VRLM: A Comprehensive Visual Reasoning Prompting Framework

Yunfei Ke(yk3108)
Data Science
Columbia University
New York, USA

Wangshu Zhu(wz2708)
Electrical Engineering
Columbia University
New York, USA

Haoyu Dong(hd2573)
Electrical Engineering
Columbia University
New York, USA

***Abstract*—Visual Question Answering (VQA) poses significant challenges in multimodal reasoning, especially when questions require external knowledge and step-by-step inference. In this work, we present a refined prompting framework called VRLM (Visual Reasoning via Language Models), built upon the original VCTP (Visual Chain-of-Thought Prompting) design. Our method retains the three-stage See–Think–Confirm loop but introduces several key enhancements: concatenated captions to provide coherent and dense visual context; dynamic few-shot construction using CLIP-based object-question relevance; and a multi-agent reasoning ensemble composed of GPT-3.5, DeepSeek V3, and LLaMA-7B. These improvements not only stabilize the model's reasoning but also allow iterative refinement and cross-model validation. Experiments on the A-OKVQA benchmark demonstrate that our framework outperforms previous prompting baselines by a significant margin, achieving a new state-of-the-art validation accuracy of 54.3%. Moreover, our framework demonstrates strong reasoning capabilities. Our work highlights the importance of strong prompting design and collaborative inference in building robust and interpretable vision-language systems.**

***Index Terms*—Chain-of-Thought Reasoning, Visual Question Answering, Multimodal Prompting**

I. Introduction

Visual Question Answering (VQA) aims to answer natural language questions about an image, requiring joint reasoning over vision, language, and commonsense knowledge. While pre-trained vision-language models have made notable progress, they still struggle with complex reasoning under sparse or ambiguous visual clues.

Recent work has extended Chain-of-Thought (CoT) [1] prompting to the visual domain. VCTP (Visual Chain-of-Thought Prompting) [2] pioneered a See–Think–Confirm pipeline that decomposes reasoning into interpretable steps. However, its effectiveness is limited by fragmented context construction, lack of cross-modal verification, and brittle predictions from a single frozen language model.

We propose **VRLM** (Visual Reasoning via Language Models), a refined system that systematically enhances each stage of the VCTP framework. Our design reimagines vision-language models as iterative reasoning agents, equipped with unified scene grounding, question-aware object selection via CLIP, and collaborative inference from multiple LLMs. This plug-and-play pipeline improves both answer accuracy and interpretability. Full experimental results demonstrate strong gains on the A-OKVQA [3] benchmark.

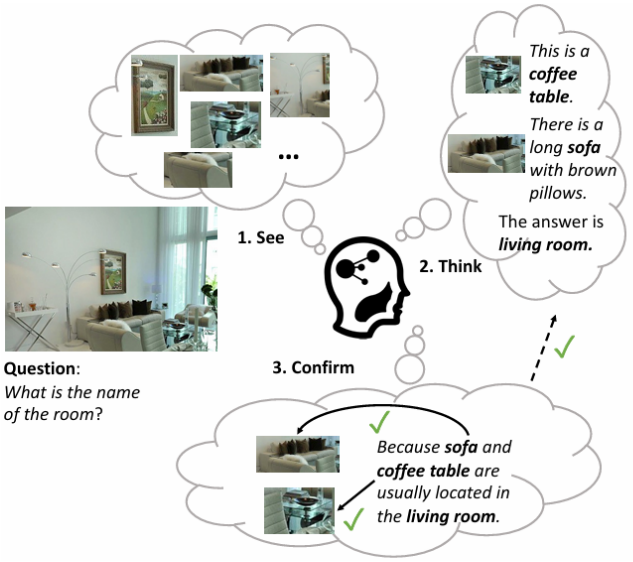

...
This is a **coffee table**.
There is a long **sofa** with brown pillows.
The answer is **living room.**
1. See
2. Think
3. Confirm
Question:
What is the name of the room?
Because ***sofa*** *and* ***coffee table*** *are usually located in the* ***living room****.*

Fig. 1: VRLM and original VCTP framework are inspired by how humanbeings react to visual questions and finally solve them.

II. Summary of the Original Paper

A. Related Work

Prompt Engineering Prompt engineering involves designing and optimizing natural-language prompts or prefixes to steer a frozen pre-trained model's behavior on downstream tasks without updating its weights [4]. Research in this area pursues four main objectives:

1) **Performance**: boost zero- and few-shot accuracy by selecting representative exemplars and crafting effective templates;
2) **Generalization**: create prompt formats that transfer across domains with minimal manual tuning;
3) **Interpretability**: expose the model's internal reasoning by structuring transparent prompt sequences;

4) **Automation**: reduce human effort via discrete search (e.g. AutoPrompt [5]) or continuous optimization (e.g. Prefix-Tuning [6], P-Tuning [7]).

Key branches include: discrete hard-prompting, soft/prefix tuning of continuous embeddings, and large-scale instruction tuning (e.g. FLAN [8], InstructGPT [9]).

In this work, we build on prompt engineering by transforming static, text-only prefixes into a three-stage interactive loop—See, Think, Confirm—that dynamically updates multimodal prompts based on intermediate outputs. During "See," we convert image features into an initial language prompt; in "Think," the model generates a provisional answer; and in "Confirm," the prompt is refined to verify correctness. This approach marries the efficiency and transparency of prompt engineering with the demands of complex visual reasoning, yielding substantial gains on the A-OKVQA benchmark.

Chain-of-Thought (CoT) reasoning embeds explicit, step-by-step "rationale" segments into prompts so that a frozen pre-trained model can perform multi-hop inference without updating its parameters. Researchers in this area aim to (1) uncover latent compositional reasoning skills in large models, (2) reduce error propagation by decomposing complex tasks into simpler sub-steps, (3) optimize the number and format of intermediate reasoning stages for different domains, and (4) enhance robustness through consensus or self-consistency mechanisms. Major research directions include:

- **Few-Shot CoT Prompting** [10]: providing exemplar chains of thought to demonstrate stepwise logic;
- **Zero-Shot CoT Triggers** [11]: using fixed trigger phrases (e.g. "Let's think step by step") to induce reasoning without examples;
- **Self-Consistency Voting** [12]: sampling multiple reasoning paths and selecting the most frequent answer;
- **Automated Rationale Generation** [13]: leveraging auxiliary models or optimization methods to create intermediate steps.

In this work, we draw on CoT principles by structuring our visual question answering pipeline into explicit intermediate reasoning stages—articulating hypotheses in natural language, then refining them based on multimodal evidence—to improve both interpretability and answer accuracy on A-OKVQA.

Visual Question Answering Visual Question Answering (VQA) combines image understanding and natural-language reasoning to generate accurate answers to questions about visual content. Researchers in this field aim to (1) learn robust joint representations of vision and language, (2) enable compositional reasoning over objects, attributes, and relations, (3) incorporate common-sense or external knowledge for open-domain queries, and (4) provide interpretable attention or reasoning traces.

