most meme to be hateful because of the rainbow flag, an icon for the LGBT community. This icon may be heavily used by memes attacking the LGBT community. Lastly, even more advanced reasoning may be required to understand the hateful context in certain meme. In the first case, PromptHate is unable to reason that the meme implies that Jesus is merely an object hung up to scare away birds, which leads to the hatefulness of the meme. The detection of the hateful meme requires deep multi-modal reasoning.

4.4 Conclusion

We have proposed PromptHate, a simple yet effective multimodal prompt-based framework that prompts PT-LMs for hateful meme classification. Our evaluations on two publicly available datasets have shown PromptHate to outperform state-of-the-art baselines. We have conducted fine-grained analyses and case studies on various prompt settings and demonstrated the effectiveness of the prompts on hateful meme classification. We have also performed error analysis and discussed some limitations of the PromptHate model. For future work, we will explore strategies for selecting better demonstrations for PromptHate and add in reasoning modules to improve PromptHate's utilization of the implicit knowledge in the PLMs.

Chapter 5

Frozen PT-VLMs with Probing-based Captioning for Hateful-content Related Descriptions

In the previous chapter, we introduced PromptHate, which leverages knowledge in PT-LMs to facilitate HMD. As PT-LMs are inherently textual, it involves the conversion for meme images to textual descriptions. By converting all input information into texts, it can prompt a PT-LM along with two demonstrative examples to predict whether or not the input is hateful by utilizing the rich background knowledge in the PT-LM. Although PromptHate achieves state-of-the-art performance, it is significantly affected by the quality of image captions, as shown in Table 5.1. Image captions that are merely generic descriptions of images may omit crucial details [92, 213], such as the race and gender of people, which are essential for hateful content detection. But with additional image tags, such as entities found in the images and demographic information about the people in the images, the same model can be significantly improved, as shown in Table 5.1. However, generating these additional image tags is laborious and costly. For instance, entity extraction is usually conducted with the Google Vision Web Entity Detection API [1], which is a

[1]https://cloud.google.com/vision/docs/detecting-web

Model	Performance	
	AUC	Acc.
PromptHate (w/o)	76.76	67.28
PromptHate	81.45	72.98

Table 5.1: Impact on detection performances on the FHM dataset [85] from image captions. (w/o) denotes models without additional entity and demographic information.

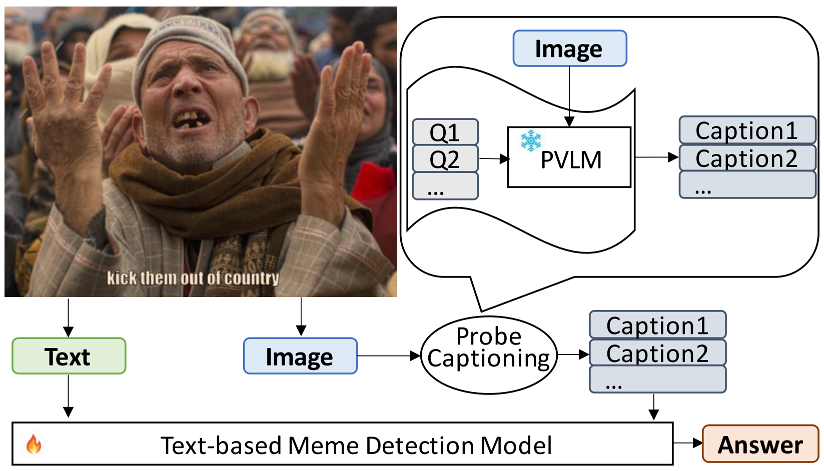


Figure 5.1: The proposed probe-captioning approach. We prompt frozen PT-VLMs via visual question answering to generate hateful content-related image captions.

paid service. Ideally, we would like to find a more affordable way to obtain entity and demographic information from the images that is critical for hateful content detection.

PromptHate suffers from information loss during image conversion. The other line of research harnesses the power of PT-VLMs such as VisualBERT [101] and ViLBERT [117], without converting images to discrete text tokens. A common approach is to fine-tune PT-VLMs with task-specific data [108, 213, 128, 176, 63]. However, it is less feasible to fine-tune the larger models such as BLIP-2 [100] and Flamingo [2] on meme detection because there are billions of trainable parameters. Therefore, computationally feasible solutions other than direct fine-tuning are needed to leverage large PT-VLMs in facilitating HMD.

Both above-mentioned approaches have their pros and cons. In this chapter, we combine the ideas from these two approaches and design a HMD method that leverages the power of a frozen PT-VLM to complement the unimodal approach of

PromptHate. Specifically, we use a set of "probing" questions to query a PT-VLM (BLIP-2 [100] in our experiments) for information related to common vulnerable targets in hateful content. The answers obtained from the probing questions will be treated as image captions (denoted as **Pro-Cap**) and used as input to a trainable HMD model. Figure 5.1 illustrates the overall workflow of the method. We refer to the step of using probing questions to generate the captions as *probing-based captioning*.

Our proposed method fills existing research gaps by: 1) Leverage a PT-VLM without any adaptation or fine-tuning, thereby reducing computational cost; 2) Instead of explicitly obtaining additional image tags with costly APIs, we utilize the frozen PT-VLM to generate captions that contain information useful for HMD. To the best of our knowledge, this is the first work that to leverage PT-VLMs in a zero-shot manner through question answering to assist in HMD. To further validate our method, we test the effect of the generated Pro-Cap on both PromptHate[20] and a BERT-based[36] HMD model.

Based on the experimental results, we observe that PromptHate with Pro-Cap (denoted as Pro-Cap$_{\text{PromptHate}}$) significantly surpasses the original PromptHate without additional image tags (i.e., about 4, 6, and 3 percentage points of absolute performance improvement on FHM [85], MAMI [42], and HarM [140] respectively). Pro-Cap$_{\text{PromptHate}}$ also achieves comparable results with PromptHate with additional image tags, indicating that probing-based captioning can be a more affordable way of obtaining image entities or demographic information. Case studies further show that Pro-Cap offers essential image details for hateful content detection, enhancing the explainability of models to some extent. Meanwhile, Pro-Cap$_{\text{BERT}}$ clearly surpasses multimodal BERT-based models of similar sizes (i.e., about 7 percentage points of absolute improvement with VisualBERT on FHM [85]), proving the generalization of the probing-based captioning method.

The Pro-Cap model falls under the category of **Frozen-Single** regarding the utilization of PTMs, as mentioned in Section 1.3. Specifically, it identify a sub-step (the conversion of images to textual descriptions) in PromptHate in need of improvement, as generic image captioning overlooks essential details for hateful content detection. It converts the need to acceptable objectives of PT-VLMs with a probing captioning technique so that frozen PT-VLMs ($\mathcal{M}_{\text{PTM}}^{\text{frozen}}$) can be used for generating hateful content-related descriptions without any adaptation. Later the generated descriptions from $\mathcal{M}_{\text{PTM}}^{\text{frozen}}$ will be incorporated with $\mathcal{M}^{\text{tune}}$ for supervised training. The contributions are two-folds: identifying the step in need of improvement and converting the step into pre-training objectives of PTMs.

In Section 5.1, we elaborate the details of the proposed method. The quantitative and qualitative results are presented in Section 5.2 and Section 5.3, respectively. Finally, we close the chapter with a conclusion.

5.1 Pro-Cap Model

5.1.1 Overview

Recall that the key idea of our method is to elicit image details that are critical for hateful content detection, such as the gender and race of the people in the image. Because these details are not always included in automatically generated image captions, we propose relying on VQA to obtain such critical information, where the questions are carefully curated to elicit demographic and other relevant information. We opt to use zero-shot VQA with a frozen PT-VLM because (1) for the intended type of questions, we do not have any VQA training data to train our own model, and (2) recent work has demonstrated promising performance of zero-shot VQA.

Specifically, we prompt the PT-VLM with K *probing questions* and regard the set of K answers from the PT-VLM as image captions, which we refer to as **Pro-Cap**. We then combine the original text $\mathcal{T}$ with Pro-Cap as input to a HMD model. We experiment with two alternative text-based HMD models: one based on BERT encoding, and the other based on our previously proposed PromptHate.

In the rest of this section, we first present the details of how we design our VQA questions to elicit the most critical details of an image for HMD. We then explain how the generated Pro-Cap is used by two alternative text-based HMD models.

Focus	**Questions**
Content	what is shown in the image?
Race	What is the race of the person in the image?
Gender	What is the gender of the person in the image?
Religion	What is the religion of the person in the image?
Nationality	Which country does the person in the image come from?
Disability	Are there disabled people in the image?
Animal	What animal is in the image?
Val Person	Is there a person in the image?
Val Animal	Is there an animal in the image?

Table 5.2: Details of questions prompting PT-VLMs. The first block of the question asks about the content of the image; questions in the second block ask about commonly seen vulnerable targets in hateful contents; the last block questions validate the existence of persons and animals.

5.1.2 Design of VQA Questions

We leverage PT-VLMs for zero-shot VQA to generate Pro-Cap as image captions. We want Pro-Cap to provide not only a general description of the image but also details critical for HMD. To obtain a general caption of the image, we design the first probing question to inquire about the generic content of the image, as shown in Table 5.2. However, such generic captions may be insufficient for HMD as hateful content usually targets persons or groups with specific characteristics, such as race, gender, or religion [42, 85]. Additionally, previous studies have shown that augmenting image representations with entities found in the image or demographic

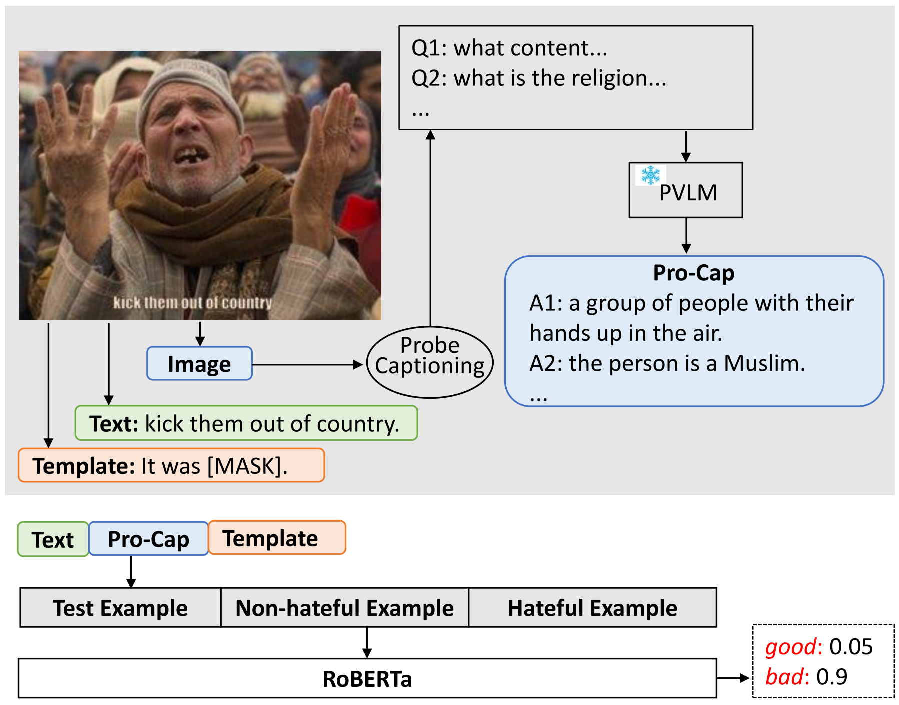


Figure 5.2: An overview of the PromptHate model and how pro-cap is used in PromptHate.

information of people in the image significantly aids HMD [213, 92]. Such details may be missing in generic image captions. Therefore, we design additional questions that aim to bring out information central to hateful content. This aligns the generated image captions more closely with the goal of HMD. Specifically, the high-level idea is to ask questions about common vulnerable targets of hateful content. Inspired by [124], which categorizes the targets of hateful memes into *Religion*, *Race*, *Gender*, *Nationality*, and *Disability*, we ask questions about these five types of targets. For example, to generate image captions that indicate the race of the people in an image, we can ask the following question: *what is the race of persons in the image?* We list the five questions designed for these five types of targets in Table 5.2. Additionally, we observe that some animals, such as pigs, are often depicted in hateful memes, frequently as a means to annoy Muslims. With this consideration, we also design a question asking about the presence of animals in the image.

In [33], the author claimed that PT-VLMs may hallucinate non-existent objects. For example, even when there is nobody in an image, PT-VLMs may generate an answer about race in response to the question *what is the race of the person in the* *image?*. To prevent such misleading information, we use two validation questions. Specifically, we inquire about the existence of persons and animals. Only when the PT-VLM responds that a person or an animal exists will we include in the Pro-Cap the answers to those person-related or animal-related questions. For instance, if the answer to the question validating the existence of people indicates that nobody is present, we will ignore all answers from questions asking about *religion*, *race*, *gender*, *nationality*, and *disability*.

We use $\mathcal{C}$ to represent the concatenation of the answers to the probing questions that are finally included as part of the Pro-Cap based on the validation results. We will then concatenate $\mathcal{T}$ and $\mathcal{C}$ together as input to a purely text-based hateful meme classification model, as shown at the bottom of Figure 5.1.

5.1.3 BERT-based Detection Model

We now introduce the first of the two alternative hateful meme classification models, which is based on BERT [36]. We first feed the concatenation of the meme text $\mathcal{T}$ and the Pro-Cap $\mathcal{C}$ into the BERT model to generate a vector $\mathbf{r} \in \mathbb{R}^d$:

$$\mathbf{r} = \text{BERT}([\mathcal{T}, \mathcal{C}]),$$

where $[\cdot, \cdot]$ represents concatenation. Next, we feed the sentence representation $\mathbf{r}$ into a linear layer for hateful meme classification:

$$\mathbf{s} = \text{Sigmoid}(\mathbf{W}^{\text{T}}\mathbf{r} + \mathbf{b}),$$

where $\mathbf{W} \in \mathbb{R}^{d \times 2}$ and $\mathbf{b}^2$ are learnable parameters.

5.1.4 PromptHate for Hateful Meme Detection

Next, we briefly recall PromptHate [20], which employs a prompt-based method to classify memes (More details in Chapter 4). PromptHate was developed to better leverage contextual background knowledge by prompting language models. Given a test meme, PromptHate first uses an image captioning model to obtain generic image captions. It then concatenates the meme text, the image captions, and a prompt template into $\mathcal{S}$: *It was* `[MASK]` **.**, to prompt a language model (LM) to predict whether the meme is hateful. Specifically, it compares the probability of the language model predicting `[MASK]` to be a positive word (e.g., *good*) given the context, versus the probability of predicting a negative word (e.g., *bad*). The approach also includes one positive and one negative example in the context, and `[MASK]` will be replaced by their respective label words. An overview of PromptHate is shown in Figure 5.2. For further details, please refer to [20].

In [20], PromptHate utilizes ClipCap [127] to generate image captions. In this work, we replace this with Pro-Cap $\mathcal{C}$. We then represent every meme $\mathcal{O}$ as $\mathcal{O} = [\mathcal{T}, \mathcal{C}, \mathcal{S}]$. With these inputs, the language models (LMs), for instance, RoBERTa [114], generate confidence scores for the masked word over their vocabulary space, $\mathcal{V}$:

$$\mathbf{p} = \text{Sigmoid}(\text{LM}([\mathcal{O}_{\text{test}}, \mathcal{O}_{\text{non-hate}}, \mathcal{O}_{\text{hate}}])),$$

where $\mathbf{p} \in \mathbb{R}^{|\mathcal{V}|}$. We extract the score for the label words as the prediction:

$$s_0 = p_i,\ \mathcal{V}_i = \mathcal{W}_{\text{pos}}, \tag{5.1}$$

$$s_1 = p_j,\ \mathcal{V}_j = \mathcal{W}_{\text{neg}}. \tag{5.2}$$

5.1.5 Model Training and Prediction

We denote the ground-truth label of a meme as $\hat{\mathbf{y}} \in \mathbb{R}^2$. If the meme is annotated as *non-hateful*, $\hat{y}_0$ will be 1 while $\hat{y}_1$ will be 0, otherwise, $\hat{y} = [0, 1]$. The binary cross-entropy loss is applied for model training:

$$\text{Loss} = -(\hat{y}_0 * log(s_0) + \hat{y}_1 * log(s_1)).$$

For model prediction, if $s_0 > s_1$, the meme will be predicted as non-hateful, otherwise, hateful.

5.2 Experiment Results

In this section, we first introduce our evaluation datasets, metrics and implementation details. Next, we introduce the baselines for comparison. Finally, we conduct qualitative analysis with case studies and error analysis to better understand the advantages and limitations of our method.

5.2.1 Experiment Settings

Evaluation Datasets. We test our proposed method on benchmarks for HMD. We evaluate our method on three datasets to better illustrate the generalization and stability of our approach. Table 5.3 presents the statistics of these datasets.

The *Facebook Hateful Meme* dataset (**FHM**) [85] was constructed by Facebook. It contains synthetic memes with added confounders such that unimodal information is insufficient for detection and deep multimodal reasoning is required. The FHM dataset contains hateful memes targeting various vulnerable groups in categories including *Religion*, *Race*, *Gender*, *Nationality*, and *Disability*. As the labels of the test split of FHM are not available, we performs evaluation on its *dev-seen* split.

Different from FHM, the *Multimedia Automatic Misogyny Identification* (**MAMI**) [42] dataset focuses on a particular type of hateful memes, namely, those targeting women. Performance on MAMI therefore reflects the capability of HMD methods for female victims.

To test our method's generalization capability, we also consider a harmful meme detection dataset, **HarM** [139]. HarM contains memes related to COVID-19, which are classified into three categories: *harmless*, *partially harmful*, and *very harmful*. We merge *partially harmful* and *harmful* into one category. Because hateful content is always regarded as harmful, we use this dataset to test the capability of generalization of our proposed method from HMD to harmful meme detection.

Datasets	Train		Test	
	#Hate.	#Non-hate.	#Hate.	#Non-hate.
FHM	3,019	5,481	247	253
HarM	1,064	1,949	124	230
MAMI	5,004	4,995	500	495

Table 5.3: Statistical distributions of datasets used for evaluation.

Evaluation Metrics. HMD is a binary classification task. In addition to detection accuracy, we also compute the Area Under the Receiver Operating Characteristics curve (AUCROC) used in prior work [108, 213, 92, 20]. We conduct experiments with **ten** random seeds and report the average performance and standard deviation. All models use the same set of random seeds.

Implementation Details. Given a meme image, we first detect the meme text with the open-source Easy-OCR tool [2] and then in-paint over the detected texts. To generate the answers to VQA questions, we prompt BLIP-2 [100], specifically the FlanT5$_{\text{XL}}$ version. We then insert the generated image captions into two text-based HMD models, i.e., the BERT-based model and the PromptHate model. For the BERT-based model, to avoid overfitting, we add a dropout rate of 0.4 to the classification layer. We use a learning rate of $2e-5$ and a batch size of 64. For PromptHate, we train the model with a batch size of 16 and empirically set the learning rate to $1.3e-5$ on FHM and $1e-5$ on the other two datasets [50]. We optimize both models with the AdamW optimizer [115] and implement them in PyTorch. Due to space limit, we provide more details (i.e., computation costs and model sizes) in Appendix B.4.

5.2.2 Baselines

We compare our method against both unimodal and multimodal models to demonstrate the effectiveness of the proposed method, where we regard models receiving information from one modality (i.e., the meme text or the meme image only) as unimodal models. Note that because Pro-Cap already contains image information,

[2]https://github.com/JaidedAI/EasyOCR

even if Pro-Cap is input into a unimodal BERT, the model is not considered to be unimodal.

For the unimodal models, we consider a text-only and an image-only model. For the text-only model, we fine-tune a pre-trained BERT model [36] based on the meme text only for meme classification, which we represent as **Text-BERT**. For the image-only model, we first extract object-level image features with an off-the-shelf feature extractor, Faster-RCNN [146], which is trained for object detection. We then perform average pooling over object features and feed the resulting vector into a classification layer. We use **Image-Region** to denote the image-only model.

For multimodal models, we categorize them into two groups: 1) fine-tuning generic multimodal models that are proposed to conduct different multimodal tasks; 2) models specifically designed for HMD. For the first type of multimodal models, we firstly consider the **MMBT-Region** model [84], which is a widely used multimodal baseline in hateful meem detection [85, 20, 140] and the model has not been pre-trained with multimodal data. Secondly, we consider several multimodal PTMs, such as VisualBERT [101] pre-trained on MS-COCO [107] (**VisualBERT COCO**) and ViLBERT pre-trained on Conceptual Captions [155] (**ViLBERT CC**). Some recently released powerful PTMs are also included such as the *Align before Fusion* model [98] (**ALBEF**) and the *Bootstrapping Language-Image Pre-training* model [99] (**BLIP**). For the second category of baselines which are designed for the meme detection task, we consider the models listed below. The **CLIP-BERT** model [140] leverages the CLIP model [143] to deal with noisy meme images, uses pre-trained BERT [36] for representing meme text, and fuses them with concatenation. The **MOMENTA** model [140] designed both local and global multimodal fusion mechanisms to exploit multimodal interactions for HMD. Note that the MOMENTA model is designed to leverage augmented image tags (the detected image entities). **DisMultiHate** [92] disentangles target information from memes as targets are essential for identifying hateful content. The **PromptHate** model [20] is what we discussed in Chapter 4.

Dataset	**FHM**		**MAMI**		**HarM**	
Model	**AUC.**	**Acc.**	**AUC.**	**Acc.**	**AUC.**	**Acc.**
Text BERT	$66.10_{\pm 0.55}$	$57.12_{\pm 0.49}$	$74.48_{\pm 0.60}$	$67.37_{\pm 0.57}$	$81.39_{\pm 0.91}$	$75.68_{\pm 1.59}$
Image-Region	$56.69_{\pm 1.05}$	$52.34_{\pm 1.39}$	$70.20_{\pm 0.63}$	$64.18_{\pm 0.81}$	$76.46_{\pm 0.47}$	$73.05_{\pm 1.80}$
VisualBERT COCO	$68.71_{\pm 1.02}$	$61.48_{\pm 1.19}$	$78.71_{\pm 0.59}$	$71.06_{\pm 0.94}$	$80.46_{\pm 1.04}$	$75.31_{\pm 1.44}$
ViLBERT CC	$73.05_{\pm 0.62}$	$64.70_{\pm 1.12}$	$77.71_{\pm 1.20}$	$69.48_{\pm 1.00}$	$84.11_{\pm 0.88}$	$78.70_{\pm 1.17}$
MMBT-Region	$72.86_{\pm 0.64}$	$65.06_{\pm 1.76}$	$79.17_{\pm 0.91}$	$70.46_{\pm 0.76}$	$85.48_{\pm 0.75}$	$79.83_{\pm 2.00}$
CLIP-BERT	$66.97_{\pm 0.34}$	$58.28_{\pm 0.63}$	$77.66_{\pm 0.64}$	$68.44_{\pm 1.07}$	$82.63_{\pm 3.83}$	$80.48_{\pm 1.95}$
DisMultiHate	$69.11_{\pm 0.84}$	$62.42_{\pm 0.72}$	$78.21_{\pm 0.61}$	$70.58_{\pm 1.13}$	$83.69_{\pm 1.33}$	$78.05_{\pm 0.73}$
PromptHate	$76.76_{\pm 0.95}$	$67.82_{\pm 1.23}$	$76.21_{\pm 1.05}$	$68.08_{\pm 0.58}$	$87.51_{\pm 0.74}$	$79.38_{\pm 1.72}$
BLIP	$76.80_{\pm 2.37}$	$69.20_{\pm 1.84}$	$80.59_{\pm 0.87}$	$71.84_{\pm 1.11}$	$87.09_{\pm 1.46}$	$81.81_{\pm 1.74}$
ALBEF	$79.40_{\pm 0.53}$	$70.58_{\pm 0.50}$	$\mathbf{83.24}_{\pm 0.93}$	$72.77_{\pm 1.00}$	$85.49_{\pm 1.23}$	$80.99_{\pm 0.80}$
Pro-Cap$_{\text{BERT}}$	$77.50_{\pm 0.58}$	$68.14_{\pm 0.64}$	$79.62_{\pm 0.91}$	$71.06_{\pm 0.88}$	$89.04_{\pm 1.00}$	$82.06_{\pm 1.92}$
Pro-Cap$_{\text{PromptHate}}$	$\mathbf{80.87}_{\pm 0.66}$	$\mathbf{72.28}_{\pm 0.90}$	$82.53_{\pm 0.49}$	$\mathbf{73.06}_{\pm 0.82}$	$\mathbf{90.25}_{\pm 0.54}$	$\mathbf{83.25}_{\pm 1.00}$

Table 5.4: Model comparison **without** any augmented image tags.

Dataset	**FHM**		**MAMI**		**HarM**	
Model	**AUC.**	**Acc.**	**AUC.**	**Acc.**	**AUC.**	**Acc.**
VisualBERT COCO	$72.56_{\pm 0.80}$	$64.28_{\pm 1.27}$	$80.84_{\pm 0.67}$	$72.86_{\pm 0.71}$	$82.96_{\pm 0.98}$	$78.81_{\pm 0.80}$
ViLBERT CC	$75.72_{\pm 0.91}$	$68.24_{\pm 0.44}$	$80.33_{\pm 1.01}$	$71.75_{\pm 1.14}$	$84.79_{\pm 1.23}$	$81.39_{\pm 1.62}$
MOMENTA	$69.17_{\pm 4.71}$	$61.34_{\pm 4.89}$	$81.68_{\pm 2.80}$	$72.10_{\pm 2.90}$	$86.32_{\pm 3.83}$	$80.48_{\pm 1.95}$
DisMultiHate	$79.89_{\pm 1.71}$	$71.26_{\pm 1.66}$	$80.08_{\pm 0.55}$	$71.87_{\pm 0.47}$	$86.39_{\pm 1.17}$	$81.24_{\pm 1.04}$
PromptHate	$81.45_{\pm 0.74}$	$72.98_{\pm 1.09}$	$79.95_{\pm 0.66}$	$70.31_{\pm 0.64}$	$90.96_{\pm 0.62}$	$84.47_{\pm 1.75}$
BLIP	$76.40_{\pm 1.49}$	$69.29_{\pm 1.44}$	$80.63_{\pm 1.05}$	$70.62_{\pm 1.48}$	$86.88_{\pm 1.15}$	$82.66_{\pm 1.13}$
ALBEF	$80.77_{\pm 0.81}$	$71.70_{\pm 0.98}$	$82.45_{\pm 0.85}$	$72.45_{\pm 0.96}$	$86.91_{\pm 0.72}$	$81.78_{\pm 1.20}$
Pro-Cap$_{\text{BERT}}$	$79.75_{\pm 1.15}$	$71.28_{\pm 0.91}$	$81.20_{\pm 0.69}$	$71.80_{\pm 1.42}$	$89.75_{\pm 1.49}$	$82.71_{\pm 1.60}$
Pro-Cap$_{\text{PromptHate}}$	$\mathbf{83.58}_{\pm 0.60}$	$\mathbf{75.10}_{\pm 0.97}$	$\mathbf{83.77}_{\pm 0.75}$	$\mathbf{73.63}_{\pm 0.75}$	$\mathbf{91.03}_{\pm 1.51}$	$\mathbf{85.03}_{\pm 1.51}$

Table 5.5: Model comparison **with augmenting the image entities and demographic information**.

5.2.3 Experiment Results

As discussed earlier, previous work has shown that additional image tags can enhance HMD. We therefore consider two settings for comparison: 1) without any augmented image tags; 2) with augmented image tags. We display the performance of models **without** augmented image tags in Table 5.4 and **with** augmented image tags in Table 5.5. The standard deviations ($\pm$) of ten random seed runs are also reported, and the best results are highlighted in bold.

Without augmented image tags: We first compare Pro-Cap$_{\text{BERT}}$ with unimodal and multimodal models that also utilize BERT as the text encoder (i.e., VisualBERT, ViLBERT, and MMBT-Region). Evidently, Text BERT, which utilizes only meme text, is substantially outperformed by Pro-Cap$_{\text{BERT}}$. This suggests that 1) visual signals are vital for HMD, and 2) the image captions obtained from the probing questions are informative.

Experiment results from multimodal pre-trained BERT-based models are presented in the second block of Table 5.4. Interestingly, Pro-Cap$_{\text{BERT}}$ still has better performances in all three datasets, surpassing the most powerful multimodal pre-trained BERT-base model, ViLBERT, by over 4% on FHM and surpassing MMBT-Region

Ans. Length	**FHM**	**MAMI**	**HarM**
No Centric	$70.08_{\pm 1.57}$	$72.78_{\pm 0.63}$	$80.11_{\pm 1.14}$
Penalty = 1	$71.94_{\pm 0.97}$	$73.06_{\pm 0.82}$	$82.09_{\pm 1.21}$
Penalty = 2	$72.28_{\pm 0.90}$	$72.91_{\pm 1.16}$	$82.85_{\pm 1.51}$
Penalty = 3	$71.40_{\pm 1.06}$	$72.47_{\pm 0.74}$	$83.25_{\pm 1.00}$
Pro-Cap$_{\text{PromptHate}}$	$\mathbf{72.28_{\pm 0.90}}$	$\mathbf{73.06_{\pm 0.82}}$	$\mathbf{83.25_{\pm 1.00}}$

Table 5.6: Ablation study about the impact from the length of VQA answers.

by about 3% on HarM. This is despite the fact that BERT has less model parameters compared with these multimodal models (e.g, ViLBERT has 252.1M parameters while BERT only has about 110M parameters). Pro-Cap$_{\text{BERT}}$ is still competitive against models specifically designed for HMD (i.e., models in the third block of Table 5.4). We provide experimental results of recently published multimodal pre-trained models (i.e., BLIP and ALBEF) in the fourth block. By comparing the simple Pro-Cap$_{\text{BERT}}$ with these models, we observe that Pro-Cap$_{\text{BERT}}$ gives comparable results. While Pro-Cap$_{\text{BERT}}$ does not out-perform ALBEF and BLIP all the time, performance is reasonably good given that in terms of trainable parameters, Pro-Cap$_{\text{BERT}}$ is three times smaller than these two pre-trained models. Meanwhile, Pro-Cap$_{\text{BERT}}$ shows even better results than the two models on HarM. Notably, HarM is a real-world dataset which is much noisier than FHM. HarM also focuses on a relatively new topic (COVID-19), which may not have been observed a lot by the two PTMs.

When comparing BLIP and ALBEF with PromptHate, which has a similar model size, PromptHate with Pro-Cap demonstrates significant advantages over the two models on three benchmarks, especially on the noisy HarM dataset. We conjecture that a possible reason is that multimodal PTMs leverage pre-training data that is relatively cleaner, on a smaller scale and primarily comprises of non-memes. This leads to some difficulties when confronted with noisy real-world memes. In contrast pure language models are pre-trained on larger and noisier data, which may lead to some intrinsic robustness. If visual signals are reasonably converted to text, pure textual models can be competitive for multimodal tasks such as HMD.

