Designing a Chain-of-Thought Enhanced Visual Question Answering System Using GPT-2

Introduction

Visual Question Answering (VQA) is the task of answering open-ended questions based on an image (What is Visual Question Answering? - Hugging Face). A VQA system must interpret visual content and understand a textual question to produce a correct natural language answer. This requires a combination of computer vision and natural language reasoning. Many VQA questions go beyond straightforward image recognition; they may involve logical inference, commonsense reasoning, or multi-step deduction about the image's content. For example, a question might ask, *"If there are no people visible, can this scene be daytime?"*, which requires reasoning about shadows and lighting rather than directly naming an object.

Chain-of-Thought (CoT) prompting is a recent approach to enhance logical reasoning in language models by making them generate a step-by-step reasoning process before giving a final answer ([2201.11903] Chain-of-Thought Prompting Elicits Reasoning in Large Language Models). Instead of responding immediately, the model is prompted to produce a "chain of thought" – a series of intermediate reasoning steps – which ideally leads to a more accurate answer. Prior work has shown that providing or eliciting such reasoning sequences can significantly improve performance on complex tasks in sufficiently large models ([2201.11903] Chain-of-Thought Prompting Elicits Reasoning in Large Language Models). This project aims to leverage CoT prompting to improve the reasoning capabilities of a VQA system. By training a model to generate explanatory reasoning (the chain of thought) followed by the answer, we hope to enable better handling of logically complex questions. Our approach uses GPT-2 as the core language model to generate answers, augmented with chain-of-thought reasoning, due to GPT-2's open availability and manageable size for fine-tuning.

(image) *Figure 1: High-level system architecture of the proposed CoT-enhanced VQA system, from data preparation to evaluation. The VQA dataset is first augmented with chain-of-thought reasoning steps for each question, then tokenized and used to fine-tune a GPT-2 model. The fine-tuned model is evaluated on a test set of image questions – either directly or with CoT prompting – and performance is measured using metrics such as Exact Match, BLEU, and ROUGE.*

Challenges Faced

Building a VQA system with limited resources presented several challenges. In early experimentation, we attempted to use other models beyond GPT-2, such as TinyLlama and Moondream2, but encountered significant limitations in a Google Colab environment:

- **TinyLlama (1.1B parameters)**: TinyLlama is an open-source language model based on LLaMA 2 architecture, valued for its compact size (only 1.1B params) and low memory footprint (TinyLlama/TinyLlama-1.1B-Chat-v1.0 · Hugging Face). We hoped its smaller scale would suit Colab. However, fine-tuning TinyLlama still proved infeasible under Colab's GPU memory constraints. In practice, fully updating a model with over a billion parameters requires substantial VRAM (for example, fine-tuning GPT-2 Large ~774M or models in this range often needs >20 GB GPU memory (Fine-Tuning GPT-2 on a Custom Dataset - GitHub Gist)). Our attempts to train TinyLlama on a 12–16 GB GPU led to out-of-memory errors unless the batch size and sequence length were drastically reduced. We explored 8-bit and 4-bit quantization to compress the model and reduce memory usage. Unfortunately, these quantization attempts introduced compatibility issues (model loading/training errors) and degraded the model's precision. The combination of memory limits and technical hurdles in quantizing TinyLlama forced us to abandon this route.

- **Moondream2**: Moondream2 is a small vision-language model (VLM) designed to run efficiently on edge devices with very little memory (Moondream2: Tiny Visual Language Model For Document Understanding | by Yogendra Sisodia | Medium). It has the appeal of being a multimodal model (able to accept images and questions directly). We considered Moondream2 to incorporate an image encoder for our VQA task. However, we faced fine-tuning restrictions with this model. Moondream2 uses custom functionalities (e.g. methods to encode images and answer questions), and adapting it for end-to-end training on our data in Colab was not straightforward. Its out-of-the-box performance on complex reasoning tasks is quite limited – *"very basic"* in the words of one evaluation, and likely needing task-specific fine-tuning to improve (Moondream2: Tiny Visual Language Model For Document Understanding | by Yogendra Sisodia | Medium). In our attempts, the model did not yield good answers without additional training, and implementing that training posed problems. We again ran into memory issues when trying to finetune Moondream2's 0.5B–2B parameter architecture, and some library dependencies (requiring `trust_remote_code` and custom model classes) made integration difficult. Quantization was also tried here to lighten the model, but similar problems arose. These issues with Moondream2's fine-tuning and its limited reasoning ability precluded its use in our project.