Major research directions include:

- **Feature Fusion & Attention** (e.g. Bottom-Up & Top-Down Attention [14], LXMERT [15]);
- **Modular & Neural-Symbolic Methods** (e.g. Neural Module Networks **andreas2016neural**, NS-VQA **yi2018neural**);
- **Pretrained Vision–Language Transformers** (e.g. ViLBERT **lu2019vilbert**, VisualBERT **li2019visualbert**);
- **Knowledge-Based VQA** (e.g. A-OKVQA [16]).

In this work, we extend these VQA advances by converting region-level visual features into dynamic prompt prefixes and iteratively verifying candidate answers through a three-stage See–Think–Confirm loop, achieving higher accuracy and transparency on A-OKVQA.

B. Methodology of VCTP

VCTP (Visual Chain-of-Thought Prompting), designed for knowledge-based visual reasoning. The model mimics human reasoning by structuring the process into three interactive stages: **See**, **Think**, and **Confirm**, executed in an iterative loop.

1) Overall Framework: Given an image-question pair $\{I, Q\}$, the framework proceeds as follows:

- **See Module**: Detects all candidate objects c_n using a scene parser (e.g., Faster R-CNN) and generates a global image caption.
- **Think Module**: Uses an LLM to attend to key visual concepts, describes them with a regional image captioner, and predicts an answer.
- **Confirm Module**: Requires the LLM to generate a supporting rationale for the predicted answer, verifies this rationale with a cross-modal classifier (e.g., CLIP), and checks for consistency in prediction.

This pipeline is repeated until the answer converges or a maximum number of iterations is reached. The detailed algorithm can be found in **Algorithm 1**.

Algorithm 1 Pipeline of the VCTP Framework

```
Require: Input Image $I$, Question $Q$
Ensure: Final Answer $a^*$ and Reasoning Process $R$
$\{c_n\}_{n=1}^{N} \leftarrow \texttt{ImageParser}(I)$ {Object detection (See)}
$\texttt{cap}_g \leftarrow \texttt{GlobalCaptioner}(I)$
Initialize $a_0 \leftarrow \emptyset$, $i \leftarrow 0$
Initialize prompts: $P_{\text{con},0}$ (Q-A-Rationale), $P_{\text{thk}}$ (Q-Object)
repeat
    $i \leftarrow i + 1$
    $c_i \leftarrow \texttt{LLM.Attend}(\{c_n\}, Q, P_{\text{thk}})$
    $\texttt{cap}_i \leftarrow \texttt{Captioner}(c_i, I)$
    $P_{\text{con},i} \leftarrow P_{\text{con},i-1} + \texttt{cap}_i$
    $a_i \leftarrow \texttt{LLM.Predict}(P_{\text{con},i}, Q)$
    $r_i \leftarrow \texttt{LLM.Confirm}(P_{\text{con},i}, Q, a_i)$
    if $\texttt{Verify}(r_i, I) > \texttt{threshold}$ then
        $P_{\text{con},i} \leftarrow P_{\text{con},i} + r_i$
    end if
until $a_i = a_{i-1}$ or $i = m_{\text{Iter}}$
$a^* \leftarrow a_i$, $R \leftarrow (\{c_j, \texttt{cap}_j\}_{j=1}^{i}, r_i)$
return $\{a^*, R\}$ =0
```

2) Advantages: VCTP provides the following benefits:

- **Effectiveness**: It consistently outperforms few-shot baselines on knowledge-based VQA benchmarks.
- **Interpretability**: Each answer is supported by a traceable rationale verified against the image.
- **Efficiency**: It avoids the computational cost of full model fine-tuning by relying on in-context prompting.

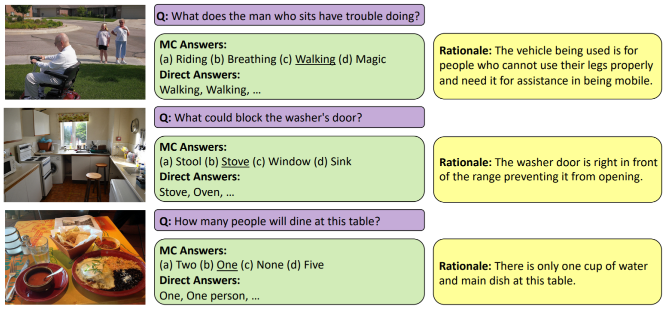
C. A-OKVQA Benchmark

A-OKVQA is a large-scale, crowdsourced benchmark for knowledge-based visual question answering. Each example consists of an image, a question, four multiple-choice (MC) options, ten reference direct answers (DA), and a human-written *rationale* explaining the reasoning steps needed to arrive at the correct answer. This rich annotation allows evaluation not only of answer accuracy but also of explanation quality and model transparency.

Table I summarizes the basic statistics of A-OKVQA. Figure 2 shows three representative examples, each illustrating the question prompt (purple), the MC options and DA references (green), and the human rationale (yellow).

Statistic	**Train**	**Val**	**Test**
Number of QA pairs	17,117	1,096	6,690
MC options per question	4	4	4
DA references per question	10	10	10
Human rationales provided	yes	yes	yes

TABLE I: Basic statistics of the A-OKVQA dataset.



Q: What does the man who sits have trouble doing?

MC Answers:
(a) Riding (b) Breathing (c) Walking (d) Magic
Direct Answers:
Walking, Walking, ...

Rationale: The vehicle being used is for people who cannot use their legs properly and need it for assistance in being mobile.

Q: What could block the washer's door?

MC Answers:
(a) Stool (b) Stove (c) Window (d) Sink
Direct Answers:
Stove, Oven, ...

Rationale: The washer door is right in front of the range preventing it from opening.

Q: How many people will dine at this table?

MC Answers:
(a) Two (b) One (c) None (d) Five
Direct Answers:
One, One person, ...

Rationale: There is only one cup of water and main dish at this table.

Fig. 2: Representative A-OKVQA examples: (left) input image, (top-right) question in purple, MC answers and DA references in green, (bottom-right) human-written rationale in yellow.

D. Key Results of Basic VCTP

Table II compares the A-OKVQA performance of our basic VCTP (without BLIP enhancements) against strong few-shot and prior fully-supervised baselines. On the validation split, VCTP achieves 46.4% accuracy—an absolute gain of 3.6 points over the previous best few-shot prompt (PICa) and 4.9 points over Chain-of-Thought prompting—while also surpassing GPV-2 (48.6% val, 40.7% test) on the test split. Baselines mentioned can be found in Appendix.A.