Reinforcing the point of proper visual signal conversion, the enhanced performance of Pro-Cap$_{\text{PromptHate}}$ over PromptHate highlights the importance of our probing-based captioning method, which provides essential cues for hateful content detection. With probe-based captioning, Pro-Cap$_{\text{PromptHate}}$ is able to conduct deep multimodal reasoning that require background knowledge (due to the good performance on FHM), is stable towards noisy real-world meme data (according to performance on HarM), and has great generalization in meme detection (according to the good performance on all three benchmarks).

With augmented image tags: For a fair comparison with recent state-of-the-art models, we consider testing our proposed probe-captioning method with the same set of augmented image tags from baselines. To utilize the augmented image tags, we simply pad these tags at the end of each textual meme representation in a similar manner to [20]. With additional image information such as entities and demographic information, most models have some improvements. An interesting thing is that neither BLIP nor ALBEF benefits much from additional image tags. This is because the additional tags are usually single words or short phrases, which may be noisy or redundant, while BLIP and ALBEF may be less capable of dealing with noisy inputs. Similar to the results in Table 5.4, when augmenting image information: 1) the simple Pro-Cap$_{\text{BERT}}$ still obviously surpasses multimodal pre-trained BERT-base models such as VisualBERT or ViLBERT; 2) the Pro-Cap$_{\text{BERT}}$ performs better than models with similar sizes but specifically designed for HMD (i.e., MOMENTA or DisMultiHate) in most cases; 3) the Pro-Cap$_{\text{BERT}}$ achieves comparable results compared with more powerful multimodal PTMs, which is about three times larger and surpasses them on the HarM dataset, which is real-world and noisy; 4) Pro-Cap$_{\text{PromptHate}}$ surpasses the original PromptHate and achieves the best performance on three benchmarks as well.

An interesting point is that comparing Pro-Cap$_{\text{PromptHate}}$ without any augmented tags and original PromptHate with augmented additional image information, they achieve comparable performance on FHM and HarM and the former even surpasses the latter on MAMI. However, extracting the additional image information is expensive and laborious, which can be replaced by probing-based captioning according to the experimental results. The equally good performance on three benchmarks highlights the stability and generalization of our proposed approach.

5.2.4 Ablation Study

In this section, we conduct ablation studies to better understand our Pro-Cap method. Specifically, we consider the impact of asking different questions and the impact of the length of answers to the probing questions. To eliminate other factors, we consider Pro-Cap$_{\text{PromptHate}}$ without any augmented image tags. For brevity, we only show accuracy in this section. We present the full results in Appendix B.2.

The impact of asking hateful-content centric questions: We first conduct an ablation study on the effect of prompting PT-VLMs with questions facilitating HMD. According to Table 5.2, the first question asks about the image content while all questions in the second block are for common vulnerable targets of hateful contents. To better understand the impact of including image captions generated by these target-specific questions, we experiment with a setting where captions from the target-specific questions are removed and only the generic caption about image content is used. The results are shown in the first block of Table 5.6. Compared with the last block of the table, we observe that with captions generated by target-specific probing questions, the model's performance improved on all three datasets, specifically with over 2% on FHM and over 3% on HarM. However, we notice minor improvement on MAMI. We believe that this is because MAMI memes are all related to woman and generic captions about meme images may already cover the gender of persons in the image. However, the other two datasets involve memes with more complexities and therefore asking a wide ragen of target-specific probing questions is more helpful. It also implies that in real-world HMD, probing-based captioning would be helpful.

The length of answers to probing questions: We apply BLIP-2 as a zero-shot VQA model. Different from existing VQA benchmarks [73, 54], where answers are often single words or short phrases, we may want the answers used as image captions to be longer and thus more informative. In this cases, we experiment with answers of different length. To conduct the analysis, we set the length penalty in BLIP-2's text decoder for answer generation with different values (i.e., 1, 2 and 3). With increased length penalty, longer answers are encouraged. We show results of model performance with different answer length in Table 5.6. The results show that detection performance is robust and does not vary much with different answer lengths. This indicates the stability of the Pro-Cap method. On the other hand, to a very small extent, different datasets do favor answers of different lengths. For instance, the HarM dataset prefers longer answers while the MAMI dataset prefers shorter answers.

5.3 Qualitative Analysis

In this section, we conduct case studies to better understand the strengths and limitations of our proposed method. We first compare Pro-Cap$_{\text{PromptHate}}$ against PromptHate with image captions and show examples in Table 5.7. From the three examples, we observe that in most cases, generic captions about the image content do not provide the key information for HMD, while asking questions about common vulnerable targets helps. For instance, in the first example, the answer from asking questions about race, country and religion all provide some key words such as *islamic* or *muslim*; in the second example, answers to questions about country and religion are important image captions and the answer to the race-related question is the most important for HMD. In contrast, we observe that the basic captions in the original PromptHate miss these crucial facts about the meme images.

Next, we conduct error analysis about our proposed probe-captioning in Table 5.8. In the first example, all probe-captions generate sufficient image captions for HMD, while the model still fails at prediction. This may be due to the current language models performing poorly in further complex reasoning. We also note that the

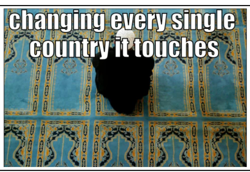
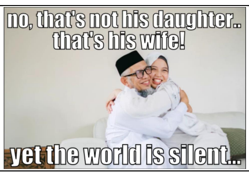
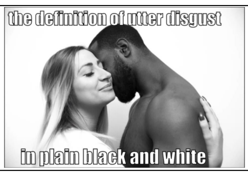
Meme		 no, that's not his daughter.. that's his wife! yet the world is silent...	
Ground Truth	Hateful (religion)	Hateful (religion)	Hateful (race)
Basic PromptHate	Non-hateful	Non-hateful	Non-hateful
Pro-Cap$_{\text{PromptHate}}$	Hateful	Hateful	Hateful
Meme text	changing every single country it touches	no that is not his daughter that is his wife yet the world is silent	the definition of utter disgust in plain black and white
Basic caption	mughal structure is one of the largest mosques in the world.	portrait of a father hugging his daughter while smiling at camera in the living room at home.	love is in the air!.
Pro-Cap	(Content:) a black cat sitting on a blue and white tiled floor. (Race:) a black person is standing on a blue and white tiled floor in islamic. (Gender:) a man in a black shirt is standing on a blue and white tiled floor with a clock on top of his head. (Country:) islamic. (Religion:) the person is a muslim and he is wearing a black t-shirt and a black sleeveless.	(Content:) a man and a woman hugging on a couch. (Race:) a white man and a white woman hugging on a white couch. (Gender:) a man and a woman hugging on a white couch. (Country:) islamic. (Religion:) an muslim man and woman hugging on a white couch.	(Content:) a black and white photo of a man and a woman. (Race:) a black man and a white woman in a black and white photo. (Gender:) a man and a woman in a black and white photo. (Country:) afghanistan. (Religion:) he is a christian.

Table 5.7: Comparison between Pro-Cap$_{\text{PromptHate}}$ with basic PromptHate. The image caption used by basic PromptHate is denoted as basic caption. Incorrect prediction in red. The content in $(\cdot)$ of the ground-truth is the target of the hateful meme.

small scale of hateful meme datasets may be inadequate for training a model to perform complex reasoning. Recent studies about large language models pre-trained with trillions of words [172] may facilitate hateful meme detection to some extent. Besides, we observe minor errors in predicted answers from the zero-shot VQA

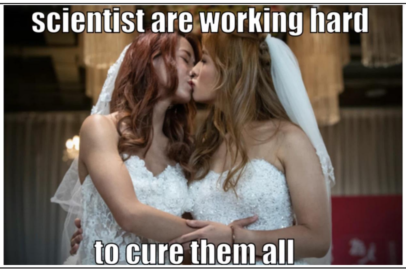
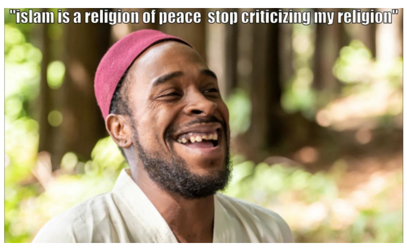
Meme		
GT	Hateful (gender)	Non-hateful
Pred	Non-hateful	Hateful
Meme text	scientist are working hard to cure them all	islam is a religion of peace stop criticizing my religion
Pro-Cap	(Content:) two women in wedding dresses kissing each other. (Race:) a white woman kissing a brunette woman in a wedding dress. (Gender:) a woman is kissing a man in a wedding dress. (Country:) the person in the image comes from a country in the philippines. (Religion:) the person in the image is a christian.	(Content:) a man with a beard laughing in the woods. (Race:) a african man with a beard and a red hat is smiling in the woods. (Gender:) a man with a beard and a red hat in front of a wooded area. (Country:) egypt is the country that the person in the image comes from. (Religion:) he is a muslim man with a beard and a red tiara on his head.

Table 5.8: Error cases of Pro-Cap$_{\text{PromptHate}}$.

model (e.g., the wrong prediction of "a woman kissing a man" when asking about gender). It highlights that with the development of better zero-shot VQA models, the our strategy could potentially facilitate more for the two text-based HMD models. The second example highlights a limitation of most hateful content detection models in that they may be biased. During the training stage, there may be hateful contents towards Muslims so that once models seen Muslims, they tend to predict the meme as hateful. To alleviate the issue, debiasing techniques may be needed. Due to space limitation, we omit visualization examples in the main pages and refer the reader to examples in Appendix B.3.

5.4 Conclusion

In this chapter, we attempt to leverage PT-VLMs in a low-computation-cost manner to aid the task of HMD. Specifically, without any fine-tuning of PT-VLMs, we probe them in a zero-shot VQA manner to generate hateful content-related image captions. With the distilled knowledge from large PT-VLMs, we observe that a simple language model, BERT, can surpass all multimodal pre-trained BERT models of a similar scale. PromptHate with probe-captioning outperforms previous results significantly and achieves the new state-of-the-art on three benchmarks.

Chapter 6

Harnessing a Composition of Tuned PTMs with Essential Reasoning Skills for Few-shot HMD

In response to the proliferation of hateful memes, as we introduced in Section 2.2.3, researchers have developed various detection methods. One strategy regards HMD as a general multimodal classification task. It directly fine-tunes PT-VLMs to bridge the multimodal gap [213, 176, 128, 108]. Alternatively, another approach integrates these PTMs within specialized architectures specifically designed for detecting hateful content [92, 89, 140, 20]. Nonetheless, both strategies predominantly rely on extensive supervised learning, necessitating large volumes of annotated data — a process that is both costly and time-consuming. Furthermore, the emergence of hateful memes tied to evolving events poses a significant challenge: acquiring and annotating sufficient training examples for each novel occurrence is often impractical. We posit that existing studies lack adequate exploration in the low-resource setting, where the detection systems must operate effectively with minimal labeled data.

To our knowledge, the niche of few-shot detection of hateful memes has yet to be thoroughly explored in existing literature. One intuitive approach to addressing this challenge is to harness the in-context learning potential of PT-VLMs [96, 9, 2]. This

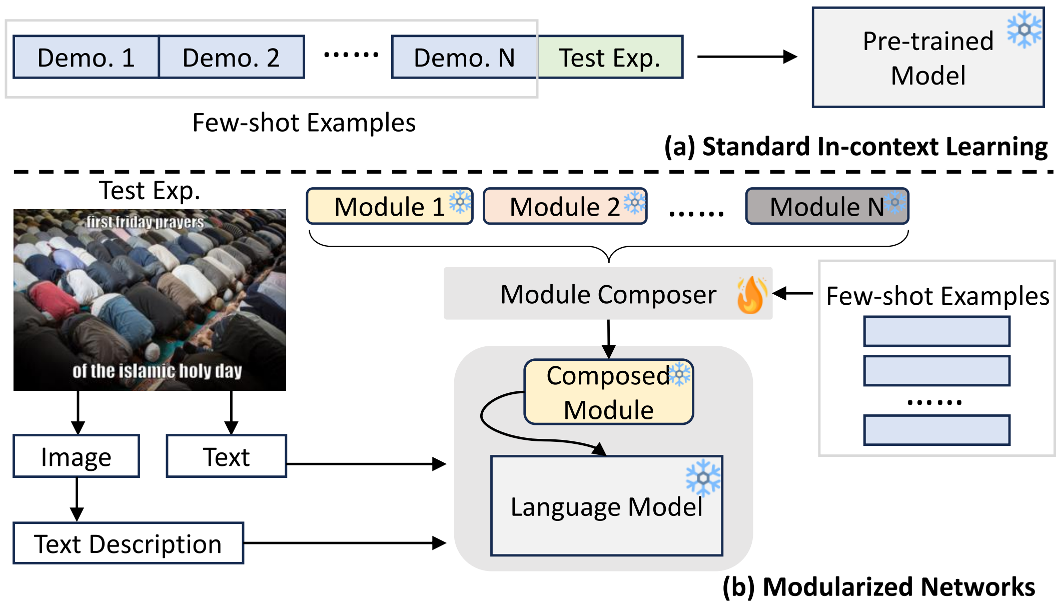


Figure 6.1: Comparison of standard in-context learning and our proposed modularized networks for few-shot HMD.

method would entail using a limited number of annotated examples as a guide for the model. For instance, when evaluating an unknown sample, a set of these annotated "demonstration" instances would be bundled with the test instance. This combined input is then processed by the PT-VLM to predict whether the content is hateful (as shown in Figure 6.1(a)). While this method shows promise in low-resource scenarios for various multimodal tasks, it underperforms specifically in the domain of few-shot HMD [9, 2]. Additionally, the process of repeatedly combining few-shot examples with test instances incurs significant computational overhead during each inference step, which may be prohibitive in practice.

PT-LMs, especially Large language models (LLMs) have recently achieved impressive results in a range of NLP tasks. To leverage the power of LLM in multimodal settings, a straightforward method is to convert images to captions and then feed the captions together with other textual inputs into an LLM. Previous work has adopted this method for VQA [171] and robot navigation [201]. In this paper, we also leverage LLMs for multimodal HMD following this strategy. To tailor LLMs to new tasks, researchers have introduced various parameter-efficient tuning methods [67, 68, 57, 103] as alternatives to adjusting entire models, which often consist of billions of parameters. One such method, known as Low-rank adaptation (LoRA) [68], strategically updates weights by decomposing them into low-rank matrices, thus reducing the parameter space. However, LoRA is not directly applicable in our few-shot setting, due to insufficient training data. Drawing inspiration from LoraHub's analysis of LoRA's compositional abilities [71], our approach involves composing tuned PTMs by composing their LoRA modules, each of which has essential reasoning skills for HMD. The LoRA modules of PTMs are built upon LLMs, such as the LLaMA model [172], and are supervised with related tasks to HMD for the acquisition of essential skills. Then, they will be combined in a modular fashion for the task of HMD. Instead of acquiring the core competencies with a few limited examples, we transfer the learned skills from relevant tasks and only need to learn to compose these skills for HMD.

For effective detection, we have pinpointed three core competencies required: (i) Grasping the concept of hateful content; (ii) Decoding the message behind multimodal memes; (iii) Elucidating the rationale behind the hateful classification of a meme. We tailor LoRA modules to these competencies by training them on three specific tasks: (a) *hate speech detection*, (b) *meme comprehension*, and (c) *hateful meme explanation*. Since meme comprehension and HMD require processing multimodal data, we transform images into textual descriptions to accommodate the text-centric nature of LLMs. Subsequently, we employ supervised data from the three tasks, other than HMD, to train the corresponding LoRA modules using the LLaMA model. With the few available examples for HMD, we train a module composer assigning importance scores over the three modules. This composer has a minimal number of parameters, equivalent to the number of modules, making it suitable for a few-shot learning scenario. With the module composer, we transfer the capabilities learned from related tasks to the task of HMD through composition of LoRA modules. In other words, we learn to compose tuned PTMs to adapt to HMD.

By integrating the composed LoRA modules into LLaMA, we create a modular network that embodies the essential detection skills. This network contrasts with traditional in-context learning methods, notably during inference: it is more efficient as it bypasses the need to process few-shot examples with each inference step, using them solely in the training phase of the module composer.

Our validation of the proposed HMD method involved comprehensive testing across three established benchmarks. Our approach not only demonstrates greater efficiency at the inference stage but also consistently outperforms established in-context learning baselines across all test datasets. Notably, even with a limited dataset of four examples (4-shot learning), our method outperforms the 32-shot implementation of the Flamingo model [2], which requires the entire set of examples to be processed for each evaluation instance.

Our approach uses the strategy of **Tuning-Composition** when applying PTMs, as introduced in Section 1.3. We identify three essential skills for HMD and tuned PTMs to acquire the three capabilities respectively. We then obtain a set of tuned models $\{\mathcal{P}_k^{\text{tune}}\}_{k=1}^{3}$. Next, based on the few-shot available annotated examples, we train a composer, learning to assign importance scores over each tuned model regarding the HMD task. We then construct modularized networks for HMD by weighted averaging tuned models. The contributions are identifying essential reasoning skills for HMD and composing a set of tuned models, each of which capable of an essential skill, for HMD.

In the following paragraphs, we will first discuss about the few-shot setting for evaluation and provide a brief introduction for LoRA in Section 6.1. Details of the proposed model are provided in Section 6.2. In Section 6.3 and Section 6.4, we present experimental results and qualitative analysis. Finally, we provide a conclusion of the chapter.

6.1 Preliminary

6.1.1 Few-shot Hateful Meme Detection

Given a multimodal meme (i.e., the meme image $\mathcal{I}$ and superimposed text $\mathcal{C}$), a HMD model is required to decide whether the meme is *hateful* or *non-hateful*. Models are first trained with labeled training data $\mathcal{D}_{\text{train}} = (\mathcal{I}^n, \mathcal{C}^n, \hat{a}^n)_{n=1}^{N}$, where $\hat{a}^n$ is the ground-truth label. Trained models are evaluated on the testing split $\mathcal{D}_{\text{test}}$. In this work, we assume a low-resource setting where only a few labeled examples are available. We follow the definition in [50, 208, 209] and assume K training examples per class are available in the K-shot setting (i.e., $N = 2 * K$ for our case). Our goal is to optimize the models based on $\mathcal{D}_{\text{train}}$ so that they can generalize well on the testing data $\mathcal{D}_{\text{test}}$.

6.1.2 Low-Rank Adaptation

LoRA [68] is a parameter-efficient tuning method, which decomposes the updates of attention weights into combination of low-rank matrices, while keeping the pre-trained weights (model parameters of LLMs) frozen. Formally, given the i-th attention weight matrix $\mathbf{W}_i \in \mathbb{R}^{p \times q}$ and its accumulated gradient update $\Delta\mathbf{W}_i$, LoRA approximates the update as follows:

$$\mathbf{W}_i := \mathbf{W}_i + \Delta\mathbf{W}_i, \tag{6.1}$$

$$:\approx \mathbf{W}_i + \mathbf{A}_i\mathbf{B}_i, \tag{6.2}$$

where $\mathbf{A}_i \in \mathbb{R}^{p \times r}$ and $\mathbf{B}_i \in \mathbb{R}^{r \times q}$ are the decomposed matrices with a low-rank r ($r \ll p, r \ll q$).

LoRA largely reduces the number of trainable parameters compared with direct fine-tuning. Additionally, it allows for the composition of an LLM with various LoRA modules, each tailored to specific tasks. In this context, a LoRA module learned from one task can be considered a module with specific capabilities.

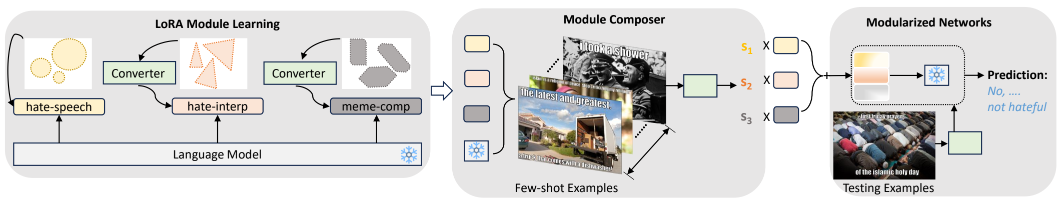


Figure 6.2: Overview of the proposed Mod-HATE model. It consists of three steps: 1) LoRA module learning from relevant tasks to obtain essential skills for hateful meme detection; 2) training a module composer with few-shot training data to learn importance scores assigned over each LoRA module; 3) the construction of modularized networks by integrating the composition of learned LoRA modules with frozen LLMs.

6.2 Mod-HATE Model

6.2.1 Overview

In this section, we introduce our innovative approach to few-shot HMD, known as Modularized Networks for Hateful Meme Detection (**Mod-HATE**). The core concept behind Mod-HATE is the acquisition of essential reasoning skills for detecting hateful memes through the learning of specialized modules. These modules are acquired from tasks closely aligned with HMD. Based on the few-shot training examples, we train a module composer to assign importance scores to these modules. Subsequently, we create a composed module by weighted averaging the learned modules. We then construct modularized networks by integrating this composed module with LLMs. The modularized networks are designed for the specific purpose of HMD. The overview of Mod-HATE is illustrated in Figure 6.2.

Specifically, we employ LoRA based on LLMs to acquire LoRA modules with reasoning capabilities from related tasks. Since some of these tasks, including HMD, involve image information, and LLMs are inherently textual, we incorporate a converter that transforms images into textual descriptions. The details of the converter, the introduction of relevant tasks, and the training of LoRA modules are elaborated in Section 6.2.2. In Section 6.2.3, we delve into the training of a module composer, responsible for generating importance scores for individual modules. Section 6.2.4 is dedicated to the construction of modularized networks

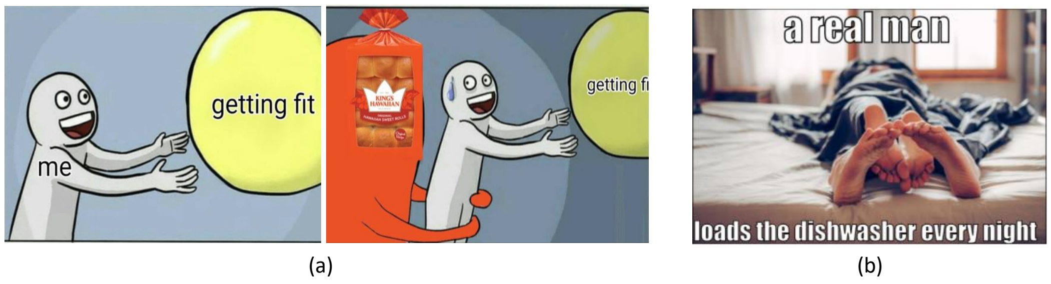


Figure 6.3: Examples of (a) meme comprehension and (b) hateful meme interpretation tasks.

by the integration of composed modules with LLMs. Lastly, we demonstrate the application of modularized networks for HMD.

6.2.2 LoRA Module Learning

In this section, we discuss the learning of LoRA modules from closely related tasks that cover essential reasoning skills for HMD.

Converter

As mentioned in Section 6.2.1, certain tasks may involve multimodal information that is beyond the comprehension of LLMs. To address this, we employ a converter (denoted as `Converter`) to translate images into textual descriptions. It's important to note that, in line with the approach taken by authors in [21], generic image captions generated by image caption generators for meme images might overlook vital cues necessary for the detection of hateful memes. Since our primary focus is not on better utilization of PT-VLMs, we have opted for an image captioning model as our converter for simplicity. We use the PT-VLM, BLIP-2, FlanT5$_{\text{XL}}$ [100] version as the captioning model. Given an image $\mathcal{I}$, `Converter` will transform it into textual descriptions, $\mathcal{T}$, of the image:

$$\mathcal{T} = \texttt{Converter}(\mathcal{I}).$$

Relevant Tasks and Supervised Data

We identify three essential skills for HMD: 1) the understanding of what constitutes *hateful* content; 2) the ability to decipher the concealed meaning in multimodal memes and 3) the capability to interpret why a meme is hateful. To acquire LoRA modules proficient in these reasoning skills, we leverage three distinct tasks, each requiring one of these skills, as follows:

Hate Speech Detection: Given a piece of text, models are required to predict whether the text is *hateful* or *non-hateful*. We transform the task into a text generation task by asking the question: *Please decide whether the sentence below is a hate speech. Text:* `[TEXT]`, where `[TEXT]` is a placeholder for the input text. If the text is annotated as *hateful*, the expected output is *Yes*, otherwise, *No*,. We aggregate three hate speech datasets: DT [34], WZ [184] and Gab [141] as the training data. To unify the datasets, we combine *non-hateful* and *offensive* tweets in DT dataset into the class of *non-hateful*.

Meme Comprehension: Given a meme image $\mathcal{I}$ and the meme texts $\mathcal{C}$, meme comprehension requires to decode the meaning of multimodal memes. For instance, the meme in Figure 6.3(a) is trying to express the temptation of sweet food to the poster. To make the image comprehensible to LLMs, we generate its textual description $\mathcal{T}$ with the `Converter`. Based on the image description and the meme text, we train a module for generating the meaning of the meme with the prompt: *Please interpret the meme according to its image caption and meme text. Image Caption:* `[CAP]`*; Meme text:* `[MEME_TEXT]`. The `[CAP]` and `[MEME_TEXT]` are placeholders and will be replaced by $\mathcal{T}$ and $\mathcal{C}$ respectively. We leverage the MEMECAP dataset [74], which consists of multimodal memes with their corresponding meanings, as the training dataset.

Hateful Meme Interpretation: The hateful meme interpretation task requires models to give explanations as to why a meme is hateful. For instance, given the meme (i.e., the meme image $\mathcal{I}$ and the meme text $\mathcal{C}$) in Figure 6.3(b), the expected output from models should be the reasoning that the meme is annotated as hateful:

the meme dehumanizes the females as sexual objects as well as less capable beings only good for dishwashing. Similarly, as the task contains multimodal information, we leverage the `Converter` for the generation of textual image descriptions. Then we prompt LLMs with the instruction and inputs: *Please explain the reason that the meme is hateful given the image caption and meme text. Image Caption:* `[CAP]`*; Meme text:* `[MEME_TEXT]`. The `[CAP]` and `[MEME_TEXT]` will be replaced by $\mathcal{T}$ and $\mathcal{C}$ respectively. To supervise the learning of modules, we use the annotated interpretation of hateful memes in [64] as training data.

Training of LoRA Modules

As introduced in Section 6.2.2, we unify all relevant tasks as text generation. We adopt the widely used cross entropy loss to optimize LLMs for text generation. We use the open-source powerful language model, LLaMA (7B) [172] as our language model. LLaMA is a decoder-only model but we only optimize for the expected output tokens rather than both the inputs and outputs. The input tokens are masked for loss computation. Instead of tuning all parameters of LLMs, we use the LoRA parameter-efficient tuning method, introduced in Section 6.1.2 to tune the LLM. After training, we regard the set of combinations of low-rank matrices $\mathbf{A}_i\mathbf{B}_i$ as the LoRA module. Therefore, we obtain a set of LoRA modules, $\{\mathcal{L}_m\}_{m=1}^M$ regarding the considered relevant tasks, where $\mathcal{L}_m$ is the m-th LoRA module. In our case, $M = 3$.

6.2.3 Module Composer

Once we obtain the list of LoRA modules, the module composer will learn how to assign importance scores to these modules based on the few-shot training examples for HMD. The optimization objective is to generate the expected outputs for few-shot examples by composing the previously acquired LoRA modules. Specifically, given the learned LoRA module set $\{\mathcal{L}_m\}_{m=1}^M$, our goal is to train the module composer to generate a corresponding set of importance scores $\{s_m\}_{m=1}^M$. The composed module,

$\mathcal{L}_{\text{comp}}$ is computed as a weighted average over the LoRA modules:

$$\mathcal{L}_{\text{comp}} = \sum_{m=1}^{M} s_m \mathcal{L}_m.$$

Indeed the weighted-average of LoRA modules will be converted into the weighted-average of their low-rank matrices:

$$\mathbf{A}_{\text{comp},i} = \sum_{m=1}^{M} s_m \mathbf{A}_i, \tag{6.3}$$

$$\mathbf{B}_{\text{comp},i} = \sum_{m=1}^{M} s_m \mathbf{B}_i, \tag{6.4}$$

where $\mathbf{A}_{\text{comp},i}$ and $\mathbf{B}_{\text{comp},i}$ are decomposed low-rank matrices for the i-th attention weight matrix's update in $\mathcal{L}_{\text{comp}}$. Next, we adapt the LLM with the composed module $\mathcal{L}_{\text{comp}}$ to construct the modularized networks. Based on the modularized networks, we optimize the module composer with both the LLM and learned LoRA modules frozen and update only the module composer with the few-shot hateful meme examples.

Similar to those multimodal tasks mentioned in Section 6.2.2 (i.e., meme comprehension and hateful meme interpretation), we use `Converter` to transform the meme image into its textual description $\mathcal{T}$. Given the meme text $\mathcal{C}$ and the meme image description $\mathcal{T}$, we ask the modularized networks the following question: *Please decide whether the meme is hateful given the image caption and meme text. Image Caption:* `[CAP]`*; Meme text:* `[MEME_TEXT]`. The `[CAP]` and `[MEME_TEXT]` will be replaced by $\mathcal{T}$ and $\mathcal{C}$ respectively. If the few-shot example is *non-hateful*, the expected output will be *No*; otherwise, *Yes*. The modularized networks are going to optimize the importance scores (i.e., the module composer) to maximize the likelihood of generating expected outputs. We denote the loss from language modeling as L_{lm}. Besides, to regularize generated importance scores, L1 normalization is added to penalize extreme values. The final loss is:

$$L = L_{\text{lm}} + \lambda \sum_{m=1}^{M} |s_m|,$$

where λ is a hyper-parameter adjusting the importance between the task loss and the normalization of the scores. We adopt the same optimization strategy as introduced in [71] and use a gradient-free optimization method, *Covariance Matrix Adaptive Evolution Strategies* [60], to minimize the loss.

6.2.4 Modularized Networks

In this section, we describe the construction of modularized networks, which will be used as the architecture for training the module composer as mentioned in Section 6.2.3. The modularized networks consist of an LLM and the LoRA adapter, which is the composition of LoRA modules for relevant tasks (i.e., $\mathcal{L}_{\text{comp}}$). The attention weight matrices in the frozen LLM will be updated as:

$$\begin{aligned} \mathbf{W}_i &:\approx \mathbf{W}_i + \mathbf{A}_{\text{comp},i}\mathbf{B}_{\text{comp},i}, && (6.5)\\ &:= \mathbf{W}_i + (\sum_{m=1}^{M} s_m \mathbf{A}_i)(\sum_{m=1}^{M} s_m \mathbf{B}_i). && (6.6) \end{aligned}$$

The networks is modularized by changing the importance scores over LoRA modules so that the importance of modules will be adjusted, to adapt to new tasks or datasets. Based on the modularized networks, we optimize the module composer and determine the final importance scores assigned to the LoRA modules. Subsequently, we apply these modularized networks to our primary task, which is the detection of hateful memes.