Given these challenges, we decided to use **GPT-2** as the backbone for our VQA system. OpenAI's GPT-2 (released in 2019) is an earlier-generation large language model that was fully open-sourced, with the largest 1.5B parameter version publicly released by November 2019

(GPT-2 - Wikipedia). We chose GPT-2 for several reasons: (1) **Open-source license** – GPT-2's model weights are freely available for research and carry a permissive license, unlike newer models that may have usage restrictions. (2) **Manageable size** – Even the largest GPT-2 (1.5B) is smaller than many modern LLMs, and smaller variants (117M, 345M) can be fine-tuned on a single GPU. This compatibility with full fine-tuning on Colab's hardware was crucial. (3) **Mature tooling** – GPT-2 is supported by Hugging Face Transformers, allowing easy loading, training, and generation. Although GPT-2 is a text-only model (it cannot directly process images), using it allowed us to focus on the *reasoning* aspect (the chain-of-thought logic) by providing textual proxies for images, and ensured we could run the entire pipeline within the limited environment. In summary, after encountering memory and training roadblocks with TinyLlama and Moondream2, GPT-2 emerged as a practical choice to demonstrate our Chain-of-Thought VQA approach.

Training Procedure

Dataset Preparation: We curated a small VQA-style dataset to fine-tune GPT-2. Due to compute limits, only a tiny subset of possible Q&A pairs was used (on the order of only dozens of examples). Each data point consisted of an image-related question, a chain-of-thought reasoning sequence, and the final answer. Since GPT-2 cannot ingest images, we effectively provided the model with *textual* training data: for each question, we wrote a brief step-by-step reasoning as if describing the logical thought process a person would take to answer, and then the correct answer. This chain-of-thought annotation was either manually created or derived from known explanations. For example, a training sample might look like: *Q: "The photo shows a table with an apple and a knife. Why might the knife be there?"* → *CoT: "The presence of an apple suggests someone was about to peel or cut it. A knife on the table would be used to cut the apple."* → *A: "To cut the apple."* By constructing the training targets in this format (reasoning + answer), we aimed to teach GPT-2 to produce the reasoning steps before the answer. We also prepared a few samples without the CoT (direct Q→A) for comparison during evaluation. All text was lowercased and stripped of unnecessary punctuation for consistency, and then tokenized with GPT-2's Byte-Pair Encoding tokenizer.

Fine-Tuning Setup: We loaded the GPT-2 model via Hugging Face Transformers in half-precision (FP16) mode to reduce memory usage. The model was initialized with pre-trained weights (we used the `gpt2` medium checkpoint as a starting point). We then fine-tuned it on our small dataset using the AdamW optimizer (a variant of Adam optimized for weight decay regularization, commonly used for transformer models). Hyperparameters were kept modest: we ran for 3 epochs over the data with a batch size of 2 and a learning rate on the order of 1e-4 (given the tiny data, a relatively higher learning rate helped it learn something within just a few epochs). Gradient accumulation was used for effective batch size when needed, and gradient checkpointing was enabled to manage memory. No complex learning rate schedules or early stopping criteria were employed due to the short training span. The entire fine-tuning took only a few minutes. Throughout training, we monitored the loss, which did decrease on the training data, indicating the model was at least fitting the few examples.