Method	**Val**	**Test**
Pythia (2018)[17]	25.2	21.9
ViLBERT (2019)[18]	30.6	25.9
LXMERT (2019)[19]	30.7	25.9
KRISP (2021)[20]	33.7	27.1
GPV-2 (2022)[21]	48.6	40.7
PICa* (2022) [22]	42.4	43.8
CoT* (2022) [1]	41.5	43.7
lightgray VCTP* [2]	**46.4**	**46.0**

TABLE II: A-OKVQA accuracy (%) for Basic VCTP vs. baselines.

III. Methodology

A. Limitations of the Original Method

While the baseline method effectively framed A-OKVQA as a language modeling task with few-shot prompting, it suffers from several critical limitations:

- **Fragmented Visual Context:** The original approach provides separate global and regional captions—VinVL confidence-based object descriptions and CLIP-derived global captions. This fragmentation can lead to incoherent visual context, making it harder for the model to form a unified understanding of the scene.
- **Model Capacity and Reasoning Depth:** Earlier generations of language models (e.g., OPT) show limited capability in performing multi-hop reasoning or handling ambiguous visual questions that require disambiguation across spatial or commonsense clues.
- **Single-agent Reasoning:** The vanilla framework relies on a single model's output, which is prone to hallucinations and lacks verification. Without any cross-check or external signal, the predictions may drift toward unstable or inconsistent answers.

B. Proposed Improvements

To address the above limitations, we propose a set of architectural and procedural enhancements targeting visual grounding, reasoning robustness, and model coordination.

- **Concatenated Caption Strategy:** We replace the separate object-based captions with a unified, concatenated caption that merges all attribute-rich object descriptions (from VinVL) into a single narrative block. This approach mimics the COCO-style ground truth captions and improves coherence by:
  - Providing the model with complete scene coverage in one context window.
  - Reducing ambiguity from disconnected object mentions.
  - Encouraging globally consistent reasoning grounded in dense visual semantics.

- **Stronger Language Models:** We systematically evaluate multiple state-of-the-art LLMs with varying architectures and tuning objectives:
  - `GPT-3.5 Turbo`: The original agent used in VCTP.
  - `DeepSeek V3`: A competitive open-source model with strong instruction-following skills.
  - `LLaMA-7B`: A decoder-only transformer, efficient for local reasoning.

  All models are used with identical prompts and caption inputs to isolate their reasoning capabilities.
- **Multi-Agent Collaboration**: To enhance robustness, we propose a two-stage inference strategy:
  - In the first stage, an agent generates an answer and reasoning chain.
  - In the second stage, a reviewer agent (DeepSeek or GPT) critiques and optionally refines this answer.

  This improves factual correctness and enables error correction through self-consistency and peer review.
- **Voting-based Answer Aggregation:** Beyond sequential refinement, we also experiment with an ensemble-style architecture where three agents independently answer the same question, and their responses are aggregated via majority voting. This method is shown to suppress outlier generations and improve performance on difficult cases.
- **Unified Prompt Structure:** All models follow a shared prompt template, which consists of:
  - A merged scene caption (see above).
  - Object-level attributes derived from scene graphs.
  - A question and few-shot in-context examples, where available.

  This ensures consistency in evaluation and removes confounding factors across models.

Collectively, these modifications improve both the factual reliability and semantic richness of answers. Our method enables broader visual-text alignment, stronger reasoning generalization, and collaborative decision-making among heterogeneous agents.

IV. Implementation

A. Data Preprocessing

This section describes the necessary cleansing and transformation of the original COCO and A-OKVQA datasets to produce the standardized intermediate files required by the downstream See/Think/Confirm modules. The overall workflow comprises four steps: COCO caption reorganization, sample line-index mapping, CLIP feature precomputation, and object similarity computation.

1) COCO Caption Reorganization: To ensure that every image can uniformly access a set of natural-language descriptions, we merge the caption annotations from COCO-2014 and COCO-2017. First, we read all captions from the COCO-2014 training and validation files (captions_train2014.json and captions_val2014json) and aggregate the multiple captions associated with each image_id. Then, using the COCO-2017 directory structure (train2017 and val2017), we remap those COCO-2014 captions onto the corresponding COCO-2017 image_ids, writing out the results as captions_train2017json and captions_val2017.json. This process not only unifies the caption format across dataset versions but also guarantees that every COCO-2017 image has a complete set of human-written descriptions, eliminating the need for online caption generation during inference.

2) Sample Line-Index Mapping: For efficient lookup, we assign a unique line index to each question sample in the A-OKVQA training and validation sets. We parse the original aokvqa_v1p0_train.json and aokvqa_v1p0_valjson, pairing each ”image_id¡-¿question_id” key with its sequential position in the list. The resulting mapping from line index to sample key is saved as aokvqa_qa_line2sample_idx_train2017.json and val2017.json. In downstream modules, this mapping allows constant-time alignment between CLIP features or object similarity dictionaries and their corresponding samples, significantly accelerating vector retrieval and matching.

3) CLIP Feature Precomputation: To support efficient Few-Shot retrieval and image–text similarity comparisons, all question texts and their corresponding images are projected offline into a shared embedding space using the OpenAI CLIP ViT-B/16 model. We encode each question into a fixed-dimensional text embedding and each COCO image into an image embedding, then save these as NumPy arrays (question.npy and convertedidx_image.npy) for both the training and validation splits. During See and Think, cosine similarity can be computed directly on these precomputed matrices, obviating the need for repeated online forward passes through the large CLIP model.

4) Object Similarity Computation: In order to supply the See module with high-quality supervision examples, we compute, in an offline pass, a relevance score for every detected object in each training sample relative to its answer or rationale text. We load the training set's scene graphs (object classes and detection confidences) and A-OKVQA annotations, then for each ”image_id¡-¿question_id” sample:

- **Answer mode:** Encode all ground-truth answers and each object's class name with the CLIP text encoder, computing cosine similarity between answer embeddings and object embeddings.
- **Rational mode:** Count how often each object's name appears in the rationale sentences as a frequency-based relevance score.

The resulting object_name → score mappings are serialized into train_object_select_¡metric¿.pk. At inference time, the See module loads these dictionaries to build Few-Shot prompts that teach the model which object is most relevant to a given question, without exposing the actual answers.

B. See Module

In the See stage, the system transforms raw visual detections and textual annotations into a focused attention signal for downstream reasoning. We decompose this process into six sequential phases, each of which is described below.