6.2.5 Model Prediction

HMD can be conceptualized as a binary classification task. This task requires the prediction of a probability for each potential class, which is crucial for certain evaluation metrics, such as the Receiver Operating Characteristics (ROC) curve [85]. From our modular networks, we extract the output probabilities corresponding to the first predicted token, denoted as $\mathbf{o}^{|\mathcal{V}|}$, wherein $\mathcal{V}$ represents the vocabulary set utilized by the LLM. For the binary outcomes of '*non-hateful*' and '*hateful*' memes, '*No*'

Datasets	**Test**	
	#Hate.	#Non-hate.
FHM	247	253
MAMI	500	495
HarM	124	230

Table 6.1: Statistical distributions of test sets.

Dataset	**# shots**	**FHM**		**MAMI**		**HarM**	
Model		**AUC.**	**Acc.**	**AUC.**	**Acc.**	**AUC.**	**Acc.**
OPT-13B	4	$49.8_{\pm 3.71}$	$50.2_{\pm 1.07}$	$54.1_{\pm 3.31}$	$50.0_{\pm 0.35}$	$54.9_{\pm 7.85}$	$59.6_{\pm 3.11}$
OPT-30B	4	$50.9_{\pm 3.00}$	$50.0_{\pm 1.68}$	$54.2_{\pm 4.39}$	$50.5_{\pm 1.05}$	$59.3_{\pm 9.19}$	$62.3_{\pm 5.13}$
OpenFlamingo-3B	4	$51.3_{\pm 1.63}$	$49.2_{\pm 0.00}$	$43.7_{\pm 0.51}$	$50.3_{\pm 0.00}$	$57.2_{\pm 1.66}$	$35.0_{\pm 0.00}$
OpenFlamingo-9B	4	$59.4_{\pm 0.33}$	$52.1_{\pm 0.72}$	$59.8_{\pm 2.11}$	$50.4_{\pm 0.90}$	$63.6_{\pm 3.15}$	$65.2_{\pm 0.22}$
Flamingo-3B	4	53.6	-	-	-	-	-
Flamingo-9B	4	62.7	-	-	-	-	-
OPT-13B	8	$50.7_{\pm 3.82}$	$50.3_{\pm 1.78}$	$56.2_{\pm 4.14}$	$52.9_{\pm 3.10}$	$61.6_{\pm 5.59}$	$48.4_{\pm 8.72}$
OPT-30B	8	$53.5_{\pm 2.61}$	$51.5_{\pm 1.90}$	$54.0_{\pm 5.56}$	$50.7_{\pm 1.84}$	$64.2_{\pm 4.88}$	$61.9_{\pm 6.40}$
OpenFlamingo-3B	8	$49.1_{\pm 0.44}$	$49.2_{\pm 0.00}$	$42.1_{\pm 1.85}$	$50.3_{\pm 0.00}$	$59.1_{\pm 2.21}$	$35.0_{\pm 0.00}$
OpenFlamingo-9B	8	$58.7_{\pm 0.94}$	$51.6_{\pm 0.52}$	$59.1_{\pm 2.86}$	$50.0_{\pm 0.15}$	$62.9_{\pm 2.69}$	$65.1_{\pm 0.23}$
Flamingo-3B	8	54.7	-	-	-	-	-
Flamingo-9B	8	63.9	-	-	-	-	-
Flamingo-3B	32	55.3	-	-	-	-	-
Flamingo-9B	32	63.5	-	-	-	-	-
Our Proposed Method							
Mod-HATE	4	$\mathbf{64.5}_{\pm 0.19}$	$\mathbf{58.0}_{\pm 1.07}$	$\mathbf{67.4}_{\pm 0.46}$	$61.0_{\pm 2.22}$	$\mathbf{73.4}_{\pm 0.27}$	$69.4_{\pm 0.42}$
Mod-HATE	8	$64.0_{\pm 0.19}$	$57.4_{\pm 0.82}$	$67.2_{\pm 0.15}$	$\mathbf{61.1}_{\pm 0.44}$	$73.1_{\pm 0.16}$	$\mathbf{69.5}_{\pm 0.35}$

Table 6.2: Comparison with existing methods for few-shot HMD.

and '*Yes*' are used as the expected outputs from the LLM, respectively. Therefore, the probability of '*No*' is used to gauge the probability of the meme being classified as non-hateful. Conversely, the probability of 'Yes' is indicative of the meme being flagged as hateful. Finally, we obtain the probability of classification, $\mathbf{a} \in \mathbb{R}^2$, where $a_0 = \mathbf{o}_i, (\mathcal{V}_i = \text{No})$ and $a_1 = \mathbf{o}_j, (\mathcal{V}_j = \text{Yes})$.

6.3 Experiment Results

In this section, we first detail the evaluation framework, encompassing the datasets used, the metrics applied for assessment, and the specifics of our implementation. Subsequently, we introduce the baseline models for comparison and present our experimental results, delineating the performance contrasts between these baselines and our proposed model. Following this, we perform ablation studies to ascertain the contribution of individual components within our method. Finally, we offer case studies to elucidate the strengths of our approach, providing deeper insights into its practical application.

6.3.1 Evaluation Setting

Datasets: To assess the effectiveness of our proposed method, we conducted evaluations using three benchmark datasets, which are commonly used in HMD studies. The *Facebook Hateful Meme* dataset (**FHM**)[85] encompasses a diverse range of synthetic memes, which are designed to include confounders that necessitate genuine multimodal reasoning for accurate classification, and these memes often target various vulnerable groups. The *Multimedia Automatic Misogyny Identification* dataset (**MAMI**)[42], contains memes specifically derogatory towards women, reflecting the common targets of online vitriol. These memes are sourced from actual content on social platforms like Twitter and Reddit. Furthermore, considering hateful memes are also harmful we also utilize the *Harmful Meme* dataset (**HarM**)[139] to test the generalizability of our method. This dataset concentrates on COVID-19 related memes and categorizes them into three levels of harm: *harmless*, *partially harmful*, and *very harmful*. For our purposes, we have combined the latter two categories under a single label: *harmful*. The statistical distributions for the original test splits of these datasets are detailed in Table 6.1.

Evaluation Protocols: For evaluation metrics, we employ standard accuracy (**Acc.**) and the Area Under the Receiver Operating Characteristics curve (**AUCROC**), consistent with benchmarks used in existing studies [85, 20, 92, 108, 213]. In the context of few-shot learning, evaluations can exhibit high variability due to the selection of sample examples. To mitigate this, we align with the approach proposed by [50], which suggests that generating multiple few-shot training sets using different random seeds can lead to a more reliable performance evaluation. We generate five sets of few-shot examples with five random seeds for each K-shot setting. Consequently, we present the average accuracy and AUCROC scores computed over the test set, following training on these various few-shot samples.

Implementation Details: To extract the meme texts on the image, we use the open-source package EasyOCR [1] for meme text detection. Before captioning meme

[1]https://github.com/JaidedAI/EasyOCR

images, in order to avoid the noise from the meme texts on the image, we follow [213] to remove the meme texts on the image. For the optimization of the module composer, we use Covariance Matrix Adaptive Evolution Strategies [60] provided by *Nevergrad* [2], the gradient-free optimization platform. More details about implementations (e.g., number of model parameters, package versions and computation costs) are provided in Appendix C.1.

6.3.2 Baselines

In this section, we introduce baselines for few-shot HMD using in-context learning. These baselines utilize a few training examples as demonstrations and prompt pre-trained models with the concatenation of demonstrations and the testing example for prediction. We examine baselines built on PT-VLMs, adept at processing combined image and text sequences for multimodal in-context learning. For instance, Flamingo, derived from the Chinchilla LLM [65], integrates additional parameters for multimodal pre-training, showing proficiency in various few-shot multimodal tasks. However, Flamingo is proprietary, limiting our performance evaluation to the FHM dataset using reported outcomes for its models with 3 billion (3B) and 9 billion (9B) parameters. Alternatively, OpenFlamingo, an open-source version modeled after Flamingo, uses the MPT LLM [3] as its foundation. We assessed OpenFlamingo with both 3B and 9B configurations. For the latter, due to the absence of multi-GPU support in OpenFlamingo, we employed Otter-9B [96], which is based on OpenFlamingo but further optimized for instructional tasks.

Further, large LLMs such as GPT-3 [15] are also recognized for in-context learning efficacy. As our approach translates visual content into textual form before leveraging LLMs, for a fair comparison, we also consider in-context learning with LLMs after the image conversion. Specifically, we employ the freely available OPT model [204] for this purpose, a widely recognized stand-in for GPT-3. For comprehensive details on the templates used to facilitate in-context learning with

[2]https://github.com/facebookresearch/nevergrad
[3]www.mosaicml.com/blog/mpt-7b

Dataset	**FHM**		**MAMI**		**HarM**	
Model	**AUC.**	**Acc.**	**AUC.**	**Acc.**	**AUC.**	**Acc.**
			Proposed Models with Individual Modules, Zero-shot			
hate-speech	64.3	56.0	**72.7**	53.6	**74.3**	65.5
hate-interp	56.8	49.4	56.9	50.5	60.9	35.0
meme-captions	47.5	51.4	46.1	48.9	40.3	64.4
			Proposed Models with Composition of Two Modules			
meme-comp, hate-speech	$63.3_{\pm 0.17}$	$54.2_{\pm 0.20}$	$69.4_{\pm 0.36}$	$52.0_{\pm 0.50}$	$70.9_{\pm 0.49}$	$65.9_{\pm 0.27}$
meme-comp, hate-interp	$59.5_{\pm 0.07}$	$49.4_{\pm 0.00}$	$55.3_{\pm 0.05}$	$50.3_{\pm 0.00}$	$56.9_{\pm 0.05}$	$35.3_{\pm 0.00}$
hate-speech, hate-interp	$64.1_{\pm 0.40}$	$56.3_{\pm 1.44}$	$67.7_{\pm 0.38}$	$58.4_{\pm 1.94}$	$72.9_{\pm 0.31}$	$69.2_{\pm 0.55}$
			Proposed Models with all Modules			
Mod-HATE	$\mathbf{64.5}_{\pm 0.19}$	$\mathbf{58.0}_{\pm 1.07}$	$67.4_{\pm 0.46}$	$\mathbf{61.0}_{\pm 2.22}$	$73.4_{\pm 0.27}$	$\mathbf{69.4}_{\pm 0.42}$

Table 6.3: Ablation studies of different modules in the 4-shot setting. *hate-speech* refers to the LoRA module for hate speech detection; *hate-interp* is the LoRA module for hateful meme interpretation; *meme-comp* is the LoRA module for meme comprehension.

# shots	**Dataset**	**H-S**	**H-I**	**M-C**
4-shots	FHM	0.4865	0.4561	-0.0013
	MAMI	0.4210	0.4707	0.0024
	HarM	0.4564	0.4532	0.0025
8-shots	FHM	0.4713	0.3921	0.0013
	MAMI	0.4139	0.4453	0.0014
	HarM	0.4127	0.4512	0.0010

Table 6.4: Weights of LoRA modules of our Mod-HATE model. **H-S** for the hate-speech LoRA, **H-I** for the hate-interp LoRA module and **M-C** for the meme-comp module.

these models, we direct the reader to Appendix C.3.

6.3.3 Experiment Results

Our experimental analysis was conducted under two few-shot learning scenarios: with 4-shot and 8-shot examples. The findings, as summarized in Table 6.2, reveal that our proposed model outperform all in-context learning baselines across all three benchmarks. These results hold true in both few-shot configurations. Furthermore, our model also demonstrates superior performance compared to the Flamingo model's 32-shot results reported in a prior study [2].

Scaling up of models. Our findings suggest that increasing the size of model parameters tends to enhance the performance of in-context learning methods. This observation is in line with recent research findings [30], indicating a positive correlation between model size and improved detection metrics. Consequently, substituting our current 7B-parameter language model, LLaMA-7B, with its larger counterparts, such as the 13B or 65B versions, could potentially lead to further advancements in performance. Such scalability suggests that our approach may be effectively adapted to larger language models, providing robust HMD capabilities with few training examples.

In-context learning with LLMs and PT-VLMs. Our study's findings suggest a marked preference for in-context learning using PT-VLMs, exemplified by Open-Flamingo and Flamingo models, over traditional LLMs such as the OPT models in the task of few-shot HMD. This difference could stem from the unique challenges that meme text and visual content interplay present, which may be particularly divergent from the data LLMs were trained on. With the limited exposure provided by few-shot examples, LLMs struggle to develop the nuanced reasoning required to decode the complex interplay of text and imagery in memes. In contrast, our method, which also utilizes LLMs, introduces a substantial enhancement by incorporating LoRA modules. These LoRA modules allow the model to master fundamental components vital for identifying hateful memes. Once these foundational skills are established, our model can adeptly adapt to the task of HMD by composing these pre-trained modules, each performs a specific reasoning task correlated to the detection task.

Number of shots. Despite increasing the number of training examples, both baseline models and our proposed method exhibit a plateau in performance, with some configurations even showing a decline. For example, Flamingo-9B with 32 shots does not surpass its 8-shot counterpart, and similarly, OpenFlamingo-9B's performance does not improve when increasing from 4 to 8 shots. This suggests that the task of HMD remains challenging within a few-shot framework, and simply adding more examples does not necessarily equate to better model performance. This phenomenon indicates that the complexity of understanding and detecting nuances in hateful memes may not be adequately addressed through quantity alone. Hence, there's a clear need for more innovative approaches to effectively leverage few-shot examples. Further research might explore alternative few-shot learning techniques, more advanced model architectures that can better capture the subtleties of multimodal data, or novel data augmentation methods that enhance the model's exposure to varied examples within the constraints of few-shot learning.

6.3.4 Ablation Study

In our ablation study, focusing on the 4-shot setting due to space constraints, we examine the contributions of different modules in our modularized networks. The results, detailed in Table 6.3, reveal that the integration of all three modules surpasses the performance of any two-module combination in terms of both accuracy and AUCROC, affirming the value of each module.

A noteworthy observation is the isolated hate-speech module's superior performance on three datasets regarding AUCROC. We found all its predictions to be *non-hateful*. This suggests the module learned with hate speech detection data is biased. A plausible reason is that hate-speech module alone is incapable of understanding the interactions between meme image and meme texts (e.g., visual metaphors). The results underscore the necessity of the additional modules for complex multimodal understanding in broader applications.

We further visualize importance scores of modules, as shown in Table 6.4. The scores do not reach the maximum due to our L1 normalization penalty that mitigates instability by avoiding extreme values in importance scores. Interestingly, even though the meme-comprehension module registers the lowest importance score, its absence negatively impacts performance. This indicates a possible underdevelopment of this module, likely due to the low resolution and noise in online-sourced memes. Enhancing the meme-comprehension module's capability through improved training on challenging data could potentially increase its contribution and overall model performance.

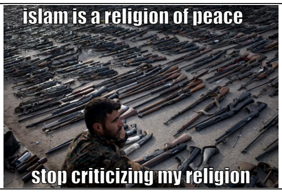
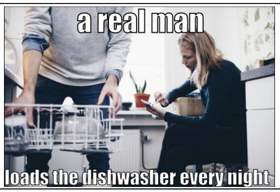
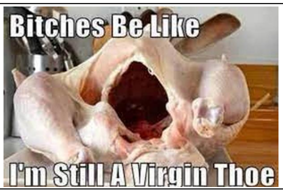
Meme	 islam is a religion of peace stop criticizing my religion		
Ground Truth	Hateful (Religion)	Non-hateful	Hateful (Gender)
hate-speech	No	No	No
hate-interp	It mocks the muslims for their violent nature.	It dehumanizes the females as less capable humans that are only good for dishwashing.	It insults the females by suggesting that they are only virgins because they are stupid.
meme-comp	Meme poster is trying to convey that Islam is a religion of violence.	Meme poster is trying to convey that men are supposed to do the dishes while women are supposed to do the laundry.	Meme poster is trying to convey that women who claim to be virgins are lying.
meme-comp, hate-speech	No	No	No
meme-comp, hate-interp	Yes, it is hateful.	True	True
hate-speech, hate-interp	Yes	No	No
Mod-HATE	Yes	No	Yes

Table 6.5: Visualization of predictions from individual modules, the compositions of two modules and from our modolarized networks. Incorrect prediction in red.

6.4 Qualitative Analysis

In this section, we conduct case studies to better understand the strengths and limitations of our proposed method.

Case Study Table 6.5 shows the predictions from individual modules, the composition of two modules, and the full Mod-HATE model of three example memes. The case studies shed light on the nuanced role that each module plays in both the identification and interpretability of hateful content within memes. It is noted that the modules dedicated to hateful meme interpretation and meme comprehension contribute significantly to the model's accuracy. These components not only aid in detection but also enhance the model's explanatory power (e.g., the outputs from hate-interp and meme-comp in the first example), offering a window into the model's decision-making process.

However, when these modules are used in isolation, their effectiveness diminishes, as they tend to provide descriptive explanations of the content rather than clear-cut classifications. This aligns with their underperformance when they stand alone, reinforcing the idea that the integration of modules is crucial for optimal functioning. A particular bias is detected in the hate-interp module (e.g., the second example), which has a propensity to incorrectly interpret content as hateful due to its training on exclusively hateful examples. This issue is mitigated when the module operates within the integrated framework of the Mod-HATE model, balancing out its predispositions.

The example also points out a challenge in the hate-interp module's ability to generalize to the diverse and often visually complex memes encountered in real-world scenarios, as shown in the third example. In contrast, the meme-comp module, which is trained on actual social media data, displays a more refined understanding of such content, including memes laced with visual metaphors. The hate-speech module's efficacy appears limited to situations where the cross-modal reasoning required is straightforward, struggling otherwise with more intricate multimodal interactions (e.g., the first and the third example).

In summary, the case study reveals that while individual modules possess their own strengths and limitations, their amalgamation leads to a synergistic improvement in the model's performance. This composite approach not only bolsters the model's detection capabilities but also augments its interpretability, offering a more comprehensive solution to the challenge of HMD.

6.5 Conclusion

In this chapter, we study the problem of HMD in the few-shot setting, where only a few labeled training examples are available. We propose a modularized networks which train a set of PTMs capable of relevant tasks to HMD and learn a composition of PTMs with the few-shot examples. Compared with standard in-context learning for few-shot HMD, our proposed method is more efficient as the few-shot examples will not serve as inputs during inference time which greatly reduces the computation costs. Our proposed method also outperformed all previous in-context learning on three benchmarks, demonstrating the effectiveness of the proposed method.

Chapter 7

Using a Composition of Frozen PTMs to Explicitly Conduct Multi-step Reasoning

VQA, the task of answering textual queries based on information contained in an image, is a multimodal task that requires comprehension and reasoning of both visual and textual content [54, 73]. Most previous work on VQA either trains VQA models from scratch (e.g., Fukui et al. [45], Anderson et al. [4]) or fine-tunes PT-VLMs for VQA (e.g., Li et al. [101], Lu et al. [117]). Thus, they rely heavily on labeled VQA data, which are expensive to obtain. VQA models based on supervised learning are also hard to generalize to new domains or new datasets [190, 22, 203].

Recently, large-scale PTMs have demonstrated strong transferability to different downstream tasks under zero-shot settings, i.e., without any training data for the downstream tasks [15, 143]. With increased pre-training data size, these models show strong zero-shot performance on various down-stream tasks, such as image classification and face detection with the CLIP model [143] and sentiment analysis and commonsense question answering with the GPT-3 model [15]. However, few studies have focused on zero-shot VQA from PTMs.

Despite the power of these PTMs, it is not straightforward to directly apply them to VQA under zero-shot settings, because they are not pre-trained with the same objective as VQA (noted the statement has been made by 2022). Some recent work converts images to tokens that PT-LMs can understand so that VQA can be converted to text-based QA [193, 171, 174, 77, 32]. However, this approach requires either a strong pre-trained image captioning model that can capture sufficient visual details or auxiliary training to obtain such a captioning model. Some other work converts VQA into a multimodal matching problem so that PT-VLMs such as CLIP can be used [161, 156]. However, complex VQA questions such as those found in the GQA dataset [73] often require spatial reasoning and/or multi-step reasoning, which PT-VLMs may not be strong at [163, 170].

VQA questions can be complicated and often require different reasoning steps such as object detection and spatial reasoning, as the example question in Figure 7.1 illustrates. Previously, people proposed Neural Module Networks [5, 69], which are modularized networks where each pre-defined module performs a specific reasoning task. These pre-defined modules are trained end-to-end from labeled VQA data. Motivated by the idea of modularization, in this chapter, we propose a modularized zero-shot network for VQA (**Mod-Zero-VQA**) by decomposing questions into sub-tasks and assigning appropriate sub-tasks to PTMs without any adaptation. Given a question, we first parse the question into basic reasoning steps explicitly. These reasoning steps will then be reconfigured and mapped to different PTMs based on a set of rules we define. Specifically, we consider the following PTMs: *OWL* [126] as the object detector, *MDETR* [80] for reference expression localization (including several skills such as relational and spatial reasoning) and *CLIP* [143] as the answer generator for open-ended questions. Considering the limited capabilities of current PT-VLMs in spatial relation understanding [163], we also define simple and general heuristics to aid spatial reasoning. Note that only when we decompose questions and reasoning chains step by step can we insert human heuristics for spatial reasoning, because we have the intermediate outputs such as objects' bounding boxes from previous steps.

This method aligns with the **Frozen-Composition** strategy for using PTMs as mentioned in Section 1.3. Specifically, here we consider the skills of object detection, relational reasoning, spatial reasoning and object/attribute recognition in VQA ($N_1 = 4$). Next, we select three PTMs ($N_2 = 3$) and design a mapping to convert identified sub-tasks into acceptable objectives of PTMs. Given a complex VQA question, we decompose the question into the reasoning chain of these sub-reasoning tasks, map the chain of sub-tasks to a sequence utilization of PTMs and leverage the sequential outputs from PTMs for the final answer prediction. The novelty is to leverage a composition of frozen PTMs to explicitly conduct multi-step reasoning in VQA, which has not been explored previously. Besides, the other contribution is to identify essential reasoning skills, select appropriate PTMs and design a mapping to sub-reasoning tasks to selected PTMs.

In the rest part of the chapter, we will first introduce background about the zero-shot VQA task and existing works and their limitations in Section 7.1. Next, we will describe the proposed model in Section 7.2. Then we evaluate the proposed model on two VQA datasets and report the results in Section 7.3. Besides, qualitative results about the visualization of explicit reasoning chains by Mod-Zero-VQA will be provided in Section 7.4. Finally, we conclude this work in Section 7.5.

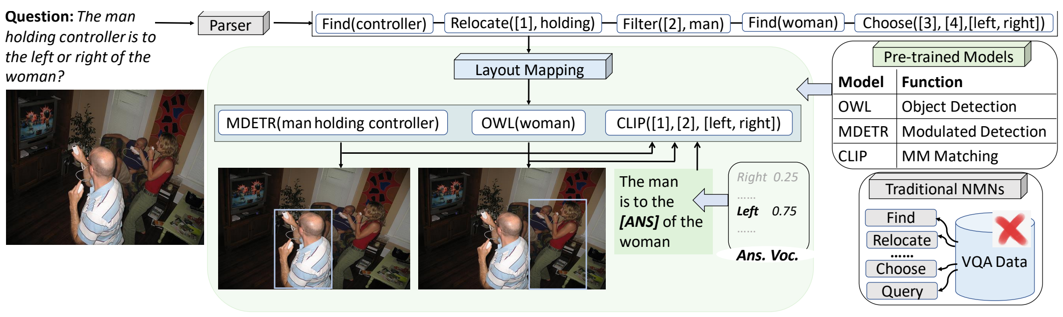


Figure 7.1: An overview of our proposed method. Instead of training modules in NMN, we propose a modularized zero-shot VQA method leveraging pre-trained models to perform different reasoning tasks.

7.1 Background

Task Definition. Given an image I and a question Q, a VQA system is expected to return an answer a. Traditional fully supervised VQA relies on a training set consisting of (image, question, answer) triplets. For zero-shot VQA, no such training data is given. However, in this proposal we assume that we can use PTMs to help us with zero-shot VQA.

Existing Zero-shot VQA Methods. Work on zero-shot VQA is very limited. We can organize existing work into the following categories. One line of work leverages the question answering capability in pre-trained language model (LMs). Some of them adopt prefix language modeling with weakly-supervised data other than VQA data (i.e., image-text pairs) to convert visual information into discrete tokens (prefix) that LMs can understand. Frozen [174], VLKD [32] and FewVLM [77] fall under this category. Some directly convert VQA images into textual descriptions so that the task of VQA changes to text-based QA and LMs can be applied. Methods in this category include PICa [193] and PnP-VQA [171]. Recent work [161, 156] converts VQA to an image-text matching problem and prompts the CLIP model [143], a large-scale vision-language model pre-trained on the image-text matching task. The prompts can be either question irrelevant such as *Quesion: `[Ques]`; Answer: `[MASK]`* (QIP by Shen et al. [156]) or question-related by converting questions into a masked statement (TAC-P by Song et al. [161]).

However, a limitation with these methods is that several of them still require training, although the training data is not in the form of VQA. Besides, converting images to captions and leveraging text-based QA may lose important visual details during the caption generation step. The two methods above using CLIP do not address the issue that CLIP model lacks compositional and spatial reasoning abilities, which has been observed in previous work [163, 170]. None of them considered explicit multi-step reasoning in VQA.

7.2 Mod-Zero-VQA Model

Our method is motivated by Neural Module Network (NMN) based VQA, which decomposes questions into reasoning steps, where each module in the NMN is pre-defined to perform a specific reasoning task. The idea allows us to select appropriate PTMs to handle different reasoning tasks in a question. Specifically, in NMN-based VQA, we first manually define a set of reasoning steps such as object detection and spatial reasoning, each represented by a *module*. A question is then explicitly decomposed and converted into a *layout* of modules, which is an executable program showing the reasoning chain to reach the final answer. The top section of Figure 7.1 shows the layout corresponding to the sample question. To train an NMN-based VQA system, usually a layout generator is separately built first, which either uses hand-crafted rules over dependency parses of questions or is a trained seq2seq model. Then, the parameters of the various VQA modules are learned from VQA training data.

For our work, we do not want to use VQA data for training. But we observe that many modules in NMN-based VQA can be supported by PTMs that have already acquired the capabilities needed by these modules. The key component of our method is therefore to map a layout of modules produced by traditional NMN-based VQA to a simplified layout of zero-shot components that can be implemented directly using PTMs.

7.2.1 Traditional VQA Modules

There is not any standard set of modules for VQA. We largely adopt the design of modules introduced by Hu et al. [69] with some minor changes. We assume that the image has been pre-processed and N bounding boxes have been detected, each represented as an embedding vector, collectively denoted as $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N)$. An attention map $\boldsymbol{\alpha}$ is defined to be a distribution over the N bounding boxes.

Table 7.1 lists the most important traditional VQA modules that we will replace with PTMs. The full list of modules can be found in Table D.1 in the appendices. It is worth explaining that besides taking in $\mathbf{V}$ and $\boldsymbol{\alpha}$ as either input or output, many modules also take in the word embeddings of some text description extracted from the question. These text embeddings are arguments to control the behaviors of the modules. For example, the `Find` module's objective is to locate an object among all the bounding boxes given. The textual input $\mathbf{g}_{\text{OBJ}}$ is therefore the word embedding of the name of the object to be found. Similarly, $\mathbf{g}_{\text{RELA}}$,$\mathbf{g}_{\text{ATTR}}$ and $\mathbf{g}_{\text{QUERY}}$ are word embeddings for the description of relation (e.g., *to the left of*), attribute (e.g., *red*) and aspect to query (e.g., querying *name*).

Module	**Inputs**
`Find`	$\mathbf{V}$, $\mathbf{g}_{\text{OBJ}}$
`Relocate`	$\boldsymbol{\alpha}$, $\mathbf{V}$, $\mathbf{g}_{\text{RELA}}$
`Filter`	$\boldsymbol{\alpha}$, $\mathbf{V}$, $\mathbf{g}_{\text{CONDI}}$
`Choose`	$\boldsymbol{\alpha}_1$, $\boldsymbol{\alpha}_2$, $\mathbf{V}$, $\mathbf{g}_{\text{RELA}^1}$, $\mathbf{g}_{\text{RELA}^2}$
`Query`	$\boldsymbol{\alpha}$, $\mathbf{V}$, $\mathbf{g}_{\text{QUERY}}$

Table 7.1: A subset of the modules in traditional NMN that we replace with pre-trained models. Modules in the first block output an attention map and those in the second block generate an answer.

Traditionally, the parameters of the modules in Table 7.1 need to be learned from VQA training data. In other words, these modules' underlying capabilities such as object recognition and relational reasoning need to be acquired from VQA data. However, we hypothesize that recently developed PTMs may already have some of these capabilities and can therefore directly equip these modules with such capabilities. For example, the `Find` module is mainly responsible for object recognition, and previously the parameters of `Find` have to be learned from scratch using VQA data. Now with a powerful pre-trained model such as OWL [126] that can recognize a wide range of objects, we can presumably directly use a model like OWL to replace the traditional `Find` module.

7.2.2 Pre-trained Models

We utilize three PTMs that we believe are highly relevant to VQA.

OWL. The Vision Transformer for Open-World Localization (OWL) model [126] is a model for open-vocabulary object detection. It is first pre-trained on large-scale image-text pairs and then fine-tuned with added detection heads and medium-sized detection data. Given the category name of an object and an image, the model is able to locate bounding box(es) in the image containing the object together with a confidence score for each box.

MDETR. The modulated DETER (DEtection TRansformer) model [80] is an end-to-end detector that can detect an object in an image conditioned on a piece of textual description of the object such as its attributes and its relation with another object in the image. The model is pre-trained on image-text pairs with explicit alignment between phrases in the text and bounding boxes of objects in the image. Given an image and the description of an object, MDETR is able to locate the bounding box(es) in the image containing the object satisfying the description. Note that different from OWL, MDETR is able to understand textual descriptions that may contain attribute information and/or complex visual relations. For example, given the description *a man holding a yellow cup is talking*, MDETR will detect the bounding box containing the man holding a yellow cup in the given image, whereas OWL is not able to use the description and will only recognize all bounding boxes containing a man. Note that we use the version of MDETR pre-trained on general modulated detection **without fine-tuning** for any downstream tasks.

CLIP. CLIP is a well-known large-scale vision-language model by OpenAI. It is pre-trained with 400M image-caption pairs through contrastive learning. Given an (image, text) pair, CLIP uses its separate image encoder and text encoder to turn the image and the text each into a vector, and the cosine similarity between the two vectors directly measures the compatibility of the two. Recent work has shown that CLIP can be directly used for VQA in a zero-shot setting, if we can come up with a set of candidate answers and transform each (question, answer) pair into a statement [161].

7.2.3 Zero-shot NMN using Pre-trained Models

Based on the descriptions of the traditional VQA modules in Section 7.2.1 and of the three PTMs we consider in Section 7.2.2, we can see that there are obvious connections between the capabilities desired by the traditional modules and the capabilities that these PTMs have already acquired.

However, the mapping between them is not trivial. First of all, there is no simple one-to-one mapping from traditional VQA modules to the PTMs. For example, the MDETR model can already perform multiple steps of reasoning to locate the desired object, so it can be used to cover a sequence of modules in an NMN layout. Second, there may be capabilities required when applying PTMs but not captured by modules defined in NMN-based VQA. In particular, the MDETR model always assumes that the object to be grounded exists in the given image, but for those questions asking for the existence of a specified object, we cannot directly use MDETR.