It is important to note that this fine-tuning procedure was constrained by the Colab environment. The GPU available (e.g. an NVIDIA T4 with ~15 GB VRAM) meant we could not increase model size or batch size without running out of memory. Running only 3 epochs was a deliberate choice to avoid lengthy training on limited session time – our goal was to demonstrate the pipeline end-to-end rather than achieve a fully optimized model. In a more realistic setting, one would train for significantly more epochs (especially with more data). Nonetheless, by the end of training, we had a GPT-2 model that had seen a few examples of chain-of-thought reasoning for VQA, and we proceeded to evaluate its performance.

Results

After fine-tuning, we evaluated the GPT-2 model on a small test set of VQA questions. We measured standard evaluation metrics used in QA/VQA tasks, including **Exact Match** (whether the predicted answer exactly matches the ground truth answer) and partial-credit metrics like **BLEU** and **ROUGE** (which measure overlap between the predicted answer and the reference answer). We conducted evaluations in two modes: (1) *Direct,* where the model is prompted with the question alone and must answer directly, and (2) *CoT,* where the model is prompted with the question and a cue to *"think step by step"*, encouraging it to first generate a chain of thought and then the answer. For the CoT mode, we extracted the final answer from the model's generated reasoning+answer sequence before scoring. The average results were as follows:

Metric	Direct	CoT Prompted
Exact Match (Accuracy)	0.0000	0.0000
BLEU	0.000078	0.000426
ROUGE-1	0.002348	0.006029
ROUGE-L	0.002348	0.006029

As the table shows, the model essentially failed to get any answer completely correct – the Exact Match accuracy was 0% in both settings. The BLEU and ROUGE scores are extremely low (nearly 0, far closer to 0 than to 1). For context, a BLEU of 0.0004 is practically zero (a perfect score would be 1.0), and ROUGE scores in the 0.005 range indicate almost no overlap with the reference answers. There was a slight uptick in BLEU and ROUGE when using chain-of-thought prompting (for example, BLEU improved from 0.000078 to 0.000426). This suggests that with the CoT prompt, the model's answers overlapped marginally more with the ground-truth answers than the direct answers did – possibly because the model, when generating a reasoning sentence, occasionally included a few more relevant words. However,

these differences are minuscule in absolute terms. In practical terms, both the direct and CoT outputs were incorrect and often unrelated to the actual answer.

Analysis of the Modest Performance: The very low scores are not surprising given the limitations of the experiment. Firstly, the dataset used for training was extremely small – far too few examples for a model to learn the broad task of visual question answering. The model likely overfitted those few training samples and failed to generalize, or simply didn't have enough signal to learn anything meaningful beyond trivial patterns. Secondly, GPT-2 was used without any actual visual input. Our system did not have an image encoder; the model never "saw" pixels from the images, only whatever textual description we provided in the chain-of-thought. This means the task was essentially reduced to a reasoning problem on text, and for many visual questions, there is no way to infer the answer from text alone. Essentially, the model was guessing. Thirdly, the short training duration (3 epochs) and limited model size further capped the performance. GPT-2 (even the 345M version) is a relatively small model by modern standards, with limited capacity to perform complex reasoning or memorization. In larger-scale studies, chain-of-thought prompting has shown dramatic gains only with very large models (e.g. 100B+ parameters) ([2201.11903] Chain-of-Thought Prompting Elicits Reasoning in Large Language Models) – our use of a small model likely did not reach the threshold where CoT is effective. Moreover, with so little data, the chain-of-thought training might not have properly taught the model to output reasoning; it may have simply confused the model or added to output length without improving accuracy.

In examining some of the model's outputs, we found that in *Direct* mode the model often produced irrelevant answers or generic sentences (likely reflecting GPT-2's pre-training on internet text rather than any understanding of the question). In *CoT* mode, the model did produce a sentence of "reasoning" but it was usually nonsense or unrelated to the question, followed by an answer that was also incorrect. For example, for a question about counting objects, the CoT output might be something like: *"Let's think step by step. There are many objects on the table, perhaps books and bottles. So the answer is: five."* – which bears no relation to the actual image or correct answer. These qualitative observations confirm that the model did not actually acquire visual reasoning ability; it was mostly parroting the format of a chain-of-thought without substance.