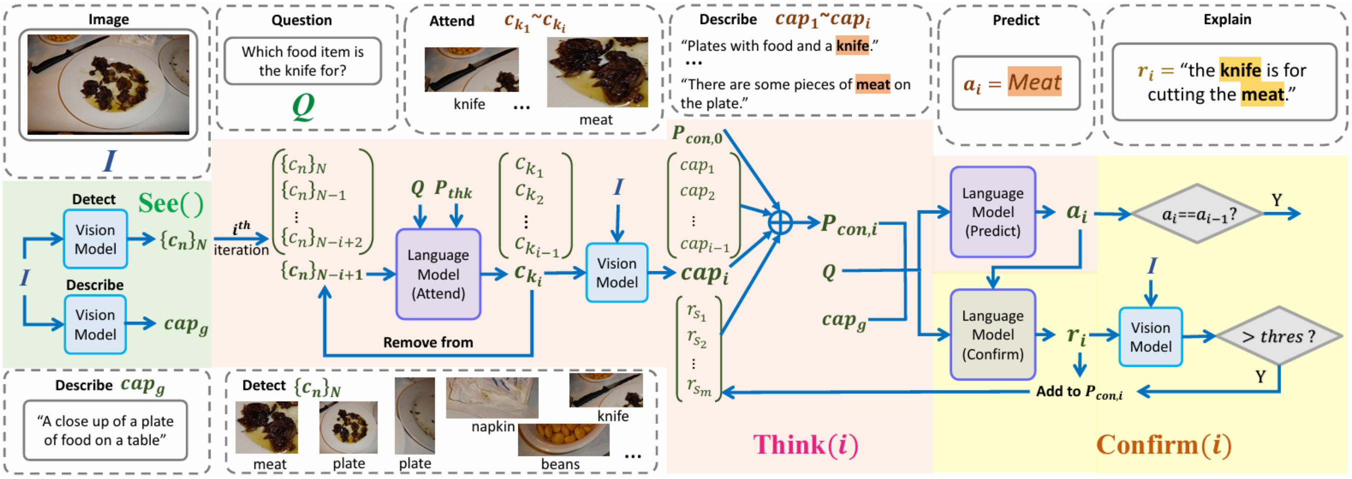


Fig. 3: The framework of our VCTP. Given an image-question pair, we first use the see module to detect all object candidates in the image and translate the whole image into a global description. Then, the think module adopts an LLM to attend to the key visual concepts, transforms the selected concept into a language description with a captioner, and leverages the LLM to predict an answer. The confirm module requires the LLM to continue the generate the supporting rationale, verify whether the rationale is consistent with the image content, and ensure that the same answer can be produced when the rationale is added to the prompt in the next iteration. Algorithm 1 provides a detailed description of the VCTP framework.

1) Prompt Composition: For each input image, three parallel JSON files provide complementary region descriptions. First, the scene graph relations file yields a list of detected objects, each annotated with its bounding box coordinates (rect), class label (class), and detection confidence (conf). Second, the attribute file augments each detection with a variable-length list of semantic attributes (for example, color or state) and corresponding confidence scores. Third, the dense caption file offers a natural-language description of the same region, keyed by its bounding box.

The See module aligns these three sources by index, ensuring that every detection is represented by a single record containing all four pieces of information. This unified representation forms the raw material for candidate region selection. Once all detections are loaded and aligned, the See module sorts them in descending order of detection confidence. The result, termed candidate_regions, is a one-dimensional list of entries of the form (confidence, class_label, attributes, dense_caption).

By ranking regions in this way, the system prioritizes high-confidence detections and guarantees that the most salient objects are considered first. The full list is retained to support multi-round iteration, allowing each round to remove previously attended regions and thereby focus on a fresh object.

2) Few-Shot Prompt Assembly: To guide the model's attention mechanism, we employ in-context learning with carefully selected examples. Given the test question, the module invokes get_interactive_context_keys, which leverages precomputed CLIP embeddings to retrieve the top-K most similar training samples under either question-only or combined image-question similarity metrics. For each of these K samples, the corresponding object-relevance dictionary is loaded and its highest-scoring object is identified.

These K examples provide explicit demonstrations of how a question and a set of candidate objects map to a single correct choice, forming the pedagogical foundation for the model's own selection. The Few-Shot examples are concatenated into a demonstration prefix. Each block follows the template:

```
{
  "Question": "What is the woman holding?",
  "Options": "woman", "book", "cup", "dog",
  "The most related option is book."
}
```

After these K blocks, the prompt appends the actual test question along with the names of all candidate regions, in the format:

```
{
  "Question": "What is the boy doing?"
  "Options": "boy", "ball", "bench", "tree"
  "The most related option is ____"
}
```

This structure ensures that the language model first internalizes the desired mapping from question to attended object, then produces its own selection in a single completion.

3) Result Extraction: With the prompt fully constructed, the See module dispatches it to the selected backbone—whether the OpenAI API, a local OPT or LLaMA model, or Google's Gemini. Critically, we constrain the model's output to exactly one of the candidate object names by applying a logit_bias: the tokens corresponding to each object label receive a strong positive bias, while all other tokens are suppressed. In the case of local models, this effect is emulated by generating a short sequence and then selecting the highest-scoring candidate token from among the object-name token IDs. As a result, the model deterministically outputs a single object name, reflecting its most confident attention choice. Finally, the selected object is mapped back to its corresponding entry in candidate_regions. That entry is removed from the list to prevent repeated attention in subsequent rounds. The chosen region is then passed to the Think module as the focal visual cue for answer generation, while the reduced candidate_regions list readies the See module for the next iteration should multi-round reasoning be required.

C. Think Module

The Think module takes as input the attended object from the See stage, together with a global image caption and any previously generated reasoning, and produces both a candidate answer and its supporting rationale. Its core consists of constructing a multi-component prompt, issuing repeated inferences to quantify model confidence, and selecting the most likely answer through log-prob maximization. Below we describe each phase in detail.

1) Prompt Composition: At the start of each reasoning step, the module builds a textual context that guides the language model toward the correct answer. The context comprises three elements. First, a global caption—either one of the ground-truth COCO captions or an optional BLIP2-generated summary—serves to ground the model in the overall scene. Immediately following this, all previously generated reasoning snippets (the accumulated "chain-of-thought") are concatenated, ensuring continuity across iterative rounds. Second, the description of the currently attended object—extracted from its scene-graph attributes and dense caption—is appended, directing the model's focus to that specific region. Finally, a block of K Few-Shot question-answer exemplars is inserted: each exemplar consists of a training-set caption, its corresponding question, the model's answer, and (when chain-of-thoughts are enabled) the exemplar's rationale. After these demonstration blocks, the actual test question is presented, prefaced by the phrase "Answer: The answer is," and left for the model to complete. This layered prompt design—global context, past reasoning, focal object description, pedagogical examples, and test question—provides a rich scaffold that conditions the model's prediction on both visual and textual evidence.

2) Few-Shot QA Demonstrations: The K exemplars are retrieved via a similarity-based index: for the current question, precomputed CLIP embeddings of training questions and images are compared against the test question embedding, and the top K most similar samples are selected. For each, the module loads its ground-truth answer and rationale, then formats a block of the form

```
{
  "Context": "A man wearing glasses is sitting at a table in a restaurant."
  "Question": "What is the man drinking?"
  "Answer": "The answer is coffee. He is holding a cup with dark liquid in front of his mouth."

}
```

By mimicking the model's own output structure on known examples, we implicitly teach it how to marshal evidence—in this case, the attended object and scene context—into a coherent answer and accompanying explanation. These demonstration blocks appear verbatim in the prompt immediately before the test-question segment.