To address these challenges, we carefully design a mapping mechanism that can map an NMN-based module layout to a simplified layout consisting of a few zero-shot modules. Three of these zero-shot modules (`OWL`, `MDETR` and `CLIP`) correspond exactly to the three PTMs introduced earlier. The rest of the zero-shot modules are defined by simple heuristic rules. We list these zero-shot modules in Table 7.2.

We now give a high-level summary of the mapping mechanism below. We first look at the last module in the NMN layout. If the last module is one of `Choose`, `Compare` and `Query`, we know that the input to this last module is either a single attention map or two attention maps, where each attention map essentially tries to capture an object matching some textual descriptions. By tracing the path in the layout leading to the attention map, we choose either the zero-shot `OWL` module

Module	**Inputs**	**Output**
`OWL`	I, OBJ	$\mathcal{B}$, s
`MDETR`	I, SENT	$\mathcal{B}$, s
`CLIP`	$\mathcal{B}, I, \mathcal{V}$	Ans.
`Count`	$\mathcal{B}$	Num.
`Exist`	$\mathcal{B}$, (ATTR/RELA)	*Yes/No*
`And`	$\texttt{Exist}_1, \texttt{Exist}_2$	*Yes/No*
`Or`	$\texttt{Exist}_1, \texttt{Exist}_2$	*Yes/No*

Table 7.2: Zero-shot modules with either pre-trained models or heuristics. The I is the VQA image, $\mathcal{V}$ is the answer vocabulary and $\mathcal{B}$ is the set of bounding boxes.

(when the path has a length of 1) or the zero-shot `MDETR` module (when the path is longer than 1 hop). This is because when the path length equals to one, it involves only object detection (corresponding to a single `Find` module in the NMN layout for generation of the attention map). When the path length is more than one, it indicates the generation of the attention map in the NMN layout involves other modules such as `Filter` and `Relocate`, which calls for the other abilities than object detection, such as language understanding, attribute recognition and relational reasoning. Different from NMN modules which takes in image features and object embeddings to generate an attention map, our zero-shot `OWL` and zero-shot `MDETR` takes in the raw image and raw texts to locate (OBJ for `OWL` and SENT for `MDETR`) to generate a set of detected bounding boxes $\mathcal{B} = \{\mathbf{b}_n\}_{n=1}^{N}$ together with their confident scores $\mathbf{s} \in \mathbb{R}^N$, where $\mathbf{b}_n \in \mathbb{R}^4$ represents the relative position and size of the detected bounding box in the image. We keep only the bounding box from either `OWL` or `MDETR` with the highest confident score and feed it to `CLIP`. We generate an answer by leveraging the capability of multimodal matching of CLIP. Specifically, given $\mathcal{B}$, we generate an input image (which we refer to as I^{in}) by either masking regions not containing those detected boxes ($|\mathcal{B}| = 2$) or cropping the image so that only the part containing the box remains ($|\mathcal{B}| = 1$). If the final NMN module is `Choose`, we generate a masked template by question conversion as in [161]; otherwise the masked template will be a simple "[`MASK`]". Then we match the image I^{in} with the template where the [`MASK`] token is replaced by each of the answer candidates in $\mathcal{V}$. We then select the answer that, when placed inside the template, best matches the image.

If the module is `Exist`, we trace back the path leading to `Exist` to determine whether the module is asking for the existence of an object, an attribute or a relation. For object existence (e.g., *is there a car*), we use the zero-shot `OWL` module. For attribute existence and relation existence, we first verify whether all mentioned nouns (objects) detected by a POS tagger in the question exist with the `OWL` module. Once we detect an object that does not exist, the predicted answer will be *no*. If all objects exist, then we generate corresponding bounding boxes leveraging either `OWL` or `MDETR` following the method described in the paragraph above. For attribute existence, we generate a pair of a positive and a negative descriptions: (ATTR, *not* ATTR), e.g., (*red*, *not red*). We then find which description aligns better with the cropped image according to $\mathbf{b}$. If the image aligns better with the positive statement, then the answer will be *yes*; otherwise, *no*. For relation existence, we generate the masked image I^{in} according to $\mathbf{b}_1$ and $\mathbf{b}_2$ (the bounding boxes of the two objects whose relation is to be checked) and a pair of opposite statements regarding the relation to be checked, following [161]. For example, if the question is to check whether *A* is holding *B*, the two opposite statements will be *A is holding B* and *A is not holding B*. For both attribute and relation existence, we use zero-shot `CLIP` for the alignment between the input image and the statements. More details and the work flows of existence-related questions are provided in Appendix D.3.

If the module is `Count`, we directly count the number of bounding boxes in $\mathcal{B}$ returned either from `OWL` or `MDETR`. Finally, if the last module is a logical `AND` or logical `OR`, we further trace to the inputs of this module, which should both be an `Exist` module. We then use the same mechanism described above for `Exist` to process the module. By receiving the outputs from the `Exist` modules, logical operations will be applied to determine the output. The deterministic logical operations can be found in Appendix D.2.

7.2.4 Spatial Heuristics

As mentioned in [163], CLIP is less capable of spatial reasoning. Using CLIP for answer generation may not be enough when it involves spatial relation understanding. Following [163], we define simple and general heuristics to perform certain types of spatial reasoning. Note that only when we decompose questions explicitly can we insert the spatial heuristics into CLIP-based answer generation because we have the intermediate outputs from previous reasoning steps.

First of all, given the coordinates and the size of a bounding box, we use manual rules (named as **SpD**) to decide its position in the image as *left, right, bottom, top*. Besides, we define heuristics, denoted as **SpC**, to solve spatial relations between two bounding boxes (e.g., *to the left of* and *to the right of*).

Details of the implementation of the spatial relation solvers can be found in Appendix D.4.

7.3 Experiment Results

7.3.1 Dataset

We evaluate the proposed modularized zero-shot VQA method on two benchmarks: GQA [73] and VQAv2 [54]. The GQA dataset consists of questions requiring multi-step reasoning and various reasoning skills. Around 94% of the questions require multiple reasoning steps. We regard it as the main dataset to demonstrate the effectiveness of the proposed method compared with the baselines. Compared with GQA, questions on the VQAv2 dataset require fewer reasoning steps and are of diverse semantics. We use VQAv2 to show the validity of our method in real-world VQA. We report standard accuracy for the GQA dataset while soft accuracy [54] for VQAv2 dataset as there are multiple ground-truth answers. We report the statistics of the datasets in Appendix D.5.

Method	GQA	VQA
Frozen	-	29.5
VLKD$_{\text{ViT-L/14}}$	-	42.6
FEWVLM$_{\text{base}}$	27.0	43.4
FEWVLM$_{\text{large}}$	29.3	47.7
PNP-VQA$_{\text{6M}}$	34.6	54.3
PNP-VQA$_{\text{11B}}$	**41.9**	**63.3**
QIP	35.9	21.4
TAP-C	36.3	38.7
Mod-Zero-VQA	**47.3**	**41.0**

Table 7.3: Experimental results on the GQA and VQA datasets. The first block are models using the text-based QA capability of LMs and the second blocks are models incorporating CLIP.

7.3.2 Implementation Details

We conduct experiments on NVIDIA Tesla V100 GPU. The thresholds for the OWL and the MDETR model to filter out detected bounding boxes of low confident scores are set to be 0.2 and 0.7 respectively. We follow [161] for the generation of the answer vocabulary $\mathcal{V}$ for open-ended questions. More details about answer vocabulary generation can be found in Appendix D.7 and more information about experiment settings can be found in Appendix D.7.

7.3.3 Main Results

Zero-shot VQA performance of the baselines mentioned in Section 7.1 and our proposed method are summarized in Table 7.3[1].

First of all, we observe that the proposed Mod-Zero-VQA method is more effective on the GQA dataset, which contains many multi-step reasoning questions. Mod-Zero-VQA clearly surpasses all baselines on GQA. The results suggest that it is effective under zero-shot settings to decompose questions when questions are compositional and require several steps of reasoning to reach the answer. Such decomposition allows us to take advantage of the capabilities of different PTMs. We also test the validity of the proposed method on real-world VQAv2 dataset, where

[1]For FEWVLM and PNP-VQA model, we show their reported performances on GQA test-dev, which should have similar distributions as the validation split of GQA.

Detector	Yes/No Qns	Other Qns	Overall
CLIP-FR	56.80	33.82	41.39
OWL	69.26	36.48	47.28
GT	76.48	38.06	50.72

Table 7.4: Performance of Mod-Zero-VQA with different object detectors on GQA.

questions require fewer reasoning steps and of diverse semantics. We can see that our method still achieves the best performance among zero-shot methods that utilize CLIP. Although better performance is achieved by several methods that utilize large language models (as shown in the first block of Table 7.3), it is worth pointing out that these methods often require caption generation as a pre-processing step, and this step poses challenges. For example, PNP-VQA generates 100 captions per question, which is laborious. There may also be redundancy because many captions are irrelevant for question answering. Another advantage of our Mod-Zero-VQA method over the other zero-shot baselines is that our method offers high interpretability by showing the explicit multi-step reasoning chain, which has not been considered by any previous work. With question decomposition, we can design modularized networks and assign reasoning tasks to PTMs which are more capable of the tasks, and with more powerful PTMs coming out, our method can be easily extended to utilize newer and more effective PTMs. Meanwhile, it is easier to pinpoint the weakest chain in a system and insert human heuristics to aid these modules.

7.3.4 Ablation Study

In our Mod-Zero-VQA method, PTMs play an important role. In this section, we show the performance of Mod-Zero-VQA when we replace PTMs listed in Section 7.2.2 with alternative models.

Replacing OWL: We tried replacing OWL with other object detectors. First, we consider an object detector combining Faster-RCNN [146] and CLIP (**CLIP-FR**). Specifically, Faster-RCNN is used to detect objects in an image and CLIP is applied to classify each detected object. Second, we use the ground-truth object annotations

Method	PT-VLMs	Overall
QIP	$\text{CLIP}_{\text{ViT-B/16}}$	35.93
	$\text{CLIP}_{\text{Res50}\times16}$	35.11
	ALBEF	34.75
TAP-C	$\text{CLIP}_{\text{ViT-B/16}}$	36.32
	$\text{CLIP}_{\text{Res50}\times16}$	38.16
	ALBEF	38.36
Mod-Zero-VQA	$\text{CLIP}_{\text{ViT-B/16}}$	47.28
	$\text{CLIP}_{\text{Res50}\times16}$	46.49
	ALBEF	**48.68**

Table 7.5: Performance of the Mod-Zero-VQA model with different PT-VLMs as the zero-shot `CLIP` for answer generation on GQA.

from Visual Genome [88] to replace object detection results (**GT**), which serves as an upper bound. Results of our zero-shot NMNs with different object detectors are provided in Table 7.4. We divide the questions into Yes/No (bindary) questions and other questions. We observe that the quality of object detection is important to the performance of zero-shot NMNs. Our model with OWL surpasses the one with CLIP-FR, which has poorer detection performance than OWL. We also observe more substantial performance drop with binary questions. We believe that this is because these questions are mostly about the existence of objects, so the object detection results affect the VQA performance more. Using Mod-Zero-VQA with the ground-truth object detection results would further improve the performance, as shown in the last row of Table 7.4. This suggests that when more accurate object detection models are developed, we can further improve the zero-shot VQA performance with our approach.

Replacing CLIP: We show the performance of replacing zero-shot `CLIP` (which is $\text{CLIP}_{\text{ViT-B/16}}$ by default in our experiments), with either $\text{CLIP}_{\text{Res50}\times16}$ or ALBEF [98], in Table 7.5. Because QIP and TAC-P convert VQA to a multi-modal matching task and both use PT-VLMs as the answer generator, we also replace the original $\text{CLIP}_{\text{ViT-B/16}}$ in these two baselines with the other PTMs. We observe that Mod-Zero-VQA gives stable performance regardless of the vision-language model used, and it always outperforms the baselines substantially. This indicates that these PTMs can all be good substitutes for the zero-shot `CLIP` module. Compared with the two CLIP models (i.e., with either ViT [40] or ResNet [62] as the visual backbone), we also notice that using ALBEF [98] as the answer generator can enhance the performance. To better understand the advantage of using ALBEF over CLIP, we provide more detailed performance in Table D.3 in Appendix D.8. ALBEF mostly benefits the proposed method in the *Query* type of questions, which usually ask about *objects*, *attributes* and *relations*. Consistent with [206], end-to-end models (i.e., ALBEF in this case) perform better than dual-encoder models (i.e., CLIP in this case) in vision understanding tasks on average. A future direction may be to select the best pre-trained model per question.

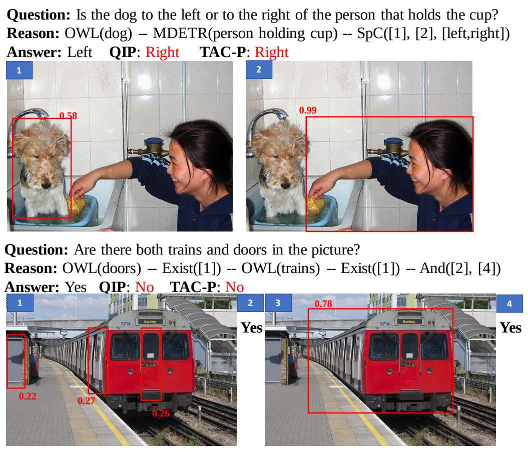
7.3.5 Out-of-Domain Generalization

Because our Mod-Zero-VQA method is not trained on any domain-specific VQA data but rather utilizes PTMs that are supposedly trained on data from a wide range of domains, we suspect that our Mod-Zero-VQA method is more robust across different domains compared with VQA models trained on specific domains and applied in cross-domain settings. We therefore also compare our Mod-Zero-VQA with fully-supervised models in the Out-of-Domain Generalization (OOD) setting. Specifically, we consider an OOD setting where test images are related to scenes not observed during training. We first identify a set of scene-related objects and restrict all training images to only those that do not contain these objects. For example, in the *Indoor* OOD setting, none of the training images should contain *sofa*, *bed* or any of the other objects that we have identified to be related to *Indoor* scenes. To build fully-supervised VQA models for comparison, we consider (1) **BUTD** [4], a classic two-stream VQA models, (2) traditional **NMNs** [5], and (3) finetuned PT-VLMs, including **VilBert** [117], **VisualBert** [101] and **ALBEF** [98].

The results are shown in Table 7.6. We can see from the table that for those supervised VQA models, when they are trained on images with different scenes, their performance on the target domain is clearly lower than our Mod-Zero-VQA.

Method	Indoor	Food	Street
BUTD	39.27	32.28	35.96
NMNs	39.45	32.47	36.05
VilBert	39.87	32.12	36.68
VisualBert	41.14	33.47	38.51
ALBEF	45.55	38.87	41.60
Mod-Zero-VQA	48.86	47.80	49.54

Table 7.6: Comparison between our Mod-Zero-VQA method and fully-supervised VQA models under the out-of-domain setting.


Question: Is the dog to the left or to the right of the person that holds the cup?
Reason: OWL(dog) -- MDETR(person holding cup) -- SpC([1], [2], [left,right])
Answer: Left **QIP**: Right **TAC-P**: Right

Question: Are there both trains and doors in the picture?
Reason: OWL(doors) -- Exist([1]) -- OWL(trains) -- Exist([1]) -- And([2], [4])
Answer: Yes **QIP**: No **TAC-P**: No

Figure 7.2: Visualization of intermediate outputs from reasoning steps of the Mod-Zero-VQA model.

Furthermore, our Mod-Zero-VQA method achieves steady performance across different scenes, whereas the supervised VQA models give fluctuated performance in different scenes. This demonstrates the robustness of our proposed method.

7.4 Qualitative Analysis

As a case study, we visualize the outputs of the reasoning steps from the proposed method and compare the prediction of the proposed method with those of QIP and TAC-P, which also leverage CLIP as the answer generator. We show two example questions and the outputs in Figure 7.2. Both questions require multiple reasoning steps.

We can see that our method gives the correct predictions while the two other methods answer wrongly. We can also see that by decomposing the questions, our method assigns each sub reasoning task to a pre-trained model capable of the task (i.e., MDETR for reference expression localization and OWL for object detection). With question decomposition, we can also better pinpoint the weaknesses of PTMs and insert human knowledge by defining simple but general heuristics (e.g., adding spatial heuristics to zero-shot `CLIP` and defining logical operations). More examples with visualization are provided in Appendix D.7.

7.5 Conclusion

In this chapter, we propose a modularized zero-shot VQA method, motivated by the idea of Neural Module Network (NMN). Instead of training modules in NMN with VQA data, we decompose questions into reasoning tasks explicitly, leverage PTMs and assign proper reasoning tasks to them. Experiments show that our model is powerful on questions requiring multi-step reasoning and applicable for real-world VQA. Besides, the proposed model is highly interpretable, which helps to pinpoint weaknesses of a VQA system, making it easier to improve a system. Our model highlights a future direction of leveraging PTMs for other complicated tasks requiring multiple reasoning capabilities.

Chapter 8

Explicit Generation and Incorporation of Knowledge with Frozen PT-LMs

In the previous chapter, we propose a zero-shot VQA model for generic VQA. In this chapter, we delve into a special VQA setting, Knowledge-based VQA (which we refer to as K-VQA in this chapter), where in addition to an image, external knowledge is needed to answer the given question. For instance, to answer the question in Figure 8.1, background knowledge about national parks in California is needed.

Early methods for K-VQA follow a *retrieve and answer* paradigm (Figure 8.1(a)), which first retrieve knowledge from external knowledge sources as additional input and then train a VQA model through supervised learning [179, 130, 131, 97]. This paradigm requires both a suitable external knowledge base and a large amount of K-VQA training data, which may not be practical for real applications when either of these resources is not available. Recently, with the fast advances of LLMs that have demonstrated remarkable zero-shot transfer capabilities, several studies applied LLMs for K-VQA under zero-shot or few-shot settings, leveraging both the extensive knowledge implicitly contained in LLMs and their built-in question

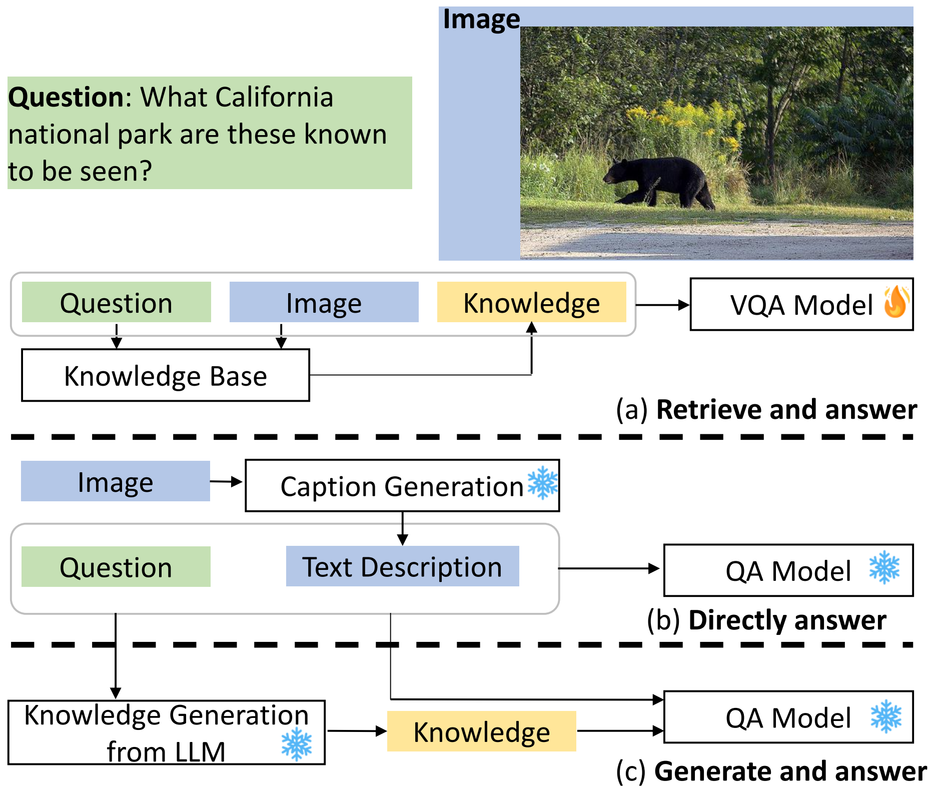


Figure 8.1: Three approaches to K-VQA: retrieve and answer, directly answer, and generate and answer.

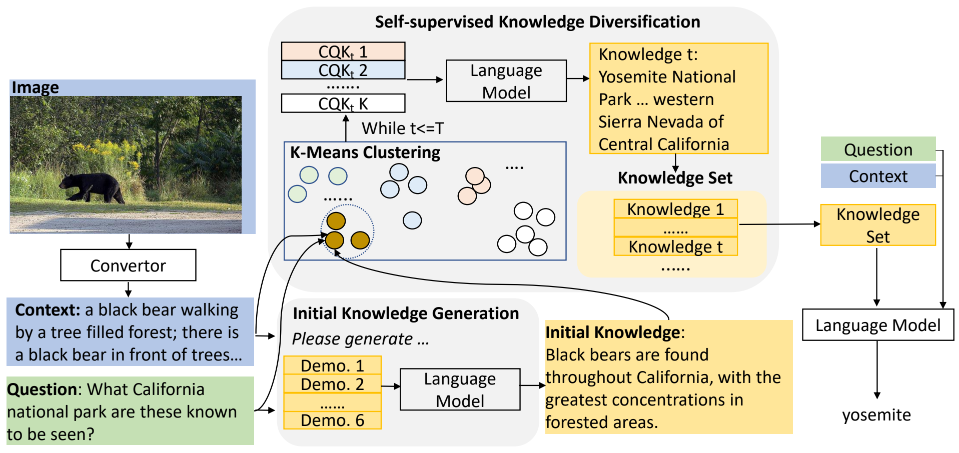
answering capability [193, 70, 58, 100, 2]. Typically, these methods first convert an image to text descriptions (i.e., captions) and then feed the captions and the question into an LLM to directly obtain the answer, as illustrated as the *directly answer* paradigm in Figure 8.1(b).

However, none of these zero-shot or few-shot methods *explicitly* states the knowledge needed to answer a question. As we know, answering K-VQA questions usually requires external knowledge not seen in the image. Even if the external knowledge is implicitly contained in the LLM used for QA, it is not immediately clear whether and how the LLM can use the relevant knowledge to answer a K-VQA question through the *directly answer* paradigm. On the other hand, recent work has shown that for text-based QA that requires multi-step reasoning, explicitly generating relevant knowledge and including it as additional input improves QA performance [112, 198]. We suspect that this is also the case for K-VQA. Furthermore, explicitly generated knowledge improves the explainability of the system. Another limitation of previous zero-shot and few-shot K-VQA methods is that some of them rely on task-specific training such as the training of a question-specific caption generation model in PromptCap [70], which still requires significant amount of training data.

In this chapter, we attempt to address these limitations of previous work. Inspired by Liu et al. [112], which uses an LLM to generate explicit knowledge statements to facilitate text-based commonsense QA, we propose a similar zero-shot K-VQA method that uses an LLM (specifically GPT-3) to *explicitly* generate potentially useful knowledge statements to facilitate K-VQA, as illustrated in Figure 8.1(c). In addition to having explicit knowledge statements, our method is also free from any additional training. To improve the diversity and coverage of the generated knowledge, we further borrow the self-supervised knowledge diversification strategy from [198]. We call our method KGENVQA. To the best of our knowledge, we are the first to test the *generate and answer* approach on K-VQA.

KGENVQA is under the **Frozen-Composition** strategy for using PTMs as mentioned in Section 1.3. Specifically, here we consider two sub-tasks in K-VQA: relevant knowledge generation and knowledge incorporation for answer prediction ($N_1 = 2$). Then we leverage two PT-LMs for each task respectively ($N_2 = 2$). Furthermore, we design proper techniques for leveraging PT-LMs for each task. The contribution of KGENVQA is first explicitly consider the K-VQA as two reasoning steps, which has not be studied before. Besides, we design strategies to map two reasoning steps to proper tasks for PTMs.

We evaluate KGENVQA on both OK-VQA [122] and A-OKVQA [154], two benchmark datasets commonly used for K-VQA. The experiments demonstrate that our generated knowledge statements are effective in improving the K-VQA performance in terms of answer accuracy, when everything else being equal, and our method can outperform SOTA zero-shot K-VQA methods that do not use extra training. We also measure the usefulness of our generated knowledge and find that the generated knowledge statements have high quality in terms of grammaticality, relevance, factuality, helpfulness, and diversity, based on manual judgement. Our



Self-supervised Knowledge Diversification
CQK$_t$ 1
CQK$_t$ 2
....
CQK$_t$ K
Language Model
Knowledge t: Yosemite National Park ... western Sierra Nevada of Central California
While t<=T
K-Means Clustering
Knowledge Set
Knowledge 1
Knowledge t
Image
Convertor
Context: a black bear walking by a tree filled forest; there is a black bear in front of trees...
Question: What California national park are these known to be seen?
Initial Knowledge Generation
Please generate ...
Demo. 1
Demo. 2
Demo. 6
Language Model
Initial Knowledge: Black bears are found throughout California, with the greatest concentrations in forested areas.
Question
Context
Knowledge Set
Language Model
yosemite

Figure 8.2: An overview of the proposed method. We first convert the image into textual descriptions and prompt LLMs with the question and manual demonstrations to obtain the initial knowledge pieces. In the second stage, we diversify knowledge by selecting a diverse set of knowledge statements in the first step as demonstrations. Lastly, we incorporate the generated knowledge for QA with a language model.

findings demonstrate that *generate and answer* is a feasible zero-shot approach to K-VQA with the additional benefit of providing explanations through the explicitly generated knowledge statements.

8.1 KGENVQA Model

8.2 Method

The high-level idea of our KGENVQA method is to leverage an LLM to generate explicit knowledge statements given an image and a question. These knowledge statements can then be combined with the image captions and the question to be passed to the same or a different LLM for zero-shot text-based QA. In this section, we first elaborate how we generate knowledge statements from an LLM using few-shot in-context learning. We then present how the generated knowledge is integrated into the question answering process.

8.2.1 Knowledge Generation

Our knowledge generation process consists of two steps: An *initial knowledge generation* step, in which we generate a single knowledge statement for each (image, question) pair in the K-VQA test dataset, and a subsequent *self-supervised knowledge diversification* step, in which we sample a diverse set of knowledge statements generated during the first step as in-context demonstrations to perform a second round of knowledge generation, in which we generate multiple knowledge statements per (image, question) pair. The motivation is that with a diverse set of in-context demonstrations, we expect the LLM to also generate knowledge statements covering different aspects of the same (image, question) pair, which may increase the chance of getting the correct answer.

Caption generation. In both knowledge generation steps, we regard an LLM (GPT-3 in our experiments) as a knowledge base because the LLM has been trained on a large amount of text covering a wide range of topics. Previous work has shown that relevant knowledge statements can be generated from an LLM if appropriate text prompts including both the contexts and some demonstrations are used [112]. However, different from text-based QA, for K-VQA, the context is an image, which cannot be directly used as input to an LLM. To address this issue, we adopt a simple solution that converts the image into one or more captions, using an off-the-shelf image captioning model. However, instead of using a general-purpose captioning model, we believe that *question-aware* captions, which focus on describing the parts of the image that are more relevant to the question, can provide better contexts for knowledge generation. Therefore, we adopt the question-aware caption generation mechanism by Tiong et al. [171], which first highlights image regions that are more relevant to the question and then generates question-aware captions with the attention-weighted image. Following the practice of Tiong et al. [171], we use multiple captions because this practice has been shown to be useful for subsequent question answering. We concatenate the multiple captions into a single sequence of tokens, which we denote as C.

Prompt template for knowledge generation. In both the initial knowledge generation step and the knowledge diversification step, to generate a single piece of knowledge, we use the following prompt template: *Please generate related background knowledge to the question*; *Context:* [C]; *Question:* [Q]; *Knowledge:*. The LLM will complete the prompt above by generating a sentence, which we treat as a knowledge statement. In order to better generate the relevant knowledge, we leverage in-context learning by including a few demonstrations, i.e., a few examples each containing a context (which are also image captions), a question, and the expected knowledge statement to be generated. During the initial knowledge generation step and the knowledge diversification step, we use different kinds of demonstrations.

Initial knowledge generation. During the initial knowledge generation step, we use six manually crafted in-context demonstrations for knowledge generation. They can be found in Appendix E.8. During this step, we generate a single knowledge statement for each (image, question) pair in a K-VQA test dataset.

Self-supervised knowledge diversification. Previous work showed that proper selection of demonstrations is of vital importance when prompting LLMs [193, 53]. We suspect that the manually crafted demonstrations may not always be proper examples for all test instances. Besides, when answering knowledge-intensive questions, oftentimes more than one piece of knowledge may be needed. For instance, to answer the question in Figure 8.2, the knowledge 1) what national parks are in California; 2) among national parks in California, which is famous for black bears. To generate multiple knowledge statements per question, a straightforward solution is to ask the LLM to return multiple pieces of knowledge. However, beam search sampling, as mentioned in [66, 177], tends to generate dull and repetitive outputs, and the improved top-k sampling [41] can only solve the issue to some extent. On the other hand, with different prompts, an LLM may generate diverse outputs [106].

Therefore, we adopt a self-supervised knowledge diversification strategy by [198] as follows. Let $\mathcal{K}_{\text{init}} = \{(C_i, Q_i, K_i)\}_{i=1}^{N}$ denote the set of (captions, question, knowledge statement) triplets obtained during the initial knowledge generation step, where K_i is the knowledge statement generated for (C_i, Q_i). We treat each triplet (C_i, Q_i, K_i) as a "silver"-labeled demonstrating example. Slightly different from [198], we hypothesize that if each time we sample a different set of the triplets from $\mathcal{K}_{\text{init}}$ as demonstrating examples for knowledge generation, and we repeat this T times for a given (image, question) pair (I, Q), then we can obtain T diversified knowledge statements for (I, Q). To further ensure that every time the demonstrating examples themselves are diverse, we first use K-means clustering to cluster the triplets in $\mathcal{K}_{\text{init}}$. Denote these K clusters as $\mathcal{K}_{\text{init}}^1, \mathcal{K}_{\text{init}}^2, \ldots, \mathcal{K}_{\text{init}}^K$. To generate T final knowledge statements for a given (I, Q) pair during the knowledge diversification step, we repeat the following process T times: (1) we randomly select one triplet from each $\mathcal{K}_{\text{init}}^k$, except the cluster the given (I, Q) pair belonging to, to form $K - 1$ demonstrating examples; (2) we use these $K - 1$ demonstrations as in-context examples to generate a knowledge statement for (I, Q), using the prompt template as described earlier. We call this strategy *self-supervised* knowledge diversification because we do not require any human to annotate diversified demonstrating examples. We will empirically compare this cluster-based strategy with a random demonstration selection strategy in our experiments. Details of how K-means clustering is done can be found in Appendix E.1.