It's worth noting that state-of-the-art VQA models achieve **much** higher performance when properly trained. For instance, large vision-language models like BLIP-2, when fine-tuned on the full VQA dataset, can reach around *80% accuracy* on the VQA v2 benchmark. Our results (effectively 0% on a toy test) are orders of magnitude behind – again, due to the toy nature of this demonstration. The takeaway from our results is that merely adding chain-of-thought on a tiny dataset with a small model is not enough to yield useful VQA performance. However, the experiment **does** successfully demonstrate the intended design pipeline (data preparation → CoT augmentation → model fine-tuning → evaluation), which was the primary objective given the resource constraints. The fact that CoT prompting yielded a slight increase in overlap metrics, albeit tiny, suggests that with more training the approach could start to pay off. In the next section, we discuss how we plan to scale up and improve this system to achieve meaningful performance.

Future Work

There are several clear avenues for improving the Chain-of-Thought VQA system in future work. The current results leave much room for enhancement, and we have identified key steps to advance the project:

- **Larger Dataset and More Training**: The most straightforward improvement is to use a much larger training dataset. Instead of a small subset, we would utilize a full VQA dataset (for example, the complete VQAv2 which has ~80k training questions, or the LoRA-VQA dataset mentioned in our literature survey). Training on thousands of diverse Q&A pairs would allow the model to learn a wide range of visual concepts and associated reasoning patterns. Additionally, we would train for more epochs and iterations to ensure the model properly converges. With more data, the model can also better learn to produce coherent chain-of-thought sequences that correlate with correct answers, rather than the essentially random chains we observed. In summary, scaling up data and training time should dramatically improve the model's accuracy from the current near-zero levels.

- **Better Computing Resources (GPU)**: Running on free Colab severely limited our choices. In future, we plan to use more powerful hardware such as NVIDIA V100 or A100 GPUs (e.g., through a cloud service or a research cluster). These GPUs provide significantly more VRAM (16 GB, 40 GB, or even 80 GB) and compute throughput. With a high-memory GPU, we could fine-tune larger models or use bigger batch sizes and sequence lengths. For example, an A100 would allow using the full GPT-2 1.5B model or even larger language models, and enable training with longer CoT sequences without memory issues. More compute also means we could employ better hyperparameter tuning (trying different learning rates, batch sizes) and possibly use mixed-precision or distributed training to speed up convergence. Overall, access to a stronger GPU will help in executing a *full* fine-tuning run that was not possible in Colab, which should yield substantially better results.

- **Stronger Language Models (LLaMA2, Mistral, StableLM, etc.)**: While GPT-2 was convenient, it is an older model with far fewer parameters and less knowledge than modern models. We plan to experiment with more recent open-source large language models. Candidates include Meta AI's **LLaMA 2** (available in sizes like 7B, 13B parameters) and newer models like **Mistral 7B** (released in 2023). These models have demonstrated significantly stronger performance on language understanding tasks. For instance, Mistral 7B is reported to *outperform* even LLaMA2 13B on various benchmarks while being only 7.3B in size (Mistral 7B | Mistral AI), and it is released under the Apache 2.0 license (permissive for usage) (Mistral 7B | Mistral AI). Using a model like LLaMA2 or Mistral would give us a much more capable reasoning engine for VQA. Similarly, Stability AI's **StableLM** models (ranging up to 7B) are another option for an open-source backbone. We anticipate that a 7B+ parameter model, with its greater capacity, would handle the chain-of-thought training far better than GPT-2 did, likely generating more

coherent and relevant reasoning steps. These models could be used in combination with a vision encoder (as discussed next) or even fine-tuned on text-only data to improve their reasoning before adding vision. The only drawback is their size – which leads to the next point about fine-tuning techniques.

- **LoRA / QLoRA for Efficient Fine-Tuning**: Fine-tuning very large models (billions of parameters) directly is computationally expensive and memory-intensive. To address this, we will employ parameter-efficient fine-tuning methods such as **Low-Rank Adaptation (LoRA)** and **Quantized LoRA