3) Ensemble Sampling and Answer Selection: To guard against the variability inherent in autoregressive generation, the Think module repeats the same prompt invocation N times. Each pass returns both the predicted answer string and a sequence of per-token log-probabilities. The sum of these log-probabilities serves as a proxy for the model's overall confidence in that particular generation. By sampling multiple times, we obtain a small ensemble of candidate answers, each with its own confidence score.

After N independent inferences, the module compares the total log-probabilities of each candidate. The answer corresponding to the highest sum is chosen as the final prediction for that reasoning round. This confidence-based selection effectively amplifies the model's most self-consistent response, reducing the chance of atypical or erroneous outputs. The retained log-probability also serves as a diagnostic metric, enabling real-time monitoring of model certainty and facilitating downstream modules to decide whether further iterations are necessary.

D. Confirm Module

The Confirm stage serves two critical purposes: validating that the model's generated reasoning remains faithful to the actual image content, and determining when iterative rounds of See–Think–Confirm can be terminated because the answer has stabilized. By combining optional CLIP-based consistency checks with a simple convergence criterion and dynamic prompt updates, Confirm closes the reasoning loop and prevents unnecessary computation.

1) CLIP-Based Consistency Verification: When the flag –with_clip_verify is enabled, each newly generated reasoning sentence undergoes a consistency check against the image via the CLIP model. Concretely, the Confirm module first encodes every clause of the chain-of-thought into text embeddings using the CLIP text encoder; it simultaneously retrieves the precomputed image embedding. By measuring the cosine similarity between each reasoning embedding and the image embedding, the system identifies statements whose similarity falls below a configurable threshold. Inconsistent sentences are discarded from the reasoning chain, while the remaining statements are rejoined and carried forward. This filter ensures that only those parts of the model's explanation that truly reflect the visual evidence survive, tightening the alignment between language and vision.

2) Answer Convergence and Early Exit: Beyond sentence-level validation, Confirm tracks the stability of the model's predicted answer across successive rounds. After each Think invocation, the module compares the newly predicted answer text to the answer produced in the previous round. If the two strings are identical, the system concludes that the model's reasoning has converged and halts any further See–Think iterations. This early-stop mechanism not only saves compute by avoiding superfluous calls but also guards against oscillatory behavior in which the model might flip back and forth between competing answers.

3) Dynamic Object Addition and Prompt Expansion: Confirm also orchestrates the dynamic growth of the reasoning context. In every new round, the object selected by See is appended to the noticed_attr_list, and the retained reasoning sentences are appended to the thoughts list. When the next Think prompt is constructed, it automatically incorporates all previously verified reasoning and the descriptions of every object that has been attended so far. This incremental prompt expansion preserves the continuity of the reasoning narrative and allows the model to integrate new visual evidence without losing sight of earlier insights. It is this interplay that completes the cyclical See→Think→Confirm paradigm, enabling robust, multi-round visual reasoning.

E. Multi-Agent Framework

To explore whether collaborative agent interactions can enhance visual reasoning, we designed two multi-agent variants that contain cooperative critique and revision on top of the base See–Think–Confirm pipeline. Each agent in these setups plays a distinct role—either producing initial answers, evaluating peer outputs, or synthesizing final responses. This structured interaction allows us to study how decentralized deliberation may improve accuracy or robustness beyond single-agent reasoning.

1) Two-Agent Collaborative Critique:

```
Here is an answer and reasoning from another agent:
Answer: <agent1_answer>. <agent1_reasoning>.

Please examine it critically. If you agree, keep it.
Otherwise, provide a better answer and justification.
```

The first structure involves two agents operating sequentially. The first agent receives the constructed prompt and produces both an answer and reasoning. This response is then passed verbatim to a second agent with a new instruction as above. The second agent either validates the initial output or proposes a revision. This setting encourages critical evaluation while introducing minimal overhead. The final answer is selected based on the second agent's output and compared to the ground truth for accuracy. While this two-agent chain adds one extra pass of computation, it effectively filters out some incorrect generations and improves answer robustness in ambiguous or underspecified cases.

2) Three-Agent Voting with Majority Consensus: The second structure extends collaboration via a three-agent ensemble with majority voting. Each agent (e.g., DeepSeek, GPT-3.5, LLaMA) independently generates an answer for the same prompt using identical context. Their answers are then aggregated, and a simple consensus mechanism is applied:

- If at least two agents agree (i.e., predicted the same answer string), that answer is selected.
- If all three disagree, one answer is selected at random to ensure continuity.

This method approximates a decentralized decision-making process where independent models cast votes based on their understanding. In our experiments, we found that such voting helped suppress idiosyncratic model errors and stabilized accuracy in hard examples. Notably, the chosen final answer is re-evaluated for accuracy using the original ground-truth comparison function, allowing us to measure performance uplift due to cross-model agreement.

V. Results & Discussion

A. Overall Performance on A-OKVQA

Table III summarizes validation accuracy of our VRLM framework against prior baselines and our VCTP reproduction. Notably, our re-implementation of VCTP scores only 41.6%, well below the 46.4% originally reported. In contrast, even the simplest VRLM variant—pairing our retrieval module with GPT-3.5—achieves 47.6%, recovering the VCTP gap and surpassing GPV-2 (48.6%) by 1.2 points.

Replacing GPT-3.5's reasoning with DeepSeek V3 further boosts performance to 50.6%, highlighting the impact of stronger retrieval. Swapping in an instruction-tuned LLaMA model raises accuracy to 53.1%, confirming that larger, more capable LMs produce more reliable multimodal reasoning. Finally, ensembling all three agents via a self-consistent voting scheme yields a new high of 54.2%, outperforming the best single agent (VRLM+LLaMA) by 1.1 points and establishing a new state of the art on A-OKVQA's validation split.

Method	**Val(%)**
GPV-2 (2022)	48.6
PICa(2022)	42.4
CoT(2022)	41.5
VCTP original paper	**46.4**
VCTP our reproduction	**41.6**
VRLM + GPT-3.5⋆	47.6
VRLM + DeepSeek V3⋆	50.6
VRLM + LLaMA⋆	53.1
VRLM + Voting Agents⋆	**54.3**

TABLE III: A-OKVQA validation accuracy (%) for Basic VCTP vs. baselines and our VRLM variants.