8.2.2 Knowledge Integration for K-VQA

With the final set of T knowledge statements generated for each (image, question) pair, we can combine them with the image captions and the question, and pass them to a pre-trained text-based QA model for answer generation. In our experiments, we use UnifiedQA [83], OPT [204] and GPT-3 [15].

8.3 Experiment Results

8.3.1 Datasets and Evaluation Metrics

To validate our proposed method, we choose two commonly used K-VQA benchmark datasets, namely, OK-VQA [122] and A-OKVQA [154]. Questions in OK-VQA need outside knowledge beyond the images to answer. A-OKVQA is an augmented version of OK-VQA that requires additional types of world knowledge. Because the ground-truth answers of the *test-split* of A-OKVQA are not available, we use its *val-split* for evaluation. In the end, the OK-VQA and A-OKVQA datasets we use contain $5,046$ and $1,100$ questions, respectively. We report the soft accuracy [54] on both datasets as there are multiple ground-truth answers for a question. Due to the limit of space, implementation details are provided in Appendix E.2.

8.3.2 Zero-shot Methods for Comparison

In this work, we focus on zero-shot K-VQA. There are models that need extra training (with labeled data other than K-VQA data). There are also some few-shot K-VQA methods where the few shots are dynamically selected from a large pool of training examples, which means they still need much training data. For fair comparison, we do not include these methods because they are not strictly zero-shot.

Below we briefly review three existing zero-shot K-VQA methods that we compare with:

PICa [193] converts images into captions with an off-the-shelf caption generator, CLIPCap [127]. The captions are regarded as contexts and fed to GPT-3 together with the question for answer prediction.

PNP-VQA [171] uses improved caption generation by exploiting an image-text matching model [99] to highlight image regions related to the question. The attended images are then used for caption generation with BLIP [99] so that the captions are question-aware. We adopt the same caption generation method in PNP-VQA in our method. PNP-VQA uses UnifiedQA [83], a pre-trained question answering model,

Model, Size		Setting	OK-VQA	A-OKVQA
U.QA	0.7B	*w/o* KGen	32.3	29.0
		w KGen	39.7	31.6
	3B	*w/o* KGen	39.6	35.5
		w KGen	44.5	36.5
	11B	*w/o* KGen	43.7	38.9
		w KGen	45.4	39.1
OPT	6.7B	*w/o* KGen	35.2	32.4
		w KGen	39.2	35.9
	13B	*w/o* KGen	37.3	35.1
		w KGen	40.2	36.0
	30B	*w/o* KGen	37.7	34.4
		w KGen	42.2	38.1

Table 8.1: Performance comparison between using and not using generated knowledge. KGen refers to knowledge generation. **U.QA** is short for UnifiedQA.

in a fusion-in-decoder (FiD) manner [75], for final answer prediction.

Img2LLM [58] follows the caption generation process in PNP-VQA. Based on the captions, it generates synthetic QA pairs as demonstrating examples when prompting the LLM for final answers. OPT [204] is used as the LLM for QA.

8.3.3 Main Results

In this section, we empirically evaluate our *generate and answer* approach in two ways: (1) We test the usefulness of the generated knowledge for K-VQA by systematically comparing our K-VQA system with and without knowledge generation. (2) We compare our *generate and answer* method with SOTA zero-shot K-VQA baselines, which do not explicitly generate knowledge.

The effect of knowledge generation. We first conduct systematic experiments to compare the *generate and answer* approach and the *directly answer* approach based on our own implementation. To see whether knowledge generation can consistently help K-VQA, we experiment with three different pre-trained QA models: UnifiedQA, OPT, and GPT-3. We choose these models because they are used in previous zero-shot K-VQA methods, namely, PNP-VQA, Img2LLM, and PICa, respectively. When

LLM	**Num. Kn.**
w/o Gen. Kn.	39.6
LLaMA$_{7B}$	42.1
LLaMA$_{13B}$	42.5
GPT-3	44.5

Table 8.2: Results on OK-VQA when using generated knowledge from different models. *w/o Gen. Kn.* denotes without using any generate knowledge. The text-based QA model is UnifiedQA$_{3B}$.

using UnifiedQA, we follow Tiong et al. [171] and adopt the FiD strategy. When using OPT, we follow Guo et al. [58] and add synthetic QA pairs as demonstrations.[1]

We first show the results of UnifiedQA and OPT on both datasets in Table 8.1. We can see that under all settings (with different QA models and different model sizes), using the generated knowledge consistently improved the final accuracy of the answers. For GPT-3, due to the API cost, we only use the first 500 questions in OK-VQA for performance comparison. We find that on these 500 test examples, the answer accuracy increased from $\mathbf{27}.\mathbf{4}$ to $\mathbf{34}.\mathbf{1}$, after adding generated knowledge.

Recently, a few open-source LLMs such as LLaMA [172] have demonstrated comparable performance to GPT-3. We have also considered LLaMA as an alternative choice to GPT-3 for knowledge generation. We incorporate the generated knowledge into UnifiedQA$_{3B}$ for answer prediction. The results from using LLaMA generated knowledge are provided in Table 8.2. According to the results, we can conclude that incorporating generated knowledge from open-source LLMs also benefits K-VQA. By increasing the size of the LLMs, the generated knowledge can more effectively facilitate the model to arrive at the final prediction. In summary, the results demonstrate that the *generate and answer* approach consistently outperforms the *directly answer* approach on both benchmark datasets under different settings.

Although our main focus is the zero-shot setting, we also experiment with the few-shot setting, and we find that there is consistent improvement of the *generate and answer* approach over the *directly answer* approach in the few-shot setting,

[1]We used the authors' code for synthetic QA pair generation. However, due to different implementation details and the different numbers of synthetic QA pairs used, the performance of our re-implemented Img2LLM base model differs from the reported performance.

Model	**Accuracy**
Previous Zero-shot Models ***without*** *Extra Training*	
$\text{PICa}_{\text{zero,175B}}$	17.7
$\text{PNP-VQA}_{\text{0.7B}}$	27.1
$\text{PNP-VQA}_{\text{3B}}$	34.1
$\text{PNP-VQA}_{\text{11B}}$	35.9
$\text{Img2LLM}_{\text{6.7B}}$	38.2
$\text{Img2LLM}_{\text{13B}}$	39.9
$\text{Img2LLM}_{\text{30B}}$	41.8
KGenVQA (Ours)	
$\text{UnifiedQA}_{\text{3B}}$	44.5
$\text{UnifiedQA}_{\text{11B}}$	**45.4**
OPT_{30B}	42.2
Zero-shot Models ***with*** *Extra Training*	
$\text{BLIP-2(OPT)}_{\text{6.7B}}$	36.4
$\text{BLIP-2(FlanT5}_{\text{XL}})_{\text{3B}}$	40.7
$\text{BLIP-2(FlanT5}_{\text{XXL}})_{\text{11B}}$	45.9
$\text{Flamingo}_{\text{3B}}$	41.2
$\text{Flamingo}_{\text{9B}}$	44.7
Few-shot Models (n=1)	
$\text{PICa}_{\text{few,175B}}$	40.8
$\text{PromptCap}_{\text{175B}}$	**48.7**

Table 8.3: Comparison with SOTA on OK-VQA.

indicating the generalization of our method to few-shot settings. Details of our few-shot experiments can be found in Appendix E.3.

Comparison with SOTA. Next, we compare our method with the state-of-the-art models. Because we focus on zero-shot K-VQA without extra training, we only compare with previous models of this nature. The comparison is shown in the top half of Table 8.3 for OK-VQA and top half of Table 8.4 for A-OKVQA. We can observe the following from the tables: (1) On both datasets, our *KGenVQA* performs better than the zero-shot baselines when model sizes are comparable. For example, on OK-VQA, our UnifiedQA 3B surpasses all previous zero-shot baselines, i.e., baselines shown in the first block of Table 8.3. On A-OKVQA, our UnifiedQA 3B only loses out to Img2LLM 30B, but this is expected because of huge difference of model size. Our method with larger model sizes (i.e., our UnifiedQA 11B and OPT 30B) outperform all zero-shot baselines without extra training.

Model	**Accuracy**
Zero-shot Models ***without*** *Extra Training*	
Img2LLM$_{6.7B}$	32.3
Img2LLM$_{13B}$	33.3
Img2LLM$_{30B}$	36.9
KGenVQA (Ours)	
UnifiedQA$_{3B}$	36.5
UnifiedQA$_{11B}$	**39.1**
OPT$_{30B}$	38.1
Few-shot Models (n=10, 32 respectively)	
PICa$_{few}$	18.1
PromptCap$_{175B}$	**56.3**

Table 8.4: Comparison with SOTA on A-OKVQA.

We also show those zero-shot models with extra training (e.g., BLIP-2 [100], Flamingo [2]) and few-shot learning models (e.g., PICa$_{few}$ [193] and PromptCap [70]). It is worth noting that strictly speaking, PICa$_{few}$ [193] and PromptCap [70] do not use the same set of few shot examples (i.e., is not few-shot learning in the traditional sense) because these two methods dynamically sample demonstrating examples from the whole K-VQA training set for each test example. Because of their benefits from either extra training or access to the entire training set, we place these models in a different category, at the bottom half of Table 8.3 and Table 8.4. Compared with these models, we can see that our KGenVQA models still surpass some models with extra training, such as BLIP-2 (FlanT5$_{XL}$) and the powerful 3B Flamingo, and achieve comparable results with 9B Flamingo, demonstrating the effectiveness of our model compared with state-of-the-art models. Even comparing with few-shot models, we observe that our best performance is higher than PICa$_{few}$ [193] and is comparable to PromptCap$_{175B}$.

It may be worth noting that on OK-VQA, PICa$_{zero}$ performs poorly probably because it uses a single image caption. In order to make a fair comparison with PICa$_{zero}$, we provide results of our method with a single image caption and without image descriptions (i.e., with generated knowledge only) in Appendix E.4. The results show steady improvements (about 16 percentage points in terms of absolute

Case	Num. Kn.	OK-VQA
Manual	1	35.9
Random	10	41.8
CoT	1	37.5
KGen	10	**44.8**

Table 8.5: Comparison of different knowledge generation methods on OK-VQA. "Num. Kn." is the number of knowledge statements used.

accuracy) on OK-VQA.

8.3.4 Ablation Studies

Knowledge generation method. We first compare our cluster-based knowledge diversification strategy with (1) using the manual prompt generated knowledge, i.e., a single piece of knowledge (Manual); (2) randomly sampling $K-1$ single knowledge statement, instead of sampling from different clusters, from the initially generated knowledge statements, $\mathcal{K}_{\text{init}}$ for knowledge diversification in the second stage (Random). Besides, we consider the idea of Chain-of-Thoughts (CoT) [185], which generates explanations before the answer generation. In K-VQA, the needed knowledge can also be regarded as a kind of explanations. Therefore, we test the widely used CoT for knowledge generation, which is an alternative to our cluster-based knowledge generation approach. We re-use the six manual demonstrations as mentioned in Chapter 8.1 and manually add answers to the questions (i.e., each demonstration consists of contexts of image descriptions, a question, a piece of related knowledge and an answer). Together with these demonstrations, we prompt GPT-3 [15] to first generate the relevant knowledge and then the answer (CoT). Due to the cost of calling GPT APIs, we only apply CoT to a subset questions on OK-VQA (200 questions). We show model performance, based on UnifiedQA$_{3B}$, with different ways of knowledge generation and show results in Table 8.5. We have a few observations: (1) using initial generated knowledge with demonstrations offers improvements but no better than KGen. This may be that fixed manual demonstrations fail to generate diverse knowledge. For a fair comparison, we also consider using a

QA Model	Num.	OK-VQA
UnifiedQA (FiD)$_{3B}$	0	39.6
	5	**44.5**
	10	**44.5**
	20	42.7
OPT$_{13B}$	0	37.3
	5	**40.2**
	10	37.2
	20	37.2
GPT-3	0	27.4
	5	**34.1**
	10	32.4
	20	31.7

Table 8.6: Performances with different numbers of knowledge statements.

single piece of knowledge from KGen, which achieves **38.8**, indicating the need of diverse prompts in knowledge generation. (2) Comparing using random selection and cluster-based selection in the self-supervised knowledge diversification stage, we find that using the cluster-based method clearly outperforms random selection, which may not generate diverse knowledge. Overall, the cluster-based knowledge generation method is better than the other methods for knowledge generation in term of K-VQA performance; (3) When we compare the CoT knowledge generation with cluster-based knowledge generation, the second method significantly wins CoT in terms the benefit to K-VQA, probably because the cluster-based method has higher chances of facilitating answer generation with diverse knowledge; Besides, we also compare the direct CoT-generated answers from GPT-3 with answers generated when prompting GPT-3 for QA incorporating our generated knowledge. Our generated knowledge results in an accuracy of 32.0 while CoT-generated knowledge leads to 29.3.

Number of knowledge statements. Next, we test how the number of knowledge statements affects the performance, using UnifiedQA$_{3B}$ (FiD), OPT$_{13B}$ and GPT-3. Due to the API costs, we choose OK-VQA as the experiment dataset for this ablation study. For GPT-3 as the QA model, we test the performance on the first 500 questions. The results are reported in Table 8.6. Intuitively, we observe improvements after

Case	Gram.	Rel.	Fact.	Help.
Ours_{max}	100.0	100.0	96.3	90.0
Ours_{avg}	99.0	100.0	94.5	67.0

Table 8.7: Evaluation of our generated knowledge in terms of four evaluation metrics.

adding more generated knowledge at first and then decrement of performance. This is probably because adding too many pieces of knowledge may potentially add noisy or redundant knowledge, which harms the performance. Besides, we notice that decoder-only models have smaller optimal number of knowledge statements than encoder-decoder FiD model. This is probably because decoder-only models (i.e., OPT and GPT-3) may have difficulty in understanding the long concatenated sentence while FiD is specifically designed for comprehension of multiple documents.

8.3.5 Evaluation of the Generated Knowledge

In this section, we conduct human evaluation to exam the quality of the generated knowledge. We follow [112] and sample 40 cases from OK-VQA dataset where the correctness of the answers would be changed (i.e., either from correct to wrong or wrong to correct) after adding the generated knowledge. For each instance, we sample 5 knowledge statements for evaluation. We ask two annotators to check the quality of the generated knowledge in terms of the evaluation metrics below. To ensure objectiveness, annotators will not know whether the predictions are changed to become correct or wrong.

Evaluation metrics. Following [112, 157], we take four metrics for evaluating generated knowledge: 1) *Grammatically*: whether it is grammatical 2) *Relevance*: whether it is related to answering the question and the image; 3) *Factuality*: whether it is factual; 4) *Helpfulness*: whether it is helpful so that it directly leads to the correct answers or provides indirect but supportive information of the correct answers. For *helpfulness*, we adopt three categories of evaluation: helpful (i.e., provides direct or indirect supportive information to correct answers), harmful (i.e., negates correct answers or support incorrect answers) or neutral (neither helpful or harmful). Besides the previously used metrics, we also consider *Diversity* as the fifth evaluation criteria, indicating the coverage of generated knowledge. Details about the definitions can be found in Appendix E.9 and the examples we provide to annotators regarding the four evaluation metrics are included in the supplementary materials.

Results. The average agreement from two annotators over four evaluation metrics is 0.67, in terms of *Fleiss Kappa* κ [90]. It indicates substantial agreement among annotators. For each criterion, we report the average score over two annotators. We consider two evaluation settings for generated knowledge: 1) *average*: taking the average scores over five pieces or knowledge; 2) *max*: take the maximum score over scores of five knowledge. The results are provided in Table 8.7. According to the results, most knowledge is grammatical, relevant to questions and factual. One interesting thing is that the generated knowledge may be relevant to questions but harmful for final answers, as the average score in term of *helpfulness* is only around 70. From the comparison with *average* and *max* scores of human evaluation, we further verify the need of knowledge diversification, which can raise the chance of generating helpful knowledge, as indicated by the maximum score of *helpfulness*, which means how likely the generated knowledge will lead to the correct answer. For diversity, we compare the five pieces knowledge generated by cluster-based selection against random selection. The average diversity of cluster-based select is $\mathbf{3.4}$, while $\mathbf{2.5}$ for random selection. It shows cluster-base selection results in more diverse knowledge, which is more likely to cover information for answering questions. It is in consistency with results in Table 8.5.

8.4 Qualitative Analysis

To better understand the advantage of our method, we compare our method with the baseline, UnifiedQA$_{3B}$ (FiD), without generated knowledge. We analyze the first 20 cases, without cherry picking, where our method answers correctly while the baseline gives wrong predictions. Among the 20 error cases of the baseline, 85%

are due to the lack of external knowledge, highlighting the advantage of our method. Due to the limitation of space, we provide the examples in Appendix E.7.

Besides, we conduct error analysis to better understand the limitations of our method. We conduct an empirical analysis for the error cases by manual checking 40 error cases from UnifiedQA$_{3B}$ (FiD) after adding generated knowledge. Among all error cases, we observe 20% are due to the undesired knowledge. Due to limitation of space, we provide visualization of the error cases in Appendix E.6. The main cause of generating misleading knowledge comes from the inaccurate image descriptions which lack details for LLMs for knowledge generation. It implies with the development of better image description generation tools, our method can be potentially improved.

8.5 Conclusion

In this chapter, we propose to generate relevant knowledge from PT-LMs, specifically LLMs, for zero-shot K-VQA. We evaluate the effectiveness of the generated knowledge by experimenting with different pre-trained QA models of varying model sizes on two K-VQA benchmarks. The experiment results show that the generated knowledge improves K-VQA performance, and our method can outperform SOTA zero-shot K-VQA methods. We further conduct human evaluation to validate the quality of the generated knowledge. The results demonstrate that the generated knowledge statements are relevant and helpful to questions in K-VQA.

Chapter 9

Conclusion and Future Directions

In this chapter, I will first summarize the contributions of the work presented in the previous chapters. I will then present future directions to be explored to better utilize PTMs for VLU tasks.

9.1 Summary of Contributions

This thesis attempts to design innovative approaches to optimize the use of PTMs for VLU tasks, with a specific focus on HMD and VQA. We categorize our proposed methods regarding whether we adopt the fine-tuning or zero-shot/few-shot learning and whether we use a single PTM or a composition of PTMs. In other words, we have explored four categories of using PTMs: Tuning-Single, Frozen-Single, Tuning-Composition and Frozen-Composition, for HMD and VQA.

In Chapter 3, we proposed the DisMultiHate model with the Tuning-Single strategy, by incorporating task-specific components. It leveraged the pre-trained BERT model [36] and tuned it so that it learned target-aware textual representation that can disentangle target entities. To achieve the goal, we used a self-supervised training diagram. The learned target-aware textual and visual representations were used for HMD. The good performance on two benchmarks demonstrated the effectiveness of the model.

In Chapter 4, we seek to leverage implicit knowledge in PT-LMs to aid HMD, which frequently requires external background knowledge. Since PT-LMs are inherently textual, the method involves a frozen pre-trained vision-language model (PT-VLM) for converting meme images into textual captions. Then the multimodal classification task is converted to a masked language modeling task. We further give two demonstrations to provide contextual information and prompt a PT-LM for the prediction. This piece of work exploits both the Tuning-Single and Frozen-Single strategy. The proposed model showed significant performance improvements over baselines, highlighting the superiority of the proposed model.

However, the work in Chapter 4 suffers from non-informative captions during image conversion to textual descriptions. Generic image descriptions may lack crucial details, such as race and gender information, vital for detecting hateful content. In Chapter 5, we proposed a probing-based captioning approach to leverage a frozen PT-VLM for complementary. Specifically, we prompt a frozen PT-VLM with hateful content-related questions and use the answers as image captions, ensuring that the captions contain critical information for hateful content detection. The method leverages a frozen PTM for the sub-step, hateful content-related image description generation, in HMD, thus following the Frozen-Single strategy of using PTMs. When incorporating the generated hateful content-related captions into text-based HMD models, significant improvements can be achieved.

Though showing good detection performance, all methods above are fully supervised which heavily rely on large volumes of training data. The dynamic nature of hateful memes tied to evolving events, nevertheless, makes it impractical to annotate sufficient training examples. In response, we present, to the best of our knowledge, a pioneering exploration of HMD tailored to the few-shot learning setting in Chapter 6. We introduce a modularized network for HMD, by harnessing the power of a composition of tuned PTMs, each of which possesses an essential reasoning capability for HMD. This piece of work falls in the category of Tuning-Composition when using PTMs.

On the other hand, we consider the other complex VLU task, VQA. VQA questions often require multiple steps of reasoning, which is still a capability that most PTMs lack. Besides, different steps in VQA reasoning chains require different skills such as object detection and relational reasoning, but a single PTM may not possess all these skills. Third, recent work on zero-shot VQA does not explicitly consider multi-step reasoning chains, which makes them less interpretable compared with a decomposition-based approach. To address the problem, we proposed a modularized zero-shot network with the Frozen-Composition strategy of utilizing PTMs. We explicitly decompose VQA questions, convert sub reasoning tasks to acceptable objectives of PTMs and assign converted tasks to proper PTMs. The experiments on two VQA benchmarks pointed out the effectiveness of the proposed model, especially when questions involved multi-step reasoning.

Expanding our inquiry, in Chapter 8, we delved into a specific VQA scenario, K-VQA, where external knowledge apart from images is indispensable to answer questions. Recent zero-shot K-VQA models lack explicit demonstration of the knowledge used to answer questions and thus lack interpretability. We propose to explicitly generate and incorporate knowledge with frozen PT-LMs for K-VQA. The method involves one PTM for knowledge generation and one PTM for question answering with the incorporation of generated knowledge. It follows the method of Frozen-Composition when using PTMs. The method improves the K-VQA performance as well as interpretability.

9.2 Future Directions

In this section, I will provide a few future directions regarding the application of PTMs to the research field of VLU.

Making Trustworthy and Reliable PT-VLMs Although PTMs, especially PT-VLMs, largely facilitate the research of VLU, the drawbacks of PT-VLMs may also pose hurdles to the research.

One type of limitation in PT-VLMs is that existing PT-VLMs tend to hallucinate about non-existent objects or non-existent attributes when answering questions about images [105, 78, 196]. The hallucinations of PT-VLMs make them less trustworthy, hindering their applications to real-world settings.

One line of research concentrates on introducing evaluation strategies, such as evaluation datasets [105, 116] and evaluation metrics [105, 78] for better understanding and quantifying of hallucinations in PT-VLMs. The other line of work tries to mitigate hallucinations in PT-VLMs [196, 109]. Each line of research suffers from a few limitations.

The proposed evaluation datasets and evaluation metrics focus on a narrow scope of hallucinations (object hallucination and attribute hallucination) and overlook other types of hallucinations such as the hallucination of non-existent relations [105, 116]. As for the mitigation of hallucinations, some works conducted further tuning of PT-VLMs with instructional data [109, 116], which is expensive with the scaling of model sizes. Moreover, the additional instructional data is limited and focuses only on augmenting negative instructions.

To make PT-VLMs trustworthy and applicable in real-world scenarios, we should consider all categories of hallucinations and propose interpretable mitigation methods. A promising direction would be decomposing long responses from PT-VLMs into atomic statements and leveraging a composition of PTMs as examiners. The high-level idea is similar to our work in [17], whereas we decompose the models' outputs instead of questions in [17]. Specifically, given a complex response, we leverage a parser to decompose it into atomic statements, each of which states an aspect of the input image (e.g., the existence of an object, the relation between two objects, or the attribute of an object). Then, for each aspect, we apply an appropriate and expertise PTM for checking. For instance, an object detector will be used for checking the existence of an object and a scene graph generator will be used for checking the relation between two objects. With feedback from examiners, we update models' responses if there are hallucinations.

Another obvious limitation in PTMs making them less reliable is their generation of toxic contents. One of our focused VLU task is HMD, for the detection of manually created hateful content. However, we should also raise our awareness of the detection of toxic contents generated by AI systems. Previous studies about PT-LMs showed that they were likely to give toxic outputs with specific prompts [136, 186]. Recent studies [180] have found a similar tendency in PT-VLMs in generating toxic content given malicious images. Meanwhile, images can also be distractions to mislead PT-VLMs to accept malicious instructions [113]. Besides, the generation of PT-VLMs involves not only texts but potentially images. However, no existing studies have explored analyzing multimodal generation from PT-VLMs. As for mitigation of harms in PT-VLMs, there is limited work [180]. It is always important to guarantee the safety of PT-VLMs before applying them in real-world scenarios.

General-Purpose VLU Systems In this thesis, we studied two specific VLU tasks, HMD and VQA, whereas in the real-world setting, different VLU tasks may be entangled. For instance, an online vision-language chat-bot may be expected to both answering questions related to input images (performing the task of VQA) and giving warnings if users input a hateful meme (performing the task of HMD).

One potential strategy for creating a general purpose VLU system is model merging, namely leveraging a composition of PTMs, each capable of a VLU task. Additionally, a PTM serves as the leader to assign tasks to appropriated models.

The assignment could be done implicitly by merging parameters in different PTMs, each capable of a specific VLU task. The leader PTM learns to compose these PTMs by evaluating their relevance to the task mentioned in the input texts from users. This idea is similar to what we explored in Chapter 6, while calling for more PTMs considering different VLU settings. Studies about model merging would facilitate this way for constructing general purpose VLU systems.

Alternatively, the assignment could be done explicitly as well. The leader could decompose a complex task into sub-tasks and choose proper PTMs for each sub-task. This idea is similar to the piece of work in Chapter 7, however, in a more general VLU setting, calling for a set of PTMs capable of different VLU tasks.

Appendix A

Appendix for PromptHate

A.1 Experiment Settings

We train all models using Pytorch on an NVIDIA Tesla V100 GPU, with 32 GB dedicated memory, CUDA-10.2. For PTMs (i.e., BERT, RoberTa, VisualBERT), we use the package, *transformers* (version $4.19.2$) from Huggingface[1]. Table A.1 lists the parameter count for all models.

Method	**# Params (M)**
Text BERT	109.9
Image Region	1.0
Late Fusion	110.9
Concat BERT	111.8
MMBT-Region	111.5
Visual BERT COCO	111.8
ViLBERT CC	252.1
CLIP BERT	111.7
MOMENTA	71.9
DisMultiHate	115.6
FT-RoBERTa	356.4
PromptHate	355.4

Table A.1: Number of parameters in HMD models.

The learning rates of models are set empirically. For BERT based models, the learning rate is set to be 2×10^{-5}, the same as in [92]. For RoBERTa-large based models (**PromptHate** and **FT-RoBERTa**), following [50], we tested learning rate

[1]https://huggingface.co/

Model	FHM		HarM	
	AUC.	Acc.	AUC.	Acc.
ClipCap+COCO	78.72	70.20	87.25	78.38
ClipCap+CC (UC)	80.38	70.08	88.56	81.94
ClipCap+CC	81.45	72.98	90.96	84.47

Table A.2: PromptHate with different image captions.

ranging from 10^{-5} to 1.5×10^{-5} and reported the best ones. Specifically, the learning rate is set to be 1.3×10^{-5} and 10^{-5} on FHM and HarM datasets, respectively. AdamW is used as the optimizer for all models. The mini-batch size is set at 16 during training. As mentioned in Section 4.1.6, we apply the multi-query ensemble strategy. The number of querying times is set at 4 on both datasets. It takes one GPU six minutes to train and validate `PromptHate` per epoch. It takes up 19 GB dedicated memory for PromptHate training. We use 10 training epochs for both PromptHate and baselines.

A.2 Analysis for Image Captions

A key data-preprocessing step in PromptHate is to covert the image into textual captions as input for PLMs. Therefore, the quality and expressiveness of the image captions may affect the prompting and ultimately affect the hateful meme classification performance. To investigate this effect, we experiment with image captions generated with ClipCap [127] pre-trained on different datasets, namely, MS COCO [107, 26] and Conceptual Caption (CC) [155].

Table A.2 shows PromptHate's performance with image captions generated using ClipCap pre-trained on COCO (**ClipCap+COCO**) and CC (**ClipCap+CC**). We observe that the ClipCap pre-trained on CC performs better than that pre-trained with COCO. A possible reason could be that the CC dataset is mainly images from web pages and rather more similar to meme images. For instance, considering the examples in Table A.3, we notice that ClipCap pre-trained on CC provided more detailed descriptions (e.g., the relation of the man and the woman of the first meme and the characteristic of the sign in the second meme) compared to COCO. On

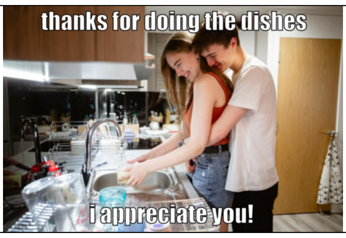
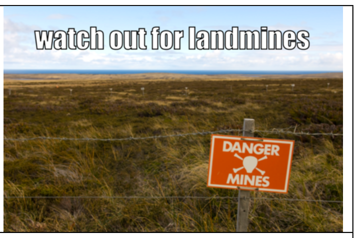
Meme		
ClipCap +COCO	a man and a woman are in a kitchen.	a sign that is on the side of a hill.
ClipCap +CC (UC)	thank you for the dishes!	a warning sign on a hillside.
ClipCap +CC	young couple in love looking at each other in kitchen.	warning sign at the entrance to the quarry.

Table A.3: Example captions generated for meme images.

the other hand, we test the generated captions using the uncleaned (i.e., without removing meme texts on images) meme images (**ClipCap+CC (UC)**). We notice that when trained with Conceptual Captions, ClipCap+CC (UC) is still able to generate some details (i.e., the characteristic of the sign in the second example). It sometimes generates comments rather than captions that describe images. It is because Conceptual Captions are from the web, and some of them are comments on meme images. Without removing texts, models will link the image to meme images and generate comments rather than captions. The difference in their performance is also significant, suggesting the importance of good quality captions in applying prompt-based models for hateful meme classification.

Appendix B

Appendix for Pro-Cap

B.1 Details for Implementation

We implement all models under the PyTorch Library with the CUDA-11.2 version. We use the Tesla V 100 GPU, each with a dedicated memory of 32GB. For models specifically implemented for HMD, we take the codes published from the author for re-implementation [1]. For PTMs which can be found under the Huggingface Library, we use the packages from Huggingface [2], specifically the BERT [36], VisualBERT [101] and the BLIP model. For ViLBERT [117], we take the released code from the authors [3]. For ALBEF [98] and BLIP-2 [100], we use the packages under the LAVIS Library[4].

For each meme image, we constrain the total length of the meme text and the generic image caption (either from the captioning model or by asking about the content of the image) to be 65. For each additional questions, we restrict its length to be shorter than 20. If the concatenation of the sentence exceeds the limited length, the sentence will be truncated, otherwise, if the sentence is shorted than the limited length, it will be padded. We set the number of training epochs to be 10 for all models.

[1]CLIP-BERT/MOMENTA: https://github.com/LCS2-IIITD/MOMENTA;DisMultiHate: https://gitlab.com/bottle_shop/safe/dismultihate; PromptHate: https://gitlab.com/bottle_shop/safe/prompthate

[2]https://huggingface.co/

[3]https://github.com/facebookresearch/vilbert-multi-task

[4]https://github.com/salesforce/LAVIS

The number of model parameters are summarized in Table B.3.

B.2 Full Ablation Study Results

Due to the limitation of space, we only show results of accuracy in ablation studies in Table 5.6. The full results including both the AUC and the accuracy are provided in Table B.4.

B.3 Visualization Cases

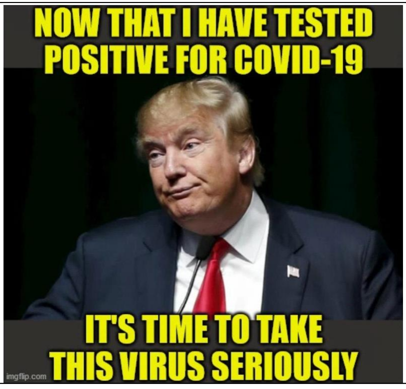
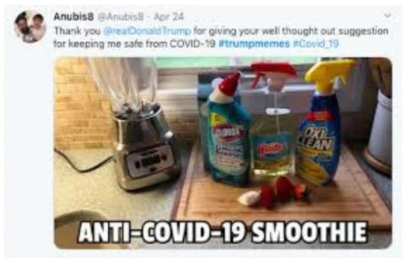
In Section 5.3, we provide visualization of cases for comparing Pro-Cap$_{\text{PromptHate}}$ with the basic PromptHate. Due to space constraints, we omit examples from the other two datasets. We provide more visualization cases in this part. The cases from the HarM dataset are illustrated in Table B.1 and the cases from the MAMI dataset are shown in Table B.2.

B.4 Results with Pro-Cap about One Target

In Section 5.2, we only report results when models use Pro-Cap from all probing questions. In this part, we report results (with entities) when using the answers from a single probing question in Table B.5.

According to the results, we observe models using answers to a single probing question are all powerful and some even surpass heuristically asking all probing questions (e.g., using the question asking about *nationality* on FHM is better than using all probing questions). It points out using all probing captions may not be the optimal solution and may generate irrelevant image descriptions. For instance, confronted with a hateful meme targeting at black people, it is meaningless to ask the religion of people in the image. Interestingly, on MAMI, when only using answers to the probing question about gender reaches teh best performance. It is because MAMI contains only hateful memes about woman. A promising direction would train the model to dynamically select probing questions essential for meme detection for different memes.

Table B.1: Comparison between Pro-Cap$_{\text{PromptHate}}$ and basic PromptHate on HarM dataset.

Meme	 NOW THAT I HAVE TESTED POSITIVE FOR COVID-19 IT'S TIME TO TAKE THIS VIRUS SERIOUSLY imgflip.com	 Anubis8 @Anubis8 · Apr 24 Thank you @realDonaldTrump for giving your well thought out suggestion for keeping me safe from COVID-19 #trumpmemes #Covid_19
Ground Truth	Hateful	Hateful
Basic Pred	Non-hateful	Non-hateful
Pro Pred	Hateful	Hateful
Meme text	now that I have tested positive for COVID-19. It's time to take this virus seriously.	Thank you reatDonald Trump for giving your well thought out suggestionfor keeping me safe from COVID-19 atrump-memes #Covid 19 Anti-covid-19 smoothie
Basic Cap-tion	i'm going to get tested for a virus!.	how to clean a kitchen sink with vinegar and food coloring.
Pro-Cap	(Generic:) trump in a suit and tie with the caption now that positive covid-19 it's time to take the virus seriously. (Race:) he is a white man in a suit and tie with a red tie and a white hat with a red hat. (Gender: he is a man in a suit and tie with a caption that says now that positive covid-19 it's time to take. (Country): us of america. (Religion: he is a christian.	(Generic:) a picture of a blender with cleaning products on it.

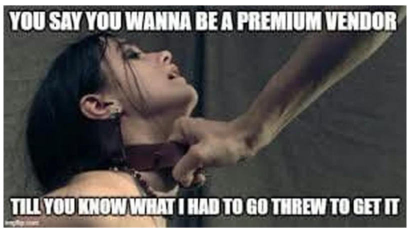
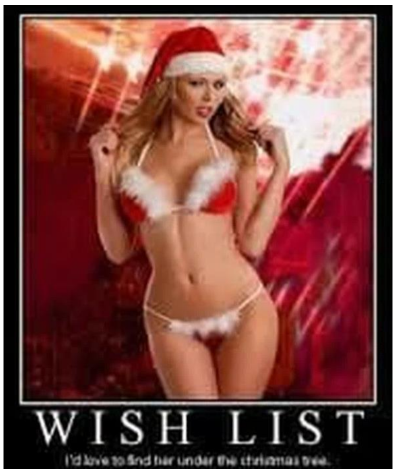
Meme	 YOU SAY YOU WANNA BE A PREMIUM VENDOR TILL YOU KNOW WHAT I HAD TO GO THREW TO GET IT imgflip.com	
Ground Truth	Hateful	Hateful
Basic Pred	Non-hateful	Non-hateful
Pro Pred	Hateful	Hateful
Meme text	you say you want to be a premium vendor until you know what I had to go threw to get it.	wish list I would love to send her under the Christmas eve.
Basic Caption	wallpaper probably with a well dressed person and a well dressed person entitled person.	western christian holiday from all of us!.
Pro-Cap	(Generic): a woman is being choked by a man. (Race): a black woman is being choked by a man in a t-shirt with a picture of a t-shirt. (Gender): a woman is being choked by a man in the image. (Country): afghanistan. (Religion): the person in the image is a christian.	(Generic): a woman in a santa claus hat posing in a bikini. (Race): a white woman in a santa claus hat posing in a sexy bikini. (Gender): a woman wearing a santa claus hat and a bikini. (Country): switzerland. (Religion): santa claus is the religion of the person in the image.

Table B.2: Comparison between Pro-Cap$_{\text{PromptHate}}$ and basic PromptHate on MAMI dataset.

Method	**# Params (M)**
Text BERT	109.9
Image Region	1.0
Visual BERT COCO	111.8
ViLBERT CC	252.1
MMBT-Region	111.5
CLIP BERT	111.7
MOMENTA	71.9
DisMultiHate	115.6
BLIP	385.0
ALBEF	209.5
BERT	109.9
PromptHate	355.4

Table B.3: Number of parameters in VQA models.

Ans. Length	**FHM**		**MAMI**		**HarM**	
Model	**AUC.**	**Acc.**	**AUC.**	**Acc.**	**AUC.**	**Acc.**
No Centric	79.08$_{\pm 0.94}$	70.08$_{\pm 1.57}$	82.26$_{\pm 0.71}$	72.78$_{\pm 0.63}$	87.04$_{\pm 0.89}$	80.11$_{\pm 1.14}$
Penalty = 1	80.76$_{\pm 1.06}$	71.94$_{\pm 0.97}$	82.53$_{\pm 0.49}$	73.06$_{\pm 0.82}$	88.34$_{\pm 0.77}$	82.09$_{\pm 1.21}$
Penalty = 2	80.87$_{\pm 0.66}$	72.28$_{\pm 0.90}$	82.27$_{\pm 0.57}$	72.91$_{\pm 1.16}$	90.25$_{\pm 0.72}$	82.85$_{\pm 1.51}$
Penalty = 3	79.62$_{\pm 0.93}$	71.40$_{\pm 1.06}$	82.36$_{\pm 0.97}$	72.47$_{\pm 0.74}$	90.25$_{\pm 0.54}$	83.25$_{\pm 1.00}$

Table B.4: Model comparison **without** any augmented image tags.

Dataset	**FHM**		**MAMI**		**HarM**	
Model	**AUC.**	**Acc.**	**AUC.**	**Acc.**	**AUC.**	**Acc.**
Race	83.63$_{\pm 0.26}$	74.28$_{\pm 1.34}$	84.00$_{\pm 0.57}$	73.51$_{\pm 1.10}$	90.43$_{\pm 0.70}$	82.26$_{\pm 1.96}$
Gender	83.91$_{\pm 0.97}$	76.08$_{\pm 1.47}$	84.34$_{\pm 1.06}$	74.21$_{\pm 0.64}$	91.05$_{\pm 0.57}$	83.16$_{\pm 1.79}$
Religion	84.85$_{\pm 0.87}$	75.52$_{\pm 1.45}$	83.90$_{\pm 0.78}$	73.95$_{\pm 0.84}$	90.86$_{\pm 0.39}$	82.15$_{\pm 1.15}$
Nationality	85.78$_{\pm 0.37}$	75.72$_{\pm 0.96}$	83.73$_{\pm 0.49}$	72.76$_{\pm 0.52}$	91.27$_{\pm 0.68}$	84.30$_{\pm 1.82}$
Disability	85.26$_{\pm 0.64}$	75.96$_{\pm 0.82}$	83.81$_{\pm 0.87}$	73.75$_{\pm 0.76}$	90.20$_{\pm 0.82}$	84.12$_{\pm 0.60}$
Animal	84.93$_{\pm 0.31}$	75.48$_{\pm 0.72}$	84.10$_{\pm 0.49}$	73.53$_{\pm 0.90}$	90.13$_{\pm 0.87}$	82.65$_{\pm 2.01}$
Pro-Cap$_{\text{PromptHate}}$	**83.58**$_{\pm 0.60}$	**75.10**$_{\pm 0.97}$	**83.77**$_{\pm 0.75}$	**73.63**$_{\pm 0.75}$	**91.03**$_{\pm 1.51}$	**85.03**$_{\pm 1.51}$

Table B.5: Model performance when only asking a single probing question.

Appendix C

Appendix for Mod-HATE

C.1 Details of Implementation

Module	**Lr.**	**Bz.**	**Epochs**
hate-speech	0.0005	16	1
meme-comp	0.0005	8	2
hate-interp	0.0005	8	2

Table C.1: Hyper-parameters for LoRA module learning. Lr. is for learning rate, Bz. is for batch size and epoch is for the number of training epochs

We implement all models under the PyTorch Library with the CUDA-11.2 version. We use the NVIDIA A40 GPU, each with a dedicated memory of 48GB. For the implementation of the OpenFlamingo model, we took the code released by the authors [9] [1]. For the implementation of LLaMA model, we leverage the HuggingFace Library [2], with the *yahma/llama-7b-hf* checkpoint [3]. The version of Huggingface is 4.33.0. For the parameter-efficient tuning with LoRA, we adopt implementation from the PEFT Library [4] of version 0.5.0. The training of LoRA modules is optimized with the Huggingface trainer. The hyper-parameters for LoRA module learning is provided in Table C.1. The ranks of all LoRA modules are set to

[1]https://github.com/mlfoundations/open_flamingo
[2]https://huggingface.co/
[3]https://huggingface.co/yahma/llama-7b-hf
[4]https://huggingface.co/docs/peft/index

be 16. We convert model parameters in LLaMA into binary float and it takes 21GB dedicated GPU memory during the inference stage with our Mod-HATE. It takes about 21GB dedicated GPU memory for training the module composer.

Model	**# Params (B)**
OPT-13B	13
OPT-30B	30
OpenFlamingo-3B	3
OpenFlamingo-9B	9
Flamingo-3B	3.2
Flamingo-9B	9.3
Mod-HATE	7

Table C.2: Number of parameters in models.

For each meme image, we constrain the length of the meme text to be 25. If the length exceeds, we will truncate the meme text. The number of model parameters are summarized in Table C.2.

C.2 Error Cases

In this section, we provide visualization of two representative error cases of our Mod-HATE in Table C.3. The first kind of error comes from inaccurate predictions from some modules, as illustrated in the first example. Though hate-interp provides good interpretation, the other modules may contribute more to the final prediction so that the meme is predicted as non-hateful. For instance, the meme-comp module fails to understand the multimodal meme. It calls for better construction of individual modules as every module matters for the final prediction. The other common error is because different memes may rely on different modules for prediction while our module composer produces the same importance scores facing up to different memes. For example, the second hateful meme can be detected correctly with a hate-speech module as there is shallow multimodal understanding. Therefore, the skills from other modules may be redundant. 2This finding has also been proven by the results in Table 6.3 and better optimization methods for generating importance scores may be needed. A better solution is that the module composer can generate instance-dependent importance scores over modules.

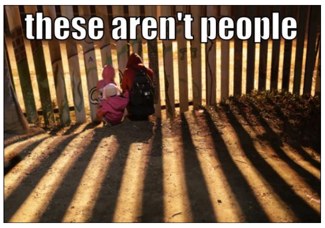
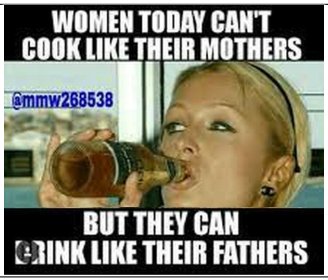
Meme		 WOMEN TODAY CAN'T COOK LIKE THEIR MOTHERS @mmw268538 BUT THEY CAN DRINK LIKE THEIR FATHERS
Ground Truth	Hateful (Nationality)	Hateful (Gender)
hate-speech	No	Yes
hate-interp	It dehumanizes the immigrants as lesser humans that are not people.	It dehumanizes the females as less capable humans that are only good for cooking and sex.
meme-comp	Meme poster is trying to convey that the two children are not people but are actually a fence.	Meme poster is trying to convey that women today can't cook like their mothers but they can drink like their fathers.
meme-comp,hate-speech	No	No
meme-comp,hate-interp	The meme is hateful because it dehumanizes the refugees.	It is hateful because it degrades women by suggesting that they are not good cooks.
hate-speech,hate-interp	No	No
Mod-Hate	No	No

Table C.3: Error cases of Mod-Hate. Incorrect prediction in red.

C.3 Templates for In-context Learning

In this section, we provide the template we used for prompting PTMs (either PT-VLMs or LLMs) in the in-context learning manner. When prompting OpenFlamingo,

# shots	**Dataset**	**H-S**	**H-E**	**M-C**
	FHM	$0.4865_{\pm 0.0339}$	$0.4561_{\pm 0.0455}$	-$0.0013_{\pm 0.0042}$
4-shots	MAMI	$0.4210_{\pm 0.0324}$	$0.4707_{\pm 0.0451}$	$0.0024_{\pm 0.0043}$
	HarM	$0.4564_{\pm 0.0454}$	$0.4532_{\pm 0.0.0453}$	$0.0025_{\pm 0.0055}$
	FHM	$0.4713_{\pm 0.0269}$	$0.3921_{\pm 0.0425}$	$0.0013_{\pm 0.0013}$
8-shots	MAMI	$0.4139_{\pm 0.0102}$	$0.4453_{\pm 0.0162}$	$0.0014_{\pm 0.0031}$
	HarM	$0.4127_{\pm 0.0132}$	$0.4512_{\pm 0.0292}$	$0.0010_{\pm 0.0012}$

Table C.4: Weights of LoRA modules of our Mod-HATE model. **H-S** for the hate-speech LoRA, **H-E** for the hate-exp LoRA module and **M-C** for the meme-captions module.

we use the template: *¡image¿User:it is an image with:* `[MEME_TEXT]` *written on it. Is it hateful? GPT: ¡answer¿*, where `[MEME_TEXT]` is a placeholder and will be replaced with the real meme text. When prompting Otter, we use the template: *¡image¿is an image with:* `[MEME_TEXT]` *written on it. Is it hateful? Answer:*. When prompting OPT after converting meme images to textual descriptions (denoted as `CAP`), the input will be: *Please decide whether the meme is hateful according to its image caption and meme text. Image Caption:* `[CAP]`*; Meme Text:* `[MEME_TEXT]` *Prediction:*. The few-shot examples will be converted into similar format of prompts and are positioned at the beginning of prompts.

C.4 Full Results of Importance Scores to LoRA Modules

We provide the full results of importance scores to LoRA modules in both 4-shot and 8-shot settings with the standard deviation in Table C.4.

Appendix D

Appendix for Mod-Zero-VQA

D.1 Modules in VQA

We summarize all modules in traditional NMNs for VQA [69, 59, 25] in Table D.1.

D.2 Logical Operations

In this section we describe the logical modules `And` and `Or`. Both of them receive outputs from two zero-shot `Exist` modules. For the `And` module, if both outputs are *yes*, it outputs *yes*; otherwise, it outputs *no*. For the `Or` module, if both outputs are *no*, it outputs *no*; otherwise, it outputs *yes*. The logical operators are deterministic.

D.3 Existence Questions

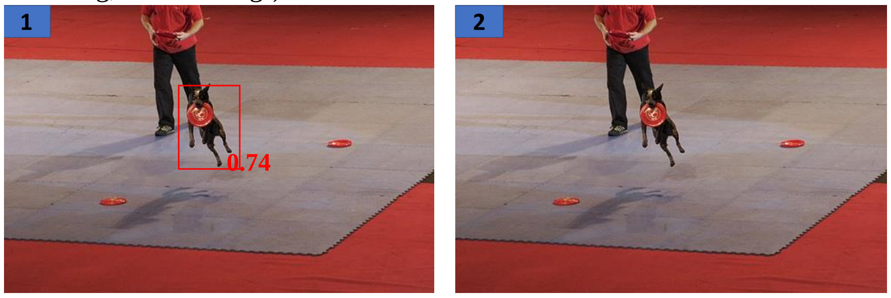
As mentioned briefly in Section 7.2.3, for questions verifying the existence of something, according to the NMN layout, we classify these questions into three types: verifying existence of objects, of attributes, and of relations. For the verification of object existence, we directly apply the zero-shot `OWL`. For both attribute and relation verification questions, we first make sure all objects mentioned in the question exist with the help of `OWL`. If any mentioned objects do not exist, the predicted answer will be *No*, as illustrated in Figure D.1. If the objects exist, we leverage either

Module	Output	Functionality
`Find`($\mathbf{V}$, $\mathbf{g}_{\text{OBJ}}$)	Att.	Locate a certain object (OBJ) in the image
`Relocate`($\boldsymbol{\alpha}$,$\mathbf{V}$,$\mathbf{g}_{\text{RELA}}$)	Att.	Transit attention from previous attention map $\boldsymbol{\alpha}$ according to the relation (RELA)
`Filter`($\boldsymbol{\alpha}$,$\mathbf{V}$,$\mathbf{g}_{\text{CONDI}}$)	Att.	Highlight objects that are attended by previous attention map $\boldsymbol{\alpha}$ and satisfy the condition (CONDI)
`Choose`($\boldsymbol{\alpha}_1$,$\boldsymbol{\alpha}_2$,$\mathbf{V}$,$\mathbf{g}_{\text{RELA}^1}$,$\mathbf{g}_{\text{RELA}^2}$)	Ans.	Choose the relation from RELA^1 and RELA^2 between highlighted regions of two attention maps
`Query`($\boldsymbol{\alpha}$,$\mathbf{V}$,$\mathbf{g}_{\text{QUERY}}$)	Ans.	Generate a final answer given the attention map, image representation and item to query (QUERY)
`Count`($\boldsymbol{\alpha}$)	Ans.	Outputs a number given the attention map of the image
`Exist`($\boldsymbol{\alpha}$)	Ans.	Output a binary answer (*yes/no*) given the attention map of the image
`And`($\boldsymbol{\alpha}_1$,$\boldsymbol{\alpha}_2$)	Ans.	Generate a binary answer (*yes/no*) given the two attention maps
`Or`($\boldsymbol{\alpha}_1$,$\boldsymbol{\alpha}_2$)	Ans.	Generate a binary answer (*yes/no*) given the two attention maps

Table D.1: The full list of modules in traditional NMNs. $\mathbf{g}_{[\cdot]}$ is the word embedding for the words in $[\cdot]$.

zero-shot `OWL` or `MDETR` to locate at objects of interests and verify the attributes and relations, with the utilization of the CLIP module. Examples are provided in Figure D.2 (for attribute verification) and Figure D.3 (for relation verification). We use CLIP for binary matching to select whether the attribute/relation exists. When multiple attributes/relations are to be verified, only when all attributes/relations exist will the predicted answer be *Yes*; otherwise, the prediction is *No*. For instance, the third example in Figure D.2 has a dark brown table, but the table is not glass, so the third step outputs *no*. The final predicted answer to the question is therefore *no*.

Question: Is the little dog catching a ball?
Reason: MDETR(little dog) – OWL(ball) -- CLIP([1], [2], [catching, not catching])



Question: Is the player wearing a hat?
Reason: OWL(player) – OWL(hat) -- CLIP([1], [2], [wearing, not wearing])

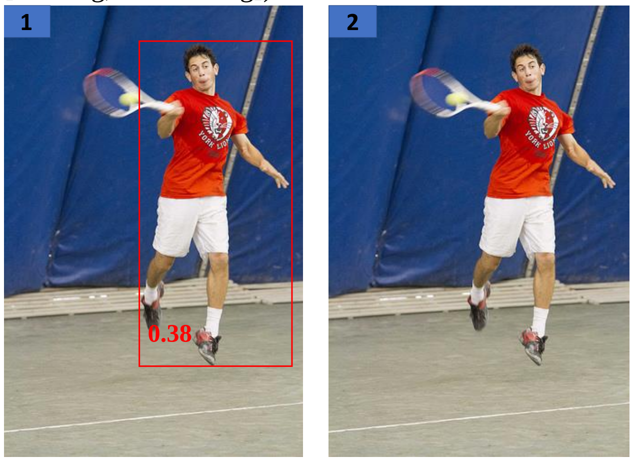
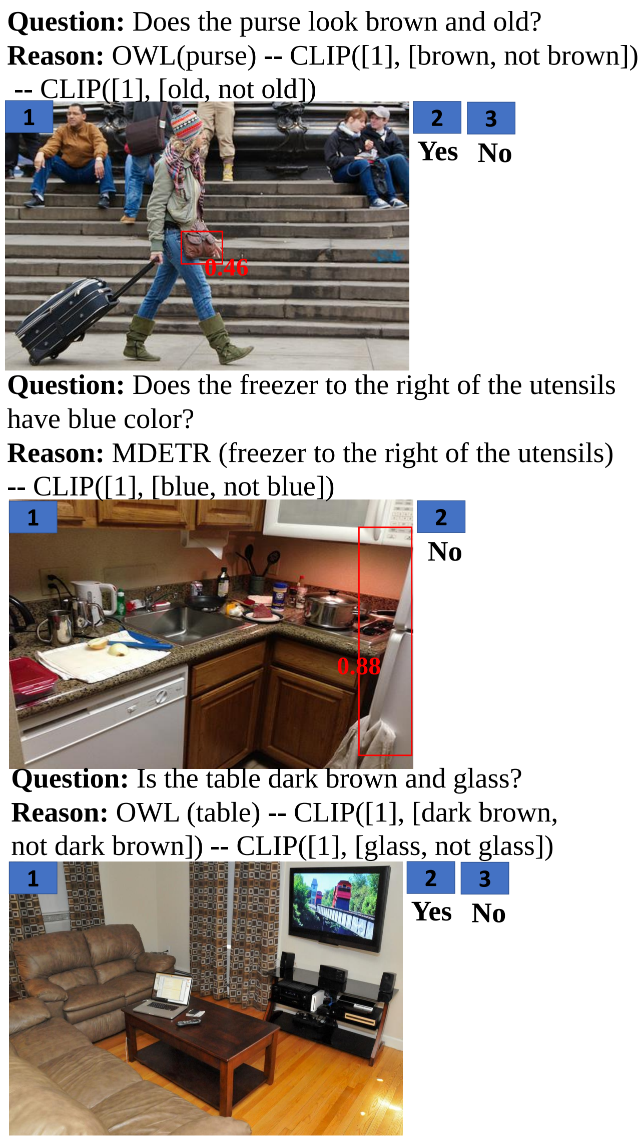


Figure D.1: Visualization of existence-related questions where mentioned objects in the questions do not exist.

D.4 Detailed Implementation for Spatial Heuristics

In this section, we give the mathematical definitions of the spatial heuristics. The input bounding box is denoted as $\mathbf{b} = (x, y, w, h)$, representing the relative position and relative size of the object in the VQA image.

`Spatial Determine (SpD)` receives an object bounding box and determines which position in the original image the object is at. The position candidates $\mathcal{P}$ are generated according to the question. When the question is asking for the horizontal position of the object, $\mathcal{P} = \{\text{left, right}\}$; When the question is asking for the vertical


Question: Does the purse look brown and old?
Reason: OWL(purse) -- CLIP([1], [brown, not brown]) -- CLIP([1], [old, not old])

Question: Does the freezer to the right of the utensils have blue color?
Reason: MDETR (freezer to the right of the utensils) -- CLIP([1], [blue, not blue])

Question: Is the table dark brown and glass?
Reason: OWL (table) -- CLIP([1], [dark brown, not dark brown]) -- CLIP([1], [glass, not glass])

Figure D.2: Visualization of questions asking about existence of attributes.

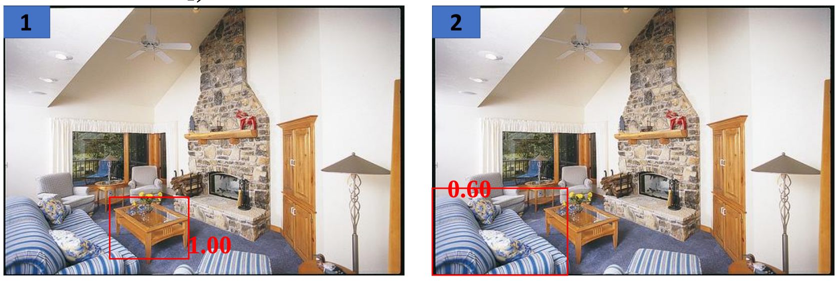
position of the object, $\mathcal{P} = \{\text{top, bottom}\}$. The SpD module is implemented as:

$$\texttt{SpD}(\mathbf{b}, \mathcal{P}) = \begin{cases} \text{left}, & \text{if} \quad x < 0.5 \\ \text{right}, & \text{else} \end{cases} \tag{D.1}$$

when $\mathcal{P} = \{\text{left, right}\}$. When $\mathcal{P} = \{\text{top, bottom}\}$, the spatial heuristic is derived as:

$$\texttt{SpD}(\mathbf{b}, \mathcal{P}) = \begin{cases} \text{top}, & \text{if} \quad y < 0.5 \\ \text{bottom}, & \text{else} \end{cases} \tag{D.2}$$

Question: Is the wood table to the left of a couch?
Reason: MDETR(wood table) – OWL(couch) -- SpC([1], [2], [to the left of, not to the left of])



Question: Do you see knives in the full drawer?
Reason: OWL(knfie) – MDETR(full drawer) -- CLIP([1], [2], [in, not in])

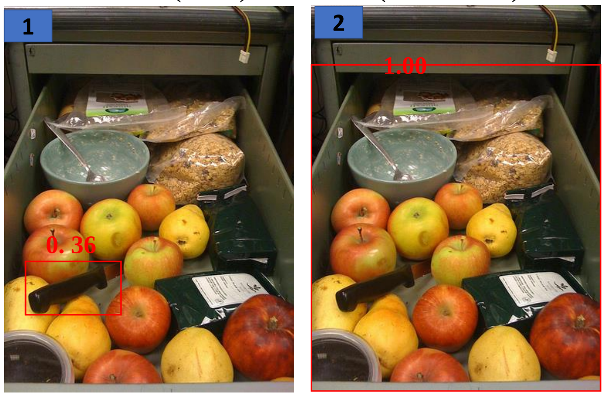


Figure D.3: Visualization of questions asking about the existence of relations.

The SpD heuristic will be used in the `Query` module when asking about either horizontal or vertical position.

Spatial Chooser (SpC) receives two bounding boxes of objects $\mathbf{b}_1, \mathbf{b}_2$ and aims to choose their spatial relations from the relation candidates in $\mathcal{C}$ ($\mathbf{b}_1$ is *RELA* $\mathbf{b}_2$). For instance, when $\mathcal{C} = \{\text{to the left of, to the right of}\}$:

$$\texttt{SpC}(\mathbf{b}_1, \mathbf{b}_2, \mathcal{C}) = \begin{cases} \text{left, if} & x_1 < x_2 \\ \text{right,} & \text{else} \end{cases} \tag{D.3}$$

When $\mathcal{C} = \{\text{above, beneath}\}$:

$$\texttt{SpC}(\mathbf{b}_1, \mathbf{b}_2, \mathcal{C}) = \begin{cases} \text{above, if} & y_1 < y_2 \\ \text{beneath,} & \text{else} \end{cases} \tag{D.4}$$

The SpC rule will be applied to the *Choose* type of questions if the choices of

Dataset	Train		Val	
	# Ques.	**# Img.**	**# Ques.**	**# Img.**
GQA	943,000	72,140	132,062	10,234
VQA	443,757	82,783	214,354	40,504

Table D.2: Statistical distributions of the GQA and the VQA dataset.

relations fall into the sets below: [{to the left of},{to the right of}] and [{above, on top of},{under, below, beneath, underneath}]

D.5 Dataset Statistics

In Table D.2, we provide statistics of the GQA and the VQA dataset. Following [161, 174], we use the validation split for testing. Specifically, we report *soft vqa scores* as there may be multiple possible answers to a question similar to previous works. [161, 174, 4, 45].

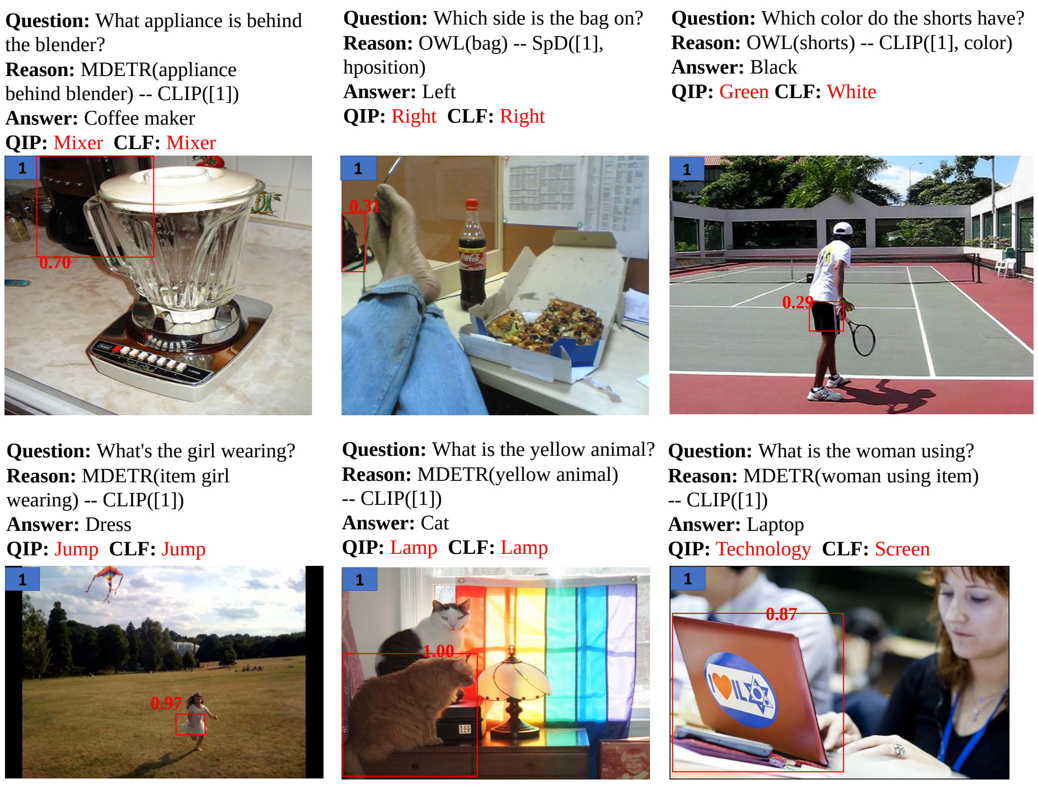
D.6 Layout Generation

The layout generation can be accomplished either with syntatic parser or a pre-trained sequence-to-sequence layout generator. On the VQA dataset, we follow [5, 69] to parse questions with Stanza[1] and transform the parsed tree into reasoning graphs where each node is a pre-defined module with rules most similar to [69]. The graphs are converted to module sequences with the post-order traversal. The linearlized module sequence is used as the layout. On GQA dataset, we leverage layouts generated by the pre-trained sequence-to-sequence layout generator from [25]. The generator adopts a coarse-to-fine two-stage generation paradigm as in [39] to encode questions and decode the sequence of module names and module inputs in two stages.