B. Output Analysis

1) Rationale Analysis:

Listing 1: Example 1: Cuisine Reasoning - Chinese vs. Italian

```
{
  "Question": "What kind of cuisine is this?",

  "Original Model Rationale": [
    "The dish contains noodles and broccoli, which
        are common ingredients in Chinese cuisine.",
    "The dish contains common ingredients found in
        Chinese cuisine, such as noodles and
        broccoli."
  ],

  "Our Model Rationale": [
    "The combination of pasta, broccoli, and oil/
        vinegar dressing is characteristic of
        Italian cuisine.",
    "The use of broccoli and pasta suggests a
        Mediterranean or Italian origin."
  ]
}
```

Listing 2: Example 2: Skateboarding Environment - Urban vs. Park

```
{
  "Question": "In what type of environment are they
      most likely riding skateboards?",
    "Original Model Rationale": [
    "The context repeatedly mentions a park,
        indicating that the skateboarders are likely
         riding their skateboards in a park
        environment.",
    "There is a lot of skateboarding and skating in
        a park."
  ],
  "Our Model Rationale": [
    "The presence of graffiti, buildings, and a town
         square suggests a city or urban setting
        where skateboarding is commonly practiced.",
    "Several vehicles are parked on the side of a
        road, hinting at a street-based city scene."
  ],
}
```

Listing 3: Example 3: Cuision Environment - General vs. Specific

```
{
  "Question": "What type of cuisine would be
      purchased here?",
    "Original Model Rationale": [
    "The food truck is full of food, indicating that
         it likely serves street food cuisine."
  ],
  "Our Model Rationale": [
    "The tamale truck and taco truck suggest that
        this place serves Mexican cuisine."
  ],
}
```

1) **Multimodal Grounding**: Our model reasons beyond keyword matching, incorporating diverse scene elements.
2) **Higher Specificity**: It links visual and semantic features to specific cultural or environmental contexts.
3) **Robust Reasoning**: Our rationales generalize better and avoid overfitting to superficial cues (e.g., merely seeing "noodles" and inferring Chinese).

Our VRLM framework not only surpasses the original VCTP—improving validation accuracy from 46.4 % to 54.2 %—but also establishes an updated framework. This gain is driven by holistic enhancements across (i) the retrieval–reasoning–voting architecture, (ii) stronger backbone language models, (iii) refined chain-of-thought prompting, and (iv) richer caption data. A detailed ablation study and further analysis are presented in the next section.

VI. Abalation & Futher Discussions

In this section, we will carry out further discussions on broader experimental settings to support the extraordinary performance of our proposed framework.

A. Caption Replacement

We investigate whether replacing the *VinVL confidence caption* originally used in VCTP with a **concatenated COCO ground truth (GT) caption**[1] can improve answer accuracy and reasoning quality in A-OKVQA.

Method	Single Caption	Multi Caption
VRLM + GPT-3.5	41.60	47.58
VRLM + DeepSeek V3	45.64	50.62
VRLM + LLaMA 7B	47.13	53.34

TABLE IV: A-OKVQA validation accuracy (%) on VRLM for single caption vs. mixed captions.

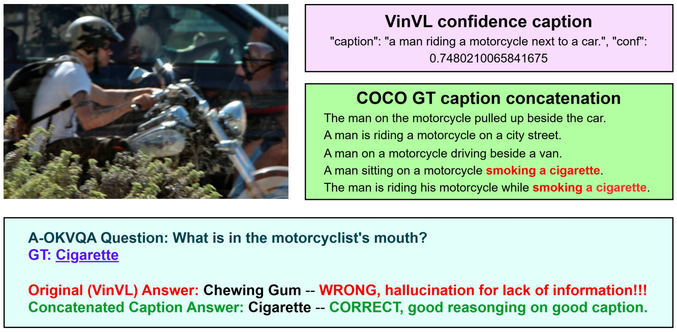
Across the three backbone architectures studied (GPT-3.5 Turbo, Qwen, and DeepSeekV3), substituting the single confidence caption with its concatenated GT counterpart raises exact-answer accuracy by **5–6 percentage points** on average.[2] Qualitatively, the richer textual context mitigates information sparsity and reduces hallucinations during chain-of-thought reasoning.

Analysis are as follows:

1) **Information completeness.** The concatenated caption aggregates multiple human annotations, yielding a denser description that covers salient objects (*"cigarette"*) and scene relations (*"man riding his motorcycle while smoking"*). This completeness supplies the language model with factual priors that are otherwise missing from confidence captions, guiding its multi-step reasoning (Fig. 4).
2) **Reduced exposure bias.** Confidence captions are generated by an external vision-language model whose distribution may not align with the downstream question. When its highest-confidence sentence omits critical attributes, the resulting prompt provides a misleading

[1] The concatenated caption is formed by joining *all* GT sentences associated with the image, preserving their order.

[2] All models were evaluated under identical prompting and decoding settings to isolate the effect of caption choice.



VinVL confidence caption

"caption": "a man riding a motorcycle next to a car.", "conf": 0.7480210065841675

COCO GT caption concatenation

The man on the motorcycle pulled up beside the car.
A man is riding a motorcycle on a city street.
A man on a motorcycle driving beside a van.
A man sitting on a motorcycle **smoking a cigarette**.
The man is riding his motorcycle while **smoking a cigarette**.

A-OKVQA Question: What is in the motorcyclist's mouth?
GT: Cigarette

Original (VinVL) Answer: Chewing Gum -- WRONG, hallucination for lack of information!!!
Concatenated Caption Answer: Cigarette -- CORRECT, good reasonging on good caption.

Fig. 4: Illustrative example: the concatenated caption explicitly mentions *“smoking a cigarette”*, allowing the model to answer *“Cigarette”* correctly, whereas the single confidence caption omits this detail, leading to the hallucinated answer *“Chewing Gum”*.

prior. Concatenation averages out such idiosyncrasies, lowering the chance of error-propagation.

3) **Stronger grounding signal.** By repeating key nouns and verbs across sentences, concatenated captions amplify attention to high-impact tokens, effectively reinforcing visual grounding cues within the transformer's cross-modal attention layers. Empirically, this manifests as a higher proportion of correct entity extractions during intermediate reasoning steps.
4) **Model-agnostic gains.** The consistent $+5-6\%$ improvement across heterogeneous backbones indicates that richer caption context is broadly beneficial and orthogonal to model capacity, suggesting a low-cost, data-centric avenue for enhancing VLM performance.

Replacing confidence captions with concatenated GT captions is a simple yet effective ablation that simultaneously boosts accuracy and reasoning fidelity while not necessarily increasing inference time given modern model capacity. Future work could explore automatic caption fusion strategies that retain these benefits without relying on ground-truth annotations.

B. Did the Reasoning Model (`deepseek-r1`) really Fail?

The `r1` variant was designed to enhance answer quality by introducing a dedicated reasoning module. However, experimental results showed otherwise: the `r1` model only achieved an accuracy of **33.97%**, which is significantly lower than the `v3` baseline at **45.64%**.