[1]https://github.com/stanfordnlp/stanza

Backbone	Yes/No		Other			
	Verify	Logical	Choose	Compare	Query	Overall
ViT-B/16	69.63	68.63	75.87	48.59	26.36	47.28
Res50× 16	68.51	68.71	75.78	41.84	25.69	46.49
ALBEF	68.08	69.99	75.93	48.40	29.38	48.68

Table D.3: Performance of the proposed model with different models for multimodal matching regarding different question types.


Question: What appliance is behind the blender?
Reason: MDETR(appliance behind blender) -- CLIP([1])
Answer: Coffee maker
QIP: Mixer **CLF:** Mixer

Question: Which side is the bag on?
Reason: OWL(bag) -- SpD([1], hposition)
Answer: Left
QIP: Right **CLF:** Right

Question: Which color do the shorts have?
Reason: OWL(shorts) -- CLIP([1], color)
Answer: Black
QIP: Green **CLF:** White

Question: What's the girl wearing?
Reason: MDETR(item girl wearing) -- CLIP([1])
Answer: Dress
QIP: Jump **CLF:** Jump

Question: What is the yellow animal?
Reason: MDETR(yellow animal) -- CLIP([1])
Answer: Cat
QIP: Lamp **CLF:** Lamp

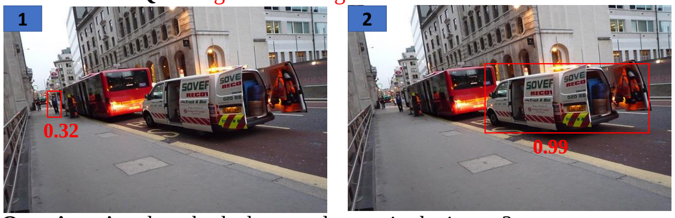
Question: What is the woman using?
Reason: MDETR(woman using item) -- CLIP([1])
Answer: Laptop
QIP: Technology **CLF:** Screen

Figure D.4: Visualization of VQA examples with short reasoning chains.

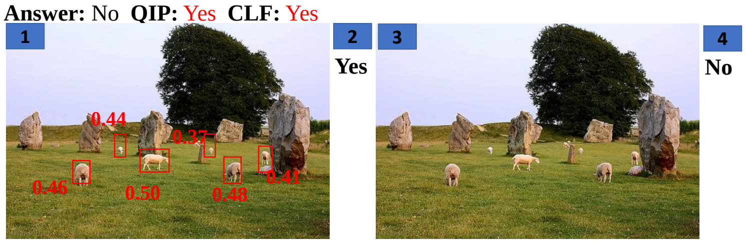
D.7 Answer Filtering

Basically, we follow [161] to narrow down the set of possible answer candidates with the language model T5 [144]. For the VQA dataset, we directly leverage the published generated candidate answers for each question from the paper [161]. For the GQA dataset, the *Verify* and *Logical* type questions have binary answers *yes/no*. For the *Compare* and *Choose*, candidate answers are available in the generated layouts. For the *Query* type of questions, we first convert questions into masked templates with a rule-based converter [35]. T5 is applied to retrieve the masked word, which filters out irrelevant answers in the answer vocabulary according to contexts.

Question: Is the person to the right or to the left of the vehicle next to the sidewalk?
Reason: OWL(person) -- MDETR(vehicle next to sidewalk) -- SpC([1], [2], [left, right])
Answer: Left **QIP:** Right **CLF:** Right



Question: Are there both sheep and geese in the image?
Reason: OWL(sheep) – Exist([1]) -- OWL(goose) – Exist([1]) – And ([2], [4])
Answer: No **QIP:** Yes **CLF:** Yes



Question: Does the door of the elevator appear to be open and metallic?
Reason: MDETR(door of elevator) – CLIP([1], [open, not open]) – CLIP ([1], [metal, not metal]
Answer: No **QIP:** Yes **CLF:** Yes

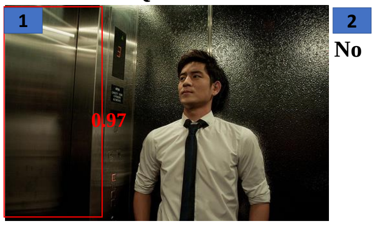


Figure D.5: Visualization of VQA examples with long reasoning chains.

D.8 Detailed Results

We provide the detailed results for replacing CLIP with ALBEF (discussed in Section 7.3) in Table D.3 considering different types of questions.

D.9 Visualization of Zero-shot NMNs

In this section, we provide more visualization examples which the zero-shot NMNs answers correctly while the baselines (QIP and TAC-P) fail. In Figure D.4, we show examples with short reasoning chain, specifically, only two-step in Mod-Zero-VQA. According to the results, we observe that each intermediate step gives interpretable outputs. By question decomposition and leveraging PTMs, our model can focus on relevant regions of the image (e.g., the first and third example in the first row of Figure D.4) so that eliminating noise from backgrounds. Without filtering irrelevant

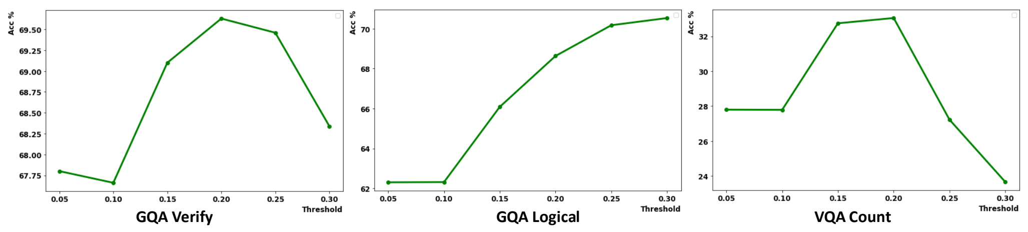


Figure D.6: Performances of the zero-shot VQA model regarding to different thresholds of confident scores in the OWL model.

information in the image, baselines pay attention to dominant objects in the image, leading to wrong predictions (e.g., the third example in the first row which QIP and TAC-P seems to focus on the ground and the T-shirt when answering the question). In Figure D.5, we visualize questions with relatively longer reasoning chains. These compositional questions usually call several reasoning capabilities, making it hard for pre-trained VL models to deal with [170]. With question decomposition, each pre-trained model takes a sub reasoning task, easing the burden from answering a complicated question.

According to the visualization, we also find a frequent error case resulting from the wrongly-generated NMN layout. The coarse-to-fine two stage generation suffers from the issue of early stopping that the generated arguments is incomplete. For instance, the ground-truth step should be `Find`(*coffee table*) while the generated result is `Find`(*coffee*).

D.10 Out-of-Distribution Setting Construction

We consider an Out-of-Domain Generalization (OOD) setting, where test images are related to scenes (i.e., *Indoor*, *Food* and *Street*) not observed during training. For the *Indoor* scene, we directly leverage the annotation from Visual Genome [88], where images are classified as indoors and outdoors. For the other two settings, we filter out training images containing those scene-specific objects and make sure a certain protion of objects in the testing images are about those objects (in other words, testing images are related to the scene). Below, we provide the lists of scene specific objects in the *Food* and *Street* scene.

Food: plate, banana, table, food, pizza, donut, fork, bowl, cheese, napkin, glass, cake, tomato, bread, apple, carrot, knife, broccoli, vegetable, fruit, cup, sauce, orange, spoon, meat, pepper, crust, onion, sandwich, home plate, topping, catcher, tray, lettuce, container, dish, bottle, batter, umpire, frosting, hot dog, egg, chicken, bat, box, mask, paper, mushroom, mug, pitcher, dispenser, liquid, label, bacon, tablecloth, nut, leaf, utensil, salad, hand, crumb, lemon, basket, mound, card, helmet, strawberry, lid, pan, seed, chair, menu, jar, player, sausage, icing, juice, shirt, spinach, sprinkle, dugout, counter, bag, flower, berry, goat, sailboat, uniform, steering wheel, glove, heel, pastry, bubble, finger, sugar, beer, oven, heart, dessert, herb

Street: car, sign, building, pole, letter, tree, tire, road, wheel, sidewalk, bus, train, street, number, door, sky, bike, windshield, truck, street light, motorcycle, leaf, traffic light, roof, ground, post, license plate, arrow, vehicle, fence, cloud, word, grass, wire, van, bicycle, gravel, bush, platform, fire hydrant, house, seat, flag, bag, pavement, step, graffiti, sticker, logo, paint, luggage, cone, chain, pipe, helmet, bridge, balcony, parking lot, jacket, plant, stop sign, train car, umbrella, taxi, lamp, box, crosswalk, flower, bench, brick, store, trash can, clock, gate, station, jean, grill, suv, driver, hook, pant, trash, tower, city, stair, rock, coat, rose, chimney, trailer, american flag, entrance

D.11 Experiment Settings

In this section, we discuss the experiment settings regarding to the size of models, method of choosing hyper-parameters and the used software packages and versions.

Model Size: We provide the number of parameters of different models in Table D.4. Our model includes the OWL model, the MDETR model, the CLIP$_{\text{ViT-B/16}}$ and the T5 model for answer filtering. It consists of $1,521$M parameters, of which the T5 model takes 770M parameters, the OWL model takes 583M parameters, the MDETR model takes 170M parameters and the CLIP model takes 151M parameters. After pre-processing object detection and answer filtering, it takes 6G GPU memory for inference.

Method	**# Params (M)**
VL-T5$_{\text{no-vqa}}$	288
FEWVLM$_{\text{base}}$	288
FEWVLM$_{\text{large}}$	804
VLKD$_{\text{ViT-L/14}}$	713
BNP-VQA$_{\text{6M}}$	669
BNP-VQA$_{\text{11B}}$	1,576
Frozen	1,040
QIP$_{\text{ViT-B/16}}$	151
TAC-P$_{\text{ViT-B/16}}$	921
Zero-shot NMNs	1,521

Table D.4: Number of parameters in VQA models.

Hyper-parameters: As we focus on the zero-shot learning setting so that there is no training process. Here we provide hyper-parameters used as thresholds. For the OWL model [126], we set the threshold of confident score as 0.2, which is set empirically, to filter out detected bounding boxes of which the confident scores are too long. We show test the robustness of the proposed zero-shot VQA model regarding to the hyper-parameter of the threshold and provide experimental results corresponding to the threshold varying from $0.05, , 0.1, 0.15, 0.2, 0.15, 0.3$ in Figure D.6. As proven in Section 4.2, the detection result mostly affects binary questions which rely more on object detection results, we here provide results for *Verify* and *Logical* type of questions on GQA. Besides, *Count* type questions also heavily rely on the quality of object detection. According to the results, we observe the zero-shot NMNs achieves relatively stable performances regarding to different thresholds for confident scores on *Verify* type questions, while less stable for the *Logical* and *Count* type questions. The stability on *Verify* questions depicts the robustness of the detection model. As *Logical* questions combines results from two *Verify* questions, the error may propagate if one predicted answer of the *Verify* question is wrong. An interesting finding is that the performance does not drop as the threshold increases. This may be that answers are biased to *no*. With the increment of thresholds, the model is more likely to answer *no*. *Count* questions are more sensitive to the threshold because lower thresholds lead to the case that uncertain regions to be detected while higher thresholds are more harmful that correctly detected objects will be filtered out. In conclusion, the threshold is important to the quality of detection and setting it from 0.2 to 0.25 gives good performances. For the MDETR model, we directly follow their published code for detection and set the threshold as 0.7[2].

Package Version: We list the software packages used as well as the corresponding versions in Table D.5.

Package	**Version)**
PyTorch	1.9.0
Transformers	4.19.2
Stanza	1.4.0
NLTK	3.2.5

Table D.5: Versions of packages used in our experiments.

[2]https://colab.research.google.com/drive/11xz5IhwqAqHj9-XAIP17yVIuJsLqeYYJ?usp=sharing

Appendix E

Appendix for KGenVQA

E.1 Details of K-Means Clustering

To divide testing instances into different clusters, we first convert each context-question-knowledge triplet into vector representations. Specifically, the context, question and the initial piece of knowledge will be concatenated and the textBERT [36] to encode the concatenated sentence. Based on the encoded textual representation, we used the *K-Means* clustering to divide all instances into K clusters. Given an instance waiting for knowledge generation, which belongs to the cluster k, instances from other clusters will serve as demonstrations. In other words, we randomly select one demonstration from each cluster except the k-th cluster so that there are $K - 1$ demonstrations for the testing example. The set of demonstrations we denote as `PSEUDO DEMO`. Then we prompt LLMs again with the self-supervised demonstrations with an input. We will iteratively conduct the process mentioned above T times where at the t-th time step we obtain a piece of knowledge $||_t$ and finally we have T knowledge pieces.

Model and Size		# shots	Setting	OK-VQA
OPT	13B	32	*w/o* KGen	36.1
		32	*w* KGen	39.6
	30B	16	*w/o* KGen	36.7
		16	*w* KGen	43.8

Table E.1: Performance comparison between using and not using generated knowledge in the few-shot setting on OK-VQA dataset. KGen refers to knowledge generation.

Model	**Model Size**
Zero-shot Models ***without*** *Extra Training*	
PICa$_{\text{zero}}$	175B
PNP-VQA	1.2B, 3.4B, 11.8B
Img2LLM	6.7B, 13B, 30B, 66B, 175B
Zero-shot Models with Extra Training	
VL-T5$_{\text{no-vqa}}$	269M
Frozen	7.1B
VLKD$_{\text{ViT-L/14}}$	832M
FewVLM	785M
BLIP-2(OPT$_{6.7\text{B}}$)	7.8B
BLIP-2(FlanT5$_{\text{XL}}$)	4.1B
BLIP-2(FlanT5$_{\text{XXL}}$)	12.1B
Flamingo	3B, 9B, 80B
Few-shot Models	
ClipCap→Cap.→GPT	175B
ClipCap→Ratl.→GPT	175B
PICa$_{\text{few}}$	175B
PromptCap	175B

Table E.2: Summarizing of models for K-VQA.

E.2 Experiment Settings

Experiment Details For knowledge generation, we use GPT-3.5 (*text-davinci-003*[1]) as our LLM, with a suggested temperature of 0.7. For the K-means clustering in knowledge diversification stage, we set the number of cluster to be 8 empirically.

For answer prediction, because exact match is adopted for evaluation, we encourage the pre-trained QA model to give short answers. For UnifiedQA, we set the length penalty to be -1; for GPT-3.5, we add the following instruction: *Generate answers with as fewer words as possible.* After answer prediction, we conduct an

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

Model	**Acc.**
*Zero-shot Models **without** Extra Training*	
PICa$_{\text{zero},175B}$	17.7
PNP-VQA$_{\text{0.7B}}$	27.1
PNP-VQA$_{\text{3B}}$	34.1
PNP-VQA$_{\text{11B}}$	35.9
Img2LLM$_{\text{6.7B}}$	38.2
Img2LLM$_{\text{13B}}$	39.9
Img2LLM$_{\text{30B}}$	41.8
Img2LLM$_{\text{66B}}$	43.2
Img2LLM$_{\text{175B}}$	45.6
Zero-shot Models with Extra Training	
VL-T5$_{\text{no-vqa}}$	5.8
Frozen	5.9
VLKD$_{\text{ViT-L/14}}$	13.3
FewVLM	16.5
BLIP-2(OPT)$_{\text{6.7B}}$	36.4
BLIP-2(FlanT5$_{\text{XL}}$)$_{\text{3B}}$	40.7
BLIP-2(FlanT5$_{\text{XXL}}$)$_{\text{3B}}$	45.9
Flamingo$_{\text{3B}}$	41.2
Flamingo$_{\text{9B}}$	44.7
Flamingo$_{\text{80B}}$	50.6
Few-shot Models	
PICa$_{\text{few},175B}$ (n=1)	40.8
PromptCap$_{175B}$ (n=1)	48.7

Table E.3: Model performancee on OK-VQA dataset. For models with different model sizes, we show the model size with subscripts.

answer post-processing step as proposed in [9].

We implement our model on NVIDIA Tesla V100 GPUs with 32 GB of dedicated memory. The system ran on CUDA version 11.1. For UnifiedQA, except 11B version, we implemented with a single GPU. For UnifiedQA 11B model and OPT model series, we implement with model parallel on four GPUs.

Package Version In this experiment, we rely on the PyTorch library, 1.13.1 version. For the implementation of BLIP [99] (used for image caption generation), we leverage the LAVIS package from Salesforce [2] (version 1.0.2), for OPT [204] and UnifiedQA model [83] we use the transformers package from Huggingface [3] (version 4.29.2), and for GPT-3.5 model, we leverage the OpenAI API [4].

[2]https://github.com/salesforce/LAVIS/tree/main/lavis
[3]https://huggingface.co/
[4]https://platform.openai.com/overview

Model	**Acc.**
*Zero-shot Models **without** Extra Training*	
Img2LLM$_{6.7B}$	33.3
Img2LLM$_{13B}$	33.3
Img2LLM$_{30B}$	36.9
Img2LLM$_{66B}$	38.7
Img2LLM$_{175B}$	42.9
Few-shot Models	
ClipCap$\rightarrow$Cap$\rightarrow$GPT$_{175B}$ (n=10)	16.6
ClipCap$\rightarrow$Rel$\rightarrow$GPT$_{175B}$	18.1
PromptCap$_{175B}$ (n=32)	56.3

Table E.4: Model performancee on A-OKVQA dataset. For models with different model sizes, we show the model size with subscripts.

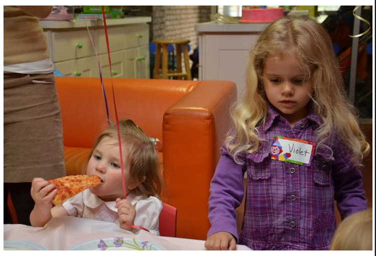
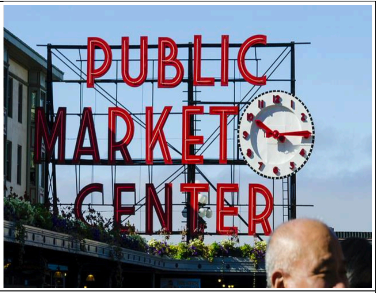
Img.		
Ques.	Which type of leather is used for making the sofa set shown in this picture?	Where in the world is this located?
GT.	cow, fake, fine grain, suede	seattle, san francisco, seattle usa, boston massachusetts
Pred.	black leather	czech republic
Cap.	two child a pizza pizza three people child up pizza. a young girl and a young girl with pizza as food. a young girl eating pizza while sitting in a booth	a sign outside of a market market sign on a clear day. the sign shows market square, with a lot of people, and a large clock. a group of people outside of a building showing a clock.
Kn.	The sofa set shown in this picture is likely made of faux leather, which is a synthetic material made to look and feel like real leather.	This market square is located in the city of Prague, Czech Republic.

Table E.5: Visualization of error cases. GT. is for ground-truth annotation, Pred. is for predictions from models, Cap. is for the image captions and Kn. is for generated knowledge.

Model Size: We show model size in Table E.2. If we one model has different versions of model size, we separate them with comma.

Image			
Question	What would happen if these items fall to the ground?	What sates are these grown in?	Name one famous person whom also has a black and white one of these?
Ground Truth	shatter, they would shatter, break, they would break	florida california, california, florida	taylor swift, russell brand, hillary clinton, ernest hemingway
Base Prediction	nothing	texas	kate winslet
Generated Knowledge	If a glass item falls to the floor, it will break.	California and Florida are the leading producers of oranges.	Taylor Swift is a famous singer and songwriter who has a black and white cat named Meredith.
Our Prediction	they would break	california	taylor swift
Image			
Question	If it gets cold enough what will happen to the area being stepped over?	What knocked the guy off his chair?	What is the white cloud behind the jet called?
Ground Truth	freeze, frozen, it will freeze over, iced	wave, water	contrail, cloud, supersonic wave
Base Prediction	snow	water splash	halo
Generated Knowledge	If it gets cold enough, the area being stepped over will freeze, creating a layer of ice on top of the snow.	The waves in the water knocked the man off his chair.	The condensation trail, or contrail, is a visible trail of condensed water vapor created by an aircraft engine or wingtip vortices under certain atmospheric conditions.
Our Prediction	frozen	wave	contrail

Table E.6: Visualization of error cases of the baseline without generated knowledge, while our method answers correctly with the help with generated knowledge. Wrong predictions are highlighted in red.

E.3 Few-shot Setting Results

We provide the results for our method in the few-shot setting on OK-VQA in the section. Specifically, we leverage the OPT model [204] as the final QA model and give a few demonstrations. Each demonstration consists of a question, an image description as the context, an answer and optional related knowledge (in the *w* KGen setting). The results are shown in Table E.1. According to the results, we observe consistent improvements after adding generated knowledge, indicating our method can generalize to the few-shot setting as well.

E.4 Fair Comparison with PICa

Considering PICa$_{\text{zero,175B}}$ leverages only a single piece of image description while our method uses multiple captions, following [171], improvements may potentially come from more detailed image descriptions. To ablate the impact from image description side, we use a single caption as the image description, similar to PICa$_{\text{zero,175B}}$. It achieves $\mathbf{33.8}$ on OK-VQA, with about $\mathbf{16}$ absolute accuracy improvements over PICa$_{\text{zero,175B}}$. Further more, we used only the generated knowledge as inputs to text-based QA models (UnifiedQA$_{\text{3B}}$). It achieves $\mathbf{33.5}$ on OK-VQA, highlighting that generated knowledge itself contains information for question answering.

E.5 Model Performance

We only provide models in a fair comparison in Section 8.3.3. In this part, we provide performance of models on K-VQA including zero-shot K-VQA models without extra training but have larger model sizes, zero-shot K-VQA models with extra training and few-shot K-VQA models. The results on OKVQA and A-OKVQA are shown in Table E.3 and Table E.4 respectively.

E.6 Error Cases

In this section, we provide visualization of two error cases of which the generated knowledge is inadequate. The reason of generating the harmful knowledge is because of inaccurate image captions. A potential way of improving our method is to improve the quality of image descriptions.

E.7 Comparison with the Baseline without Knowledge

In this section, we provide visualization of error cases of the baseline model without knowledge and compare with our method. The visualized examples are shown in

Num.	Content
1	Context:The company in the image is Monsanto. There are two men selling products. The logo behind two men is Monsanto. Question:What does company in the image own? Knowledge:Monsanto is a multinational agrochemical and agricultural biotechnology corporation. It is one of the world's leading producers of roundup, a glyphosate herbicide.
2	Context:The red vegetable is tomato. There is a sandwich with tomato and lettuce. There is a sandwich on the table. Question:Where can this red vegetable be found? Knowledge:tomatoes are usually planted in gardens.
3	Context:The man is playing tennis. The man is holding a tennis racket. A man is in a competition of tennis. Question:What English city is famous for a tournament for the sport this man is playing? Knowledge:The Wimbledon Championships is the oldest tennis tournament in the world.
4	Context:a plate with ham, tomatoes, meat, and sliced peppers on top of it. breakfast and bacon eggs scrambled toast. a breakfast sandwich, tomatoes, bacon, and eggs Question:what food in the photo has a lot of c vitamin? Knowledge:Tomatoes and tomato products are rich sources of folate, vitamin C, and potassium. Eggs contain decent amounts of vitamin D, vitamin E, vitamin B6, calcium and zinc. Bacon provides a good amount of B vitamins.
5	Context:a man sitting in front of a laptop computer smiling and posing for the camera. a man wearing glasses sitting in front of a laptop. a man in glasses and glasses at a desk with laptop. Question:what purpose do the glasses the man is wearing serve? Knowledge:Glasses are typically used for vision correction, such as with reading glasses and glasses used for nearsightedness.
6	Context:a bedroom with a bed, wall paper and lamp. a bed with storage underneath it in a room. a bed in a small room with pillows and box drawers. Question:what was the largest size of that platform that we have? Knowledge:Single size is 91 cm x 190 cm. Super single size is 107 cm x 190 cm. Queen size is 152 cm x 190 cm. King size is 182 cm x 190 cm.

Table E.7: Contents of manual prompts.

Table E.6. Noted, we do not perform cherry-picking. The visualized cases are the first six error cases of the baseline model on OK-VQA while being correctly addressed by our method. To keep the table tidy, we only present one piece of generated table. According to the visualization, we observe our generated knowledge largely benefit addressing these questions in need of external knowledge.

E.8 Manual Prompts

Here we provide a full list of six manual prompts in Table E.7. Before the demonstrations, we also add an instruction: *Please generate related background knowledge to the question:* in the front. Knowledge are collected from searching with Google.

E.9 Details for Human Evaluation

In this part, we provide more details about human evaluation about the knowledge quality. We invite two annotators for evaluation of 40 questions with five pieces of generated knowledge. Firstly, they will be given an instruction, indecating the definition of the K-VQA task, an example of the K-VQA task and the goal of the evaluation. Next, we describe what information (i.e., question, ground-truth answer, generated knowledge, and image) will be provided to them and the denotations of the information. Thirdly, we elaborate the definitions of four metrics. For the metrics of *Relevance*, *Factuality* and *Helpfulness*, besides definitions, we provide a few concrete examples in texts to make it easier for understanding. The definifions and examples are provided in Table E.8. For the full information of the annotated knowledge, please refer to the Supplementary file.

Attributes	**Definition**	**Example**
Grammaticality	Whether the knowledge statement is grammatical (e.g., whether a complete and fluent sentence; whether human can understand the sentence).	None
Relevance	Whether a knowledge statement is relevant to the given question. A statement is relevant if it covers the same topic as the question or contains a salient concept that is the same as or similar to the one in the question (provided indirect but related information).	[Image]: a bedroom with a bed [Question]: what was the largest size of that platform that we have? [Knowledge]: Single size is 91 cm x 190 cm. Super single size is 107 cm x 190 cm. Queen size is 152 cm x 190 cm. King size is 182 cm x 190 cm. [Judge]:Relevant. Because the information is related to the topic on bed size.
Factuality	Whether a knowledge statement is (mostly) factually correct or not. If there are exceptions or corner cases, it can still be considered factual if they are rare or unlikely.	[Image]: a triangle in the image [Question]: what shape is the object in the image? [Knowledge]: A rectangle is a shape with two equal sides [Judge]: Not factual, because a rectangle has four sides [Image]: a limousine; a car [Question]: how many doors does the vehicle in the image have? [Knowledge]: A limousine has four doors. [Judge]: Factual. [Image]: a human being [Question]: how many fingers does this creature have? [Knowledge]: A human hand has four fingers and a thumb. [Judge]: Factual, despite that there are exceptions – people with disabilities may have less or more fingers.
Helpfulness	Whether a knowledge statement is (mostly) factually correct or not. If there are exceptions or corner cases, it can still be considered factual if they are rare or unlikely.	[Image]: a subway in the image [Question]: How often you take this transportation back and forth to work per week? [Knowledge]: You take the subway back and forth to work five days a week [Judge]: Helpful. Because the statement directly supports the answer. [Image]: a spider [Question]: how many legs does the animal in the image have? [Knowledge]: Arachnids have eight legs [Judge]: Helpful. Although the statement does not directly refer to spiders, together with the fact that ”spiders are a kind of arachnids” it completes a reasoning chain in deriving the answer. [Image]: two persons are playing chess [Question]: what are the results of the game? [Knowledge]: A game of chess has two outcomes [Judge]: Harmful. Since the statement supports answering ”two outcomes” instead of ”three outcomes”. [Image]: a person in the white background. [Question]: How many chromosomes does the creature have? [Knowledge]: human beings are mammals. [Judge]: Neutral. The knowledge does not provide information in favor or contrast of answering the question.

Table E.8: Definitions and examples for evaluation metrics.

Bibliography

[1] A. Agrawal, D. Batra, D. Parikh, and A. Kembhavi. Don't just assume; look and answer: Overcoming priors for visual question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4971–4980, 2018.

[2] J. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J. L. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Binkowski, R. Barreira, O. Vinyals, A. Zisserman, and K. Simonyan. Flamingo: a visual language model for few-shot learning. In *NeurIPS*, 2022.

[3] B. AlKhamissi, F. Ladhak, S. Iyer, V. Stoyanov, Z. Kozareva, X. Li, P. Fung, L. Mathias, A. Celikyilmaz, and M. T. Diab. Token: Task decomposition and knowledge infusion for few-shot hate speech detection. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 2109–2120, 2022.

[4] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 6077–6086, 2018.

[5] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural module networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 39–48, 2016.

[6] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: visual question answering. In *2015 IEEE International Conference on Computer Vision, ICCV*, pages 2425–2433, 2015.

[7] A. Arango, J. Pérez, and B. Poblete. Hate speech detection is not as easy as you may think: A closer look at model validation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR*, pages 45–54, 2019.

[8] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. G. Ives. Dbpedia: A nucleus for a web of open data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007*, volume 4825, pages 722–735, 2007.

[9] A. Awadalla, I. Gao, J. Gardner, J. Hessel, Y. Hanafy, W. Zhu, K. Marathe, Y. Bitton, S. Y. Gadre, S. Sagawa, J. Jitsev, S. Kornblith, P. W. Koh, G. Ilharco, M. Wortsman, and L. Schmidt. Openflamingo: An open-source framework for training large autoregressive vision-language models. *CoRR*, abs/2308.01390, 2023.

[10] M. R. Awal, R. Cao, R. K. Lee, and S. Mitrovic. Angrybert: Joint learning target and emotion for hate speech detection. In *Advances in Knowledge Discovery and Data Mining - 25th Pacific-Asia Conference, PAKDD*, volume 12712 of *Lecture Notes in Computer Science*, pages 701–713, 2021.

[11] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760, 2017.

[12] T. Baltrusaitis, C. Ahuja, and L. Morency. Multimodal machine learning: A survey and taxonomy. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(2):423–443, 2019.

[13] H. Bao, W. Wang, L. Dong, Q. Liu, O. K. Mohammed, K. Aggarwal, S. Som, S. Piao, and F. Wei. Vlmo: Unified vision-language pre-training with mixture-of-modality-experts. In *NeurIPS*, 2022.

[14] H. Ben-Younes, R. Cadène, M. Cord, and N. Thome. MUTAN: multimodal tucker fusion for visual question answering. In *IEEE International Conference on Computer Vision, ICCV*, pages 2631–2639, 2017.