Method	**Validation (%)**
VRLM + DeepSeek R1	33.97
VRLM + DeepSeek V3	45.64

TABLE V: A-OKVQA validation accuracy (%) on single caption for reasoning model R1 vs. general model V3.

1) Failure Modes of Reasoning-Augmented Models:

- **Structural Noise and Prompt Drift:** The additional reasoning layer introduced more tokens and shifted the model's focus away from concise answering. As the prompt grew longer, the likelihood of deviating from the expected “Answer: ...” format increased.
- **Unreliable Reasoning Completion:** In some samples, the reasoning was logically sound but did not conclude with a clear final answer. In some cases, the answer was embedded within a paragraph, or followed non-standard phrasing, making it harder to extract and evaluate.

2) Evaluation Fragility for Expressive Predictions:

```
V3: Answer: smartphone. (acc: 0.0)
R1: Answer: mobile phone. (acc: 0.3)
Ground Truths: {phone, phone, phone, phone, cell
    phone, mobile phone, vodafone, cell phone, phone
    , phone}
```

The evaluation protocol in A-OKVQA is based on a voting mechanism: each question has ten free-form human-provided answers. A model prediction is considered fully correct (score = 1.0) if it matches at least 4 annotator responses, partially correct (score = 0.9, 0.6, or 0.3) if it matches three, two, or one, and incorrect (score = 0.0) otherwise. While this strategy captures ambiguity and allows for flexible evaluation, it also introduces fragility in practice.

Specifically, our models (e.g. DeepSeek R1 and V3) often produce more specific or semantically richer answers. However, these answers may not occur frequently in the ground-truth annotations. For example, a prediction like "`mobile phone`" might only appear once among the ten annotations, whereas a simpler variant like "`phone`" appears six times. Consequently, the former receives a lower score (0.3) despite being equally or even more accurate.

This scoring scheme thus penalizes informative predictions and increases false negatives, especially when models use uncommon synonyms (e.g., "`smartphone`", "`flip phone`") or more descriptive terms (e.g., "`light brown horse`" vs. "`horse`"). The issue is further amplified when reasoning chains wrap the answer in nonstandard formats, making extraction harder and alignment less reliable.

We suggest complementing this heuristic matching with semantic-aware metrics (e.g., BERTScore) or relaxing the matching threshold to include any correct answer present in the annotation set, even if below the majority threshold.

C. Multi-Agent Analysis

In addition to single-agent models, we explored a two-agent pipeline (`multi-agent`) where:

- Agent A generates an answer and a rationale;
- Agent B evaluates this output and either confirms it or proposes a better alternative.

This collaborative setup aims to simulate self-correction and refinement. However, it achieved 51.24%, only a small or no improvement from a single deepseek v3 model.

Method	Validation(%)
VRLM + DeepSeek V3	50.62
VRLM + DeepSeek two-agents	51.24

TABLE VI: A-OKVQA validation accuracy (%) on merged caption for single model vs. two-agent.

a) Observations and Analysis::

- **Redundant Structure:** The VCTP framework already includes rich visual and linguistic context. Adding an extra agent often led to repetition rather than meaningful refinement.
- **Limited Disagreement:** The second agent frequently agreed with the first, even when the answer was incorrect. Without enforced diversity or error detection, the multi-agent process became a confirmation rather than a critique.
- **Increased Prompt Complexity:** Passing the full context, first answer, and reasoning into a new prompt expanded the token length and risked prompt overflow or truncation, further degrading performance.
- **Noisy Corrections:** When the second agent disagreed, the revised answer was sometimes worse or semantically inconsistent with the original context, leading to accuracy drops.

b) Improved Diversity via Independent Voting: To further enhance robustness, we experimented with a multi-model voting strategy instead of a critique-based chain. Specifically, we ensemble three independently trained models: `GPT-3.5`, `DeepSeek V3`, and `LLaMA-7B`. Each model independently predicts an answer, and the final result is selected via majority voting:

- If at least two models agree, the common answer is selected;
- If all predictions differ, one is randomly chosen.

This approach achieved **54.31** accuracy, outperforming both the single-agent and two-agent pipelines.

c) Why Voting Works::

- **Model Complementarity:** Different models make different types of errors. Combining them allows correct answers from one model to override mistakes from another.
- **Suppression of Outliers:** Rare or hallucinated predictions by one model are filtered out unless corroborated by others.

d) Limitation of voting agents: During case study, we found that GPT-3.5 and deepseek-v3 are more often to have similar answers and same mistakes. Moreover, they both suffer from giving a more detailed answer but lower score. In the future, we may modify the evaluation methods and improve the

D. Comparison for Different Model

To investigate the impact of the underlying language model on the performance of our VRLM framework, we compare three representative backbone models: GPT-3.5, DeepSeek V3, and LLaMA 7B. As shown in Table VII, larger models do not necessarily yield better results unless tailored reasoning capabilities are incorporated.

Despite being the smallest among the three, LLaMA 7B achieves the highest validation accuracy (53.34%), outperforming GPT-3.5 (47.68%) by 5.7 points and DeepSeek V3 (50.62%) by 2.7 points. This indicates that instruction-tuned smaller models can be more effective than larger general-purpose models when equipped with proper prompting strategies and architectural integration. The result highlights the importance of model alignment and domain-specific tuning over raw scale, especially in reasoning-heavy tasks like A-OKVQA. However, LLaMA 7B exhibited limited reasoning capability, while the performance of GPT-3.5 and DeepSeek may have been hindered by their tendency to produce overly detailed or specific answers.

Method	Params (B)	Val Acc(%)
VRLM + GPT-3.5	175	47.58
VRLM + DeepSeek V3	13	50.62
VRLM + LLaMA 7B	7	53.34

TABLE VII: A-OKVQA validation accuracy (%) and model size on VRLM for different base model.

VII. Conclusion & Future Work

In this paper, we present VRLM, a comprehensive prompting framework for visual question answering, which builds upon and substantially enhances the original VCTP architecture. By introducing a unified caption strategy, stronger backbone language models, and multi-agent reasoning schemes, our system improves both accuracy and interpretability on the A-OKVQA benchmark.

We show that replacing the fragmented, confidence-based visual descriptions with merged ground-truth captions significantly boosts the model's ability to understand complex scenes. In addition, we evaluate several state-of-the-art language models—including GPT-3.5, DeepSeek V3, and LLaMA-7B—and demonstrate that reasoning-aligned prompting is more impactful than raw model size. Surprisingly, LLaMA-7B, though smaller, outperforms GPT-3.5 when used with our retrieval-enhanced prompting.

To address reasoning instability and hallucination, we explored two multi-agent variants. The two-agent critique pipeline yielded only marginal improvements due to low disagreement and prompt redundancy. In contrast, our voting-based ensemble of three independent agents led to a new state-of-the-art validation accuracy of 54.31%, outperforming all single-agent baselines. This validates the power of decentralized model consensus in ambiguous visual contexts.

Future Work. Our current framework relies on string-based answer matching against a 10-way human annotation set. This evaluation scheme penalizes semantically correct but lexically rare predictions (e.g., `"mobile phone"` vs.

"phone"), especially for models producing richer or more specific outputs. We plan to introduce semantic-aware metrics (e.g., BERTScore) and relaxed matching thresholds to improve fairness in scoring. Moreover, dynamic prompt tuning and adaptive agent routing—based on image complexity and model confidence—are promising directions to further enhance sample efficiency and robustness.

We believe our findings shed light on the value of integrating prompt engineering, model coordination, and reasoning traceability for future visual language intelligence systems.

Acknowledgment

We are grateful to Prof. Kostic and TAs of course EECS 6691 for instructions during this project.

References

[1] J. Wei, X. Wang, D. Schuurmans, and et al., "Chain of thought prompting elicits reasoning in large language models," *arXiv preprint arXiv:2201.11903*, 2022.

[2] Z. Chen, Q. Zhou, Y. Shen, *et al.*, "Visual chain-of-thought prompting for knowledge-based visual reasoning," in *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence (AAAI-24)*, Association for the Advancement of Artificial Intelligence, 2024. [Online]. Available: https://arxiv.org/abs/2402.07283.

[3] D. Schwenk, A. Khandelwal, C. Clark, M. Yatskar, and Y. Choi, "A-okvqa: A benchmark for visual question answering using world knowledge," in *European Conference on Computer Vision (ECCV)*, Springer Nature Switzerland, 2022, pp. 146–162. [Online]. Available: https://arxiv.org/abs/2206.01718.

[4] T. B. Brown, B. Mann, N. Ryder, *et al.*, "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[5] T. Shin, Y. Razeghi, R. Logan, E. Wallace, and S. Singh, "Autoprompt: Eliciting knowledge from language models with automatically generated prompts," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 4222–4235.

[6] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021, pp. 3045–3059.

[7] X. Liu, K. Ji, Y. Fu, *et al.*, "P-tuning: Prompt tuning can be comparable to fine-tuning across scales," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2021, pp. 61–68.

[8] J. Wei, M. Bosma, V. Zhao, *et al.*, "Finetuned language models are zero-shot learners," in *International Conference on Learning Representations (ICLR)*, 2022.

[9] L. Ouyang, J. Wu, X. Jiang, *et al.*, "Training language models to follow instructions with human feedback," in *Advances in Neural Information Processing Systems*, 2022.

[10] J. Wei, X. Wang, D. Schuurmans, *et al.*, "Chain of thought prompting elicits reasoning in large language models," in *Advances in Neural Information Processing Systems*, 2022.

[11] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Tanaka, "Large language models are zero-shot reasoners," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

[12] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," in *International Conference on Learning Representations (ICLR)*, 2023.

[13] Y. Zhang, Y. Bai, X. Wang, *et al.*, "Automatic chain-of-thought prompting in large language models," in *Advances in Neural Information Processing Systems*, 2022.

[14] P. Anderson, X. He, C. Buehler, *et al.*, "Bottom-up and top-down attention for image captioning and visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 6077–6086.

[15] 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 (EMNLP)*, 2019, pp. 5100–5111.

[16] K. Marino, M. Rastegari, A. Farhadi, and R. Mottaghi, "Ok-vqa: A visual question answering benchmark requiring external knowledge," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 3195–3204.

[17] Y. Jiang, I. Misra, M. Rohrbach, and et al., "Pythia: A modular framework for vision language research," in *NeurIPS Workshop on Visually Grounded Interaction and Language (ViGIL)*, 2018. [Online]. Available: https://arxiv.org/abs/1807.09956.

[18] J. Lu, D. Batra, D. Parikh, and S. Lee, "Vilbert: Pre-training task-agnostic visiolinguistic representations for vision-and-language tasks," in *NeurIPS*, 2019. [Online]. Available: https://arxiv.org/abs/1908.02265.

[19] H. Tan and M. Bansal, "Lxmert: Learning cross-modality encoder representations from transformers," in *EMNLP*, 2019. [Online]. Available: https://arxiv.org/abs/1908.07490.

[20] K. Marino, X. Luo, Y. Yu, D. Parikh, A. Schwing, and T. Darrell, "Krisp: Integrating implicit and symbolic knowledge for open-domain knowledge-based vqa," in *CVPR*, 2021. [Online]. Available: https://arxiv.org/abs/2103.14694.

[21] A. Gupta, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Gpv-2: Scaling and evaluating general purpose vision-

language models," in *CVPR*, 2022. [Online]. Available: https://arxiv.org/abs/2202.02317.

[22] J. Yang, R. Menon, M. Bansal, D. Parikh, and S. Lee, "Pica: Prompting gpt-3 to answer open-ended questions," *arXiv preprint arXiv:2109.05014*, 2022.

TABLE VIII: Individual Student Contributions in Fractions

UNI	**† yk3108**	**† wz2708**	**† hd2573**
Last Name	Ke	Zhu	Dong
Fraction of (useful) total contribution	1/3	1/3	1/3
What I did 1	Topic Choosing & Method Review	Method Review	Topic Choosing & Method Review
What I did 2	Methodology, Experiment Report	Implemention, Expirement Report	Coding and Implementation
What I did 3	DeepSeek, Agents and Ablation Study parts	LLaMA and Base Model Comparison parts	GPT and Caption parts, and Directory Settings
What I did 4	Report Writing of Methodology, Rationale Analysis, Conclusions	Report Writing of Introduction, Implementation, Rationale Analysis, Base Model Comparation	Report Writing of Original Methodology, Results, Caption Ablation

† *All authors contributed equally.*

Appendix

A. Summary of Existing Baselines

This appendix details the baselines compared in Table II used by original VCTP paper, including their core contributions and original references.

- **Pythia (2018)**: A modular re-implementation of the Bottom-Up/Top-Down VQA model that won the VQA Challenge 2018. https://arxiv.org/abs/1807.09956
- **ViLBERT (2019)**: A two-stream Transformer with co-attention for joint vision–language pretraining. https://arxiv.org/abs/1908.02265
- **LXMERT (2019)**: A cross-modal Transformer aligning language and vision via a two-stage encoder. https://arxiv.org/abs/1908.07490
- **KRISP (2021)**: Combines implicit knowledge from BERT with symbolic KBs using graph reasoning. https://arxiv.org/abs/2103.14694
- **GPV-2 (2022)**: A general-purpose vision model trained with webly-supervised concept expansion. https://arxiv.org/abs/2202.02317
- **PICa* (2022)**: A few-shot method using GPT-3 as an implicit KB with caption-based prompting. https://arxiv.org/abs/2109.05014
- **CoT* (2022)**: Chain-of-Thought prompting enables reasoning via intermediate natural language steps. https://arxiv.org/abs/2201.11903