[15] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS*, 2020.

[16] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner. Understanding disentangling in β-vae. *arXiv preprint arXiv:1804.03599*, 2018.

[17] R. Cao and J. Jiang. Modularized zero-shot VQA with pre-trained models. *CoRR*, abs/2305.17369, 2023.

[18] R. Cao and R. K. Lee. Hategan: Adversarial generative-based data augmentation for hate speech detection. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING*, pages 6327–6338, 2020.

[19] R. Cao, R. K. Lee, and T. Hoang. Deephate: Hate speech detection via multi-faceted text representations. In *WebSci '20: 12th ACM Conference on Web Science*, pages 11–20, 2020.

[20] R. Cao, R. K. Lee, W. Chong, and J. Jiang. Prompting for multimodal hateful meme classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 321–332, 2022.

[21] R. Cao, M. S. Hee, A. Kuek, W. Chong, R. K. Lee, and J. Jiang. Pro-cap: Leveraging a frozen vision-language model for hateful meme detection. In *Proceedings of the 31st ACM International Conference on Multimedia, MM*, pages 5244–5252, 2023.

[22] W. Chao, H. Hu, and F. Sha. Cross-dataset adaptation for visual question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 5716–5725, 2018.

[23] D. Chatzakou, N. Kourtellis, J. Blackburn, E. D. Cristofaro, G. Stringhini, and A. Vakali. Mean birds: Detecting aggression and bullying on twitter. In *Proceedings of the 2017 ACM on Web Science Conference, WebSci*, pages 13–22, 2017.

[24] R. T. Chen, X. Li, R. Grosse, and D. Duvenaud. Isolating sources of disentanglement in vaes. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 2615–2625, 2018.

[25] W. Chen, Z. Gan, L. Li, Y. Cheng, W. Y. Wang, and J. Liu. Meta module network for compositional visual reasoning. In *IEEE Winter Conference on Applications of Computer Vision, WACV*, pages 655–664. IEEE, 2021.

[26] X. Chen, H. Fang, T. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick. Microsoft COCO captions: Data collection and evaluation server. *CoRR*, 2015.

[27] Y. Chen, Y. Zhou, S. Zhu, and H. Xu. Detecting offensive language in social media to protect adolescent online safety. In *2012 International Conference on Privacy, Security, Risk and Trust, PASSAT*, pages 71–80, 2012.

[28] K. Chiu and R. Alexander. Detecting hate speech with GPT-3. *CoRR*, abs/2103.12407, 2021.

[29] J. Cho, J. Lei, H. Tan, and M. Bansal. Unifying vision-and-language tasks via text generation. In *Proceedings of the 38th International Conference on Machine Learning, ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 1931–1942, 2021.

[30] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel. Palm: Scaling language modeling with pathways. *CoRR*, abs/2204.02311, 2022.

[31] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, S. Narang, G. Mishra, A. Yu, V. Y. Zhao, Y. Huang, A. M. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei. Scaling instruction-finetuned language models. *CoRR*, abs/2210.11416, 2022.

[32] W. Dai, L. Hou, L. Shang, X. Jiang, Q. Liu, and P. Fung. Enabling multimodal generation on CLIP via vision-language knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL*, pages 2383–2395, 2022.

[33] W. Dai, Z. Liu, Z. Ji, D. Su, and P. Fung. Plausible may not be faithful: Probing object hallucination in vision-language pre-training. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics EACL*, pages 2136–2148, 2023.

[34] T. Davidson, D. Warmsley, M. W. Macy, and I. Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM*, pages 512–515, 2017.

[35] D. Demszky, K. Guu, and P. Liang. Transforming question answering datasets into natural language inference datasets. *CoRR*, abs/1809.02922, 2018.

[36] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pages 4171–4186, 2019.

[37] Y. Ding, J. Yu, B. Liu, Y. Hu, M. Cui, and Q. Wu. Mukea: Multimodal knowledge extraction and accumulation for knowledge-based visual question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 5079–5088, 2022.

[38] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati. Hate speech detection with comment embeddings. In *Proceedings of the 24th International Conference on World Wide Web Companion, WWW*, pages 29–30, 2015.

[39] L. Dong and M. Lapata. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 731–742, 2018.

[40] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR*, 2021.

[41] A. Fan, M. Lewis, and Y. N. Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 889–898, 2018.

[42] E. Fersini, F. Gasparini, G. Rizzi, A. Saibene, B. Chulvi, P. Rosso, A. Lees, and J. Sorensen. Semeval-2022 task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation, SemEval@NAACL*, pages 533–549, 2022.

[43] P. Fortuna and S. Nunes. A survey on automatic detection of hate speech in text. *ACM Comput. Surv.*, 51(4):85:1–85:30, 2018.

[44] P. Fortuna and S. Nunes. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)*, 51(4):1–30, 2018.

[45] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 457–468, 2016.

[46] B. Gambäck and U. K. Sikdar. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online, ALW@ACL*, pages 85–90, 2017.

[47] Z. Gan, Y.-C. Chen, L. Li, C. Zhu, Y. Cheng, and J. Liu. Large-scale adversarial training for vision-and-language representation learning. *arXiv preprint arXiv:2006.06195*, 2020.

[48] H. Gao, J. Mao, J. Zhou, Z. Huang, L. Wang, and W. Xu. Are you talking to a machine? dataset and methods for multilingual image question answering. *CoRR*, abs/1505.05612, 2015.

[49] P. Gao, Z. Jiang, H. You, P. Lu, S. C. H. Hoi, X. Wang, and H. Li. Dynamic fusion with intra- and inter-modality attention flow for visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 6639–6648, 2019.

[50] T. Gao, A. Fisch, and D. Chen. Making pre-trained language models better few-shot learners. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing, ACL/IJCNLP*, pages 3816–3830, 2021.

[51] F. Gardères, M. Ziaeefard, B. Abeloos, and F. Lécué. Conceptbert: Concept-aware representation for visual question answering. In *Findings of the Association for Computational Linguistics: EMNLP*, volume EMNLP 2020 of *Findings of ACL*, pages 489–498, 2020.

[52] Y. Goldberg. A primer on neural network models for natural language processing. *J. Artif. Intell. Res.*, 57:345–420, 2016.

[53] H. Gonen, S. Iyer, T. Blevins, N. A. Smith, and L. Zettlemoyer. Demystifying prompts in language models via perplexity estimation. *CoRR*, abs/2212.04037, 2022.

[54] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 6325–6334, 2017.

[55] T. Gröndahl, L. Pajola, M. Juuti, M. Conti, and N. Asokan. All you need is: Evading hate speech detection. In *Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security, CCS*, pages 2–12, 2018.

[56] L. Gui, B. Wang, Q. Huang, A. Hauptmann, Y. Bisk, and J. Gao. KAT: A knowledge augmented transformer for vision-and-language. *CoRR*, 2021.

[57] D. Guo, A. M. Rush, and Y. Kim. Parameter-efficient transfer learning with diff pruning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP*, pages 4884–4896, 2021.

[58] J. Guo, J. Li, D. Li, A. M. H. Tiong, B. Li, D. Tao, and S. C. H. Hoi. From images to textual prompts: Zero-shot VQA with frozen large language models. *CoRR*, abs/2212.10846, 2022.

[59] N. Gupta, K. Lin, D. Roth, S. Singh, and M. Gardner. Neural module networks for reasoning over text. In *8th International Conference on Learning Representations, ICLR*, 2020.

[60] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of 1996 IEEE International Conference on Evolutionary Computation*, pages 312–317, 1996.

[61] S. M. Harabagiu, S. J. Maiorano, and M. Pasca. Open-domain textual question answering techniques. *Nat. Lang. Eng.*, 9(3):231–267, 2003.

[62] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 770–778, 2016.

[63] M. S. Hee, R. K. Lee, and W. Chong. On explaining multimodal hateful meme detection models. In *WWW '22: The ACM Web Conference*, pages 3651–3655, 2022.

[64] M. S. Hee, W. Chong, and R. K. Lee. Decoding the underlying meaning of multimodal hateful memes. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAIa*, pages 5995–6003, 2023.

[65] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J. W. Rae, O. Vinyals, and L. Sifre. Training compute-optimal large language models. *CoRR*, abs/2203.15556, 2022.

[66] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR*, 2020.

[67] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. de Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, volume 97, pages 2790–2799, 2019.

[68] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR*, 2022.

[69] R. Hu, J. Andreas, M. Rohrbach, T. Darrell, and K. Saenko. Learning to reason: End-to-end module networks for visual question answering. In *IEEE International Conference on Computer Vision, ICCV*, pages 804–813, 2017.

[70] Y. Hu, H. Hua, Z. Yang, W. Shi, N. A. Smith, and J. Luo. Promptcap: Prompt-guided task-aware image captioning. *CoRR*, abs/2211.09699, 2022.

[71] C. Huang, Q. Liu, B. Y. Lin, T. Pang, C. Du, and M. Lin. Lorahub: Efficient cross-task generalization via dynamic lora composition. *CoRR*, abs/2307.13269, 2023.

[72] L. Huang, W. Wang, J. Chen, and X. Wei. Attention on attention for image captioning. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV*, pages 4633–4642, 2019.

[73] D. A. Hudson and C. D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 6700–6709, 2019.

[74] E. Hwang and V. Shwartz. Memecap: A dataset for captioning and interpreting memes. *CoRR*, abs/2305.13703, 2023.

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

[76] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

[77] W. Jin, Y. Cheng, Y. Shen, W. Chen, and X. Ren. A good prompt is worth millions of parameters: Low-resource prompt-based learning for vision-language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics ACL*, pages 2763–2775, 2022.

[78] L. Jing, R. Li, Y. Chen, M. Jia, and X. Du. FAITHSCORE: evaluating hallucinations in large vision-language models. *CoRR*, abs/2311.01477, 2023.

[79] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. B. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1988–1997, 2017.

[80] A. Kamath, M. Singh, Y. LeCun, G. Synnaeve, I. Misra, and N. Carion. MDETR - modulated detection for end-to-end multi-modal understanding. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV*, pages 1760–1770, 2021.

[81] K. Kärkkäinen and J. Joo. Fairface: Face attribute dataset for balanced race, gender, and age. *arXiv preprint arXiv:1908.04913*, 2019.

[82] C. Kervadec, G. Antipov, M. Baccouche, and C. Wolf. Roses are red, violets are blue... but should VQA expect them to? In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2776–2785, 2021.

[83] D. Khashabi, S. Min, T. Khot, A. Sabharwal, O. Tafjord, P. Clark, and H. Hajishirzi. Unifiedqa: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP*, volume EMNLP 2020, pages 1896–1907, 2020.

[84] D. Kiela, S. Bhooshan, H. Firooz, and D. Testuggine. Supervised multimodal bitransformers for classifying images and text. In *Visually Grounded Interaction and Language (ViGIL), NeurIPS Workshop*, 2019.

[85] D. Kiela, H. Firooz, A. Mohan, V. Goswami, A. Singh, P. Ringshia, and D. Testuggine. The hateful memes challenge: Detecting hate speech in multimodal memes. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[86] J. Kim, J. Jun, and B. Zhang. Bilinear attention networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS*, pages 1571–1581, 2018.

[87] W. Kim, B. Son, and I. Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 5583–5594, 2021.

[88] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L. Li, D. A. Shamma, M. S. Bernstein, and L. Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vis.*, 123(1):32–73, 2017.

[89] G. K. Kumar and K. Nandakumar. Hate-clipper: Multimodal hateful meme classification based on cross-modal interaction of CLIP features. *CoRR*, abs/2210.05916, 2022.

[90] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

[91] H. Le, N. Chen, and S. Hoi. VGNMN: Video-grounded neural module networks for video-grounded dialogue systems. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3377–3393, July 2022.

[92] R. K. Lee, R. Cao, Z. Fan, J. Jiang, and W. Chong. Disentangling hate in online memes. In *MM '21: ACM Multimedia Conference*, pages 5138–5147, 2021.

[93] J. Lei, L. Yu, M. Bansal, and T. L. Berg. TVQA: localized, compositional video question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1369–1379, 2018.

[94] J. L. Leidner. Open-domain question answering from large text collection, m. pa[scedil]ca. *J. Log. Lang. Inf.*, 13(3):373–376, 2004.

[95] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 7871–7880, 2020.

[96] B. Li, Y. Zhang, L. Chen, J. Wang, J. Yang, and Z. Liu. Otter: A multi-modal model with in-context instruction tuning. *CoRR*, abs/2305.03726, 2023.

[97] G. Li, X. Wang, and W. Zhu. Boosting visual question answering with context-aware knowledge aggregation. In *MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020*, pages 1227–1235, 2020.

[98] J. Li, R. R. Selvaraju, A. Gotmare, S. R. Joty, C. Xiong, and S. C. Hoi. Align before fuse: Vision and language representation learning with momentum distillation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems*, pages 9694–9705, 2021.

[99] J. Li, D. Li, C. Xiong, and S. C. H. Hoi. BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning, ICML*, volume 162, pages 12888–12900, 2022.

[100] J. Li, D. Li, S. Savarese, and S. C. H. Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning, ICML*, volume 202, pages 19730–19742, 2023.

[101] L. H. Li, M. Yatskar, D. Yin, C. Hsieh, and K. Chang. Visualbert: A simple and performant baseline for vision and language. *CoRR*, abs/1908.03557, 2019.

[102] X. Li, X. Yin, C. Li, P. Zhang, X. Hu, L. Zhang, L. Wang, H. Hu, L. Dong, F. Wei, Y. Choi, and J. Gao. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Computer Vision - ECCV*, volume 12375, pages 121–137, 2020.

[103] X. L. Li and P. Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP*, pages 4582–4597, 2021.

[104] Y. Li, Y. Song, L. Cao, J. R. Tetreault, L. Goldberg, A. Jaimes, and J. Luo. TGIF: A new dataset and benchmark on animated GIF description. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2016.

[105] Y. Li, Y. Du, K. Zhou, J. Wang, W. X. Zhao, and J. Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 292–305, 2023.

[106] Y. Li, Z. Lin, S. Zhang, Q. Fu, B. Chen, J.-G. Lou, and W. Chen. Making large language models better reasoners with step-aware verifier. *CoRR*, abs/2206.02336, 2023.

[107] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. In *Computer Vision - ECCV 2014 - 13th European Conference*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755, 2014.

[108] P. Lippe, N. Holla, S. Chandra, S. Rajamanickam, G. Antoniou, E. Shutova, and H. Yannakoudakis. A multimodal framework for the detection of hateful memes. *arXiv preprint arXiv:2012.12871*, 2020.

[109] F. Liu, K. Lin, L. Li, J. Wang, Y. Yacoob, and L. Wang. Aligning large multi-modal model with robust instruction tuning. *CoRR*, abs/2306.14565, 2023.

[110] H. Liu and P. Singh. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.

[111] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *CoRR*, abs/2304.08485, 2023.

[112] J. Liu, A. Liu, X. Lu, S. Welleck, P. West, R. L. Bras, Y. Choi, and H. Hajishirzi. Generated knowledge prompting for commonsense reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL*, pages 3154–3169, 2022.

[113] X. Liu, Y. Zhu, J. Gu, Y. Lan, C. Yang, and Y. Qiao. Mm-safetybench: A benchmark for safety evaluation of multimodal large language models. *CoRR*, 2311.17600, 2024.

[114] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[115] I. Loshchilov and F. Hutter. Fixing weight decay regularization in adam. *CoRR*, 2017.

[116] H. Lovenia, W. Dai, S. Cahyawijaya, Z. Ji, and P. Fung. Negative object presence evaluation (NOPE) to measure object hallucination in vision-language models. *CoRR*, abs/2310.05338, 2023.

[117] J. Lu, D. Batra, D. Parikh, and S. Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019,*, pages 13–23, 2019.

[118] J. Ma, C. Zhou, H. Yang, P. Cui, X. Wang, and W. Zhu. Disentangled self-supervision in sequential recommenders. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 483–491, 2020.

[119] L. Ma, Z. Lu, and H. Li. Learning to answer questions from image using convolutional neural network. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 3567–3573, 2016.

[120] M. Malinowski and M. Fritz. Towards a visual turing challenge. *CoRR*, abs/1410.8027, 2014.

[121] M. Malinowski, M. Rohrbach, and M. Fritz. Ask your neurons: A neural-based approach to answering questions about images. In *2015 IEEE International Conference on Computer Vision, ICCV*, pages 1–9, 2015.

[122] K. Marino, M. Rastegari, A. Farhadi, and R. Mottaghi. OK-VQA: A visual question answering benchmark requiring external knowledge. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 3195–3204, 2019.

[123] K. Marino, X. Chen, D. Parikh, A. Gupta, and M. Rohrbach. KRISP: integrating implicit and symbolic knowledge for open-domain knowledge-based VQA. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 14111–14121, 2021.

[124] L. Mathias, S. Nie, A. Mostafazadeh Davani, D. Kiela, V. Prabhakaran, B. Vidgen, and Z. Waseem. Findings of the WOAH 5 shared task on fine grained hateful memes detection. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, Aug. 2021.

[125] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 11048–11064, 2022.

[126] M. Minderer, A. A. Gritsenko, A. Stone, M. Neumann, D. Weissenborn, A. Dosovitskiy, A. Mahendran, A. Arnab, M. Dehghani, Z. Shen, X. Wang, X. Zhai, T. Kipf, and N. Houlsby. Simple open-vocabulary object detection with vision transformers. *CoRR*, abs/2205.06230, 2022.

[127] R. Mokady, A. Hertz, and A. H. Bermano. Clipcap: CLIP prefix for image captioning. *CoRR*, abs/2111.09734, 2021.

[128] N. Muennighoff. Vilio: State-of-the-art visio-linguistic models applied to hateful memes. *CoRR*, 2020.

[129] H. Nam, J. Ha, and J. Kim. Dual attention networks for multimodal reasoning and matching. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2156–2164, 2017.

[130] M. Narasimhan and A. G. Schwing. Straight to the facts: Learning knowledge base retrieval for factual visual question answering. In *Computer Vision - ECCV*, pages 460–477, 2018.

[131] M. Narasimhan, S. Lazebnik, and A. G. Schwing. Out of the box: Reasoning with graph convolution nets for factual visual question answering. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems*, pages 2659–2670, 2018.

[132] D. Nguyen and T. Okatani. Improved fusion of visual and language representations by dense symmetric co-attention for visual question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 6087–6096, 2018.

[133] C. Nobata, J. R. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang. Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web, WWW*, pages 145–153, 2016.

[134] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[135] V. Ordonez, G. Kulkarni, and T. L. Berg. Im2text: Describing images using 1 million captioned photographs. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain*, pages 1143–1151, 2011.

[136] N. Ousidhoum, X. Zhao, T. Fang, Y. Song, and D.-Y. Yeung. Probing toxic content in large pre-trained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4262–4274, 2021.

[137] E. Papegnies, V. Labatut, R. Dufour, and G. Linarès. Graph-based features for automatic online abuse detection. In *Statistical Language and Speech Processing - 5th International Conference, SLSP*, volume 10583 of *Lecture Notes in Computer Science*, pages 70–81, 2017.

[138] F. Petroni, T. Rocktäschel, S. Riedel, P. S. H. Lewis, A. Bakhtin, Y. Wu, and A. H. Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP*, pages 2463–2473, 2019.

[139] S. Pramanick, D. Dimitrov, R. Mukherjee, S. Sharma, M. S. Akhtar, P. Nakov, and T. Chakraborty. Detecting harmful memes and their targets. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP*, pages 2783–2796, 2021.

[140] S. Pramanick, S. Sharma, D. Dimitrov, M. S. Akhtar, P. Nakov, and T. Chakraborty. MOMENTA: A multimodal framework for detecting harmful memes and their targets. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 4439–4455, 2021.

[141] J. Qian, A. Bethke, Y. Liu, E. M. Belding, and W. Y. Wang. A benchmark dataset for learning to intervene in online hate speech. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP*, pages 4754–4763, 2019.

[142] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[143] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML*, volume 139, pages 8748–8763, 2021.

[144] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

[145] M. Ren, R. Kiros, and R. S. Zemel. Exploring models and data for image question answering. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, pages 2953–2961, 2015.

[146] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

[147] P. Röttger, B. Vidgen, D. Nguyen, Z. Waseem, H. Z. Margetts, and J. B. Pierrehumbert. Hatecheck: Functional tests for hate speech detection models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP*, pages 41–58, 2021.

[148] A. Saha, M. M. Khapra, and K. Sankaranarayanan. Towards building large scale multimodal domain-aware conversation systems. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence*, pages 696–704, 2018.

[149] E. H. Sanchez, M. Serrurier, and M. Ortner. Learning disentangled representations via mutual information estimation. In *European Conference on Computer Vision*, pages 205–221. Springer, 2020.

[150] T. Schick and H. Schütze. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL*, pages 255–269, 2021.

[151] T. Schick and H. Schütze. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the Conference of the North American Chapter*

of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, pages 2339–2352, 2021.

[152] A. Schmidt and M. Wiegand. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media, SocialNLP@EACL*, pages 1–10, 2017.

[153] A. Schmidt and M. Wiegand. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10, 2017.

[154] D. Schwenk, A. Khandelwal, C. Clark, K. Marino, and R. Mottaghi. A-OKVQA: A benchmark for visual question answering using world knowledge. In *Computer Vision - ECCV*, volume 13668, pages 146–162, 2022.

[155] P. Sharma, N. Ding, S. Goodman, and R. Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 2556–2565, 2018.

[156] S. Shen, L. H. Li, H. Tan, M. Bansal, A. Rohrbach, K. Chang, Z. Yao, and K. Keutzer. How much can CLIP benefit vision-and-language tasks? In *The Tenth International Conference on Learning Representations, ICLR*, 2022.

[157] V. Shwartz, P. West, R. L. Bras, C. Bhagavatula, and Y. Choi. Unsupervised commonsense question answering with self-talk. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 4615–4629, 2020.

[158] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA*, 2015.

[159] A. Singh, R. Hu, V. Goswami, G. Couairon, W. Galuba, M. Rohrbach, and D. Kiela. FLAVA: A foundational language and vision alignment model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2022.

[160] V. K. Singh, S. Ghosh, and C. Jose. Toward multimodal cyberbullying detection. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systemss*, pages 2090–2099, 2017.

[161] H. Song, L. Dong, W. Zhang, T. Liu, and F. Wei. CLIP models are few-shot learners: Empirical studies on VQA and visual entailment. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL*, pages 6088–6100, 2022.

[162] W. Su, X. Zhu, Y. Cao, B. Li, L. Lu, F. Wei, and J. Dai. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*, 2019.

[163] S. Subramanian, W. Merrill, T. Darrell, M. Gardner, S. Singh, and A. Rohrbach. Reclip: A strong zero-shot baseline for referring expression comprehension. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL*, pages 5198–5215, 2022.

[164] Y. Sung, J. Cho, and M. Bansal. VL-ADAPTER: parameter-efficient transfer learning for vision-and-language tasks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 5217–5227, 2022.

[165] S. Suryawanshi, B. R. Chakravarthi, M. Arcan, and P. Buitelaar. Multimodal meme dataset (multioff) for identifying offensive content in image and text. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying, TRAC@LREC*, pages 32–41, 2020.

[166] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems*, pages 3104–3112, 2014.

[167] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1–9, 2015.

[168] H. Tan and M. Bansal. LXMERT: learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP*, pages 5099–5110, 2019.

[169] W. L. Taylor. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433, 1953.

[170] T. Thrush, R. Jiang, M. Bartolo, A. Singh, A. Williams, D. Kiela, and C. Ross. Winoground: Probing vision and language models for visio-linguistic compositionality. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 5228–5238, 2022.

[171] A. M. H. Tiong, J. Li, B. Li, S. Savarese, and S. C. H. Hoi. Plug-and-play VQA: zero-shot VQA by conjoining large pretrained models with zero training. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 951–967, 2022.

[172] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

[173] T. H. Trinh and Q. V. Le. A simple method for commonsense reasoning. *CoRR*, 2018.

[174] M. Tsimpoukelli, J. Menick, S. Cabi, S. M. A. Eslami, O. Vinyals, and F. Hill. Multimodal few-shot learning with frozen language models. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems, NeurIPS*, pages 200–212, 2021.

[175] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 5998–6008, 2017.

[176] R. Velioglu and J. Rose. Detecting hate speech in memes using multimodal deep learning approaches: Prize-winning solution to hateful memes challenge. *CoRR*, 2020.

[177] A. K. Vijayakumar, M. Cogswell, R. R. Selvaraju, Q. Sun, S. Lee, D. J. Crandall, and D. Batra. Diverse beam search for improved description of complex scenes. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 7371–7379, 2018.

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

[179] P. Wang, Q. Wu, C. Shen, A. R. Dick, and A. van den Hengel. FVQA: fact-based visual question answering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(10):2413–2427, 2018.

[180] X. Wang, X. Yi, H. Jiang, S. Zhou, Z. Wei, and X. Xie. Tovilag: Your visual-language generative model is also an evildoer. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 3508–3533, 2023.

[181] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL*, pages 13484–13508, 2023.

[182] Z. Wang, J. Yu, A. W. Yu, Z. Dai, Y. Tsvetkov, and Y. Cao. Simvlm: Simple visual language model pretraining with weak supervision. In *The Tenth International Conference on Learning Representations, ICLR*, 2022.

[183] Z. Waseem. Are you a racist or am I seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the First Workshop on NLP and Computational Social Science, NLP+CSS@EMNLP*, pages 138–142, 2016.

[184] Z. Waseem and D. Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the Student Research Workshop, SRW@HLT-NAACL*, pages 88–93, 2016.

[185] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

[186] J. Wen, P. Ke, H. Sun, Z. Zhang, C. Li, J. Bai, and M. Huang. Unveiling the implicit toxicity in large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1322–1338, 2023.

[187] G. Xiang, B. Fan, L. Wang, J. I. Hong, and C. P. Rosé. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In *21st ACM International Conference on Information and Knowledge Management, CIKM*, pages 1980–1984, 2012.

[188] H. Xu and K. Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *Computer Vision - ECCV*, volume 9911 of *Lecture Notes in Computer Science*, pages 451–466, 2016.

[189] J. Xu, T. Mei, T. Yao, and Y. Rui. MSR-VTT: A large video description dataset for bridging video and language. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 5288–5296, 2016.

[190] Y. Xu, L. Chen, Z. Cheng, L. Duan, and J. Luo. Open-ended visual question answering by multi-modal domain adaptation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP*, volume EMNLP 2020 of *Findings of ACL*, pages 367–376, 2020.

[191] Z. Yang, X. He, J. Gao, L. Deng, and A. J. Smola. Stacked attention networks for image question answering. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 21–29, 2016.

[192] Z. Yang, D. Yang, C. Dyer, X. He, A. J. Smola, and E. H. Hovy. Hierarchical attention networks for document classification. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016.

[193] Z. Yang, Z. Gan, J. Wang, X. Hu, Y. Lu, Z. Liu, and L. Wang. An empirical study of GPT-3 for few-shot knowledge-based VQA. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI*, pages 3081–3089, 2022.

[194] Y. Yao, A. Zhang, Z. Zhang, Z. Liu, T. Chua, and M. Sun. CPT: colorful prompt tuning for pre-trained vision-language models. *CoRR*, 2021.

[195] Q. Ye, H. Xu, G. Xu, J. Ye, M. Yan, Y. Zhou, J. Wang, A. Hu, P. Shi, Y. Shi, C. Li, Y. Xu, H. Chen, J. Tian, Q. Qi, J. Zhang, and F. Huang. mplug-owl: Modularization empowers large language models with multimodality. *CoRR*, abs/2304.14178, 2023.

[196] S. Yin, C. Fu, S. Zhao, T. Xu, H. Wang, D. Sui, Y. Shen, K. Li, X. Sun, and E. Chen. Woodpecker: Hallucination correction for multimodal large language models. *CoRR*, abs/2310.16045, 2023.

[197] F. Yu, J. Tang, W. Yin, Y. Sun, H. Tian, H. Wu, and H. Wang. Ernie-vil: Knowledge enhanced vision-language representations through scene graph. *arXiv preprint arXiv:2006.16934*, 2020.

[198] W. Yu, D. Iter, S. Wang, Y. Xu, M. Ju, S. Sanyal, C. Zhu, M. Zeng, and M. Jiang. Generate rather than retrieve: Large language models are strong context generators. In *The Eleventh International Conference on Learning Representations, ICLR*, 2023.

[199] R. Zellers, Y. Bisk, A. Farhadi, and Y. Choi. From recognition to cognition: Visual commonsense reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 6720–6731, 2019.

[200] A. Zeng, A. Wong, S. Welker, K. Choromanski, F. Tombari, A. Purohit, M. S. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, and P. Florence. Socratic models: Composing zero-shot multimodal reasoning with language. *CoRR*, 2022.

[201] A. Zeng, M. Attarian, B. Ichter, K. M. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. S. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, and P. Florence. Socratic models: Composing zero-shot multimodal reasoning with language. In *The Eleventh International Conference on Learning Representations, ICLR*, 2023.

[202] D. Zhang, R. Cao, and S. Wu. Information fusion in visual question answering: A survey. *Inf. Fusion*, 52:268–280, 2019.

[203] M. Zhang, T. Maidment, A. Diab, A. Kovashka, and R. Hwa. Domain-robust VQA with diverse datasets and methods but no target labels. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 7046–7056, 2021.

[204] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. T. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer. OPT: open pre-trained transformer language models. *CoRR*, abs/2205.01068, 2022.

[205] W. Zhang, G. Liu, Z. Li, and F. Zhu. Hateful memes detection via complementary visual and linguistic networks. *arXiv preprint arXiv:2012.04977*, 2020.

[206] T. Zhao, T. Zhang, M. Zhu, H. Shen, K. Lee, X. Lu, and J. Yin. Vl-checklist: Evaluating pre-trained vision-language models with objects, attributes and relations. *CoRR*, abs/2207.00221, 2022.

[207] X. Zhong. Classification of multimodal hate speech–the winning solution of hateful memes challenge. *arXiv preprint arXiv:2012.01002*, 2020.

[208] K. Zhou, J. Yang, C. C. Loy, and Z. Liu. Conditional prompt learning for vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 16795–16804, 2022.

[209] K. Zhou, J. Yang, C. C. Loy, and Z. Liu. Learning to prompt for vision-language models. *Int. J. Comput. Vis.*, 130(9):2337–2348, 2022.

[210] L. Zhou, H. Palangi, L. Zhang, H. Hu, J. J. Corso, and J. Gao. Unified vision-language pre-training for image captioning and VQA. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence*, pages 13041–13049, 2020.

[211] Y. Zhou and Z. Chen. Multimodal learning for hateful memes detection. *arXiv preprint arXiv:2011.12870*, 2020.

[212] D. Zhu, J. Chen, X. Shen, X. Li, and M. Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *CoRR*, abs/2304.10592, 2023.

[213] R. Zhu. Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution. *CoRR*, 2020.

[214] Y. Zhu, O. Groth, M. S. Bernstein, and L. Fei-Fei. Visual7w: Grounded question answering in images. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4995–5004, 2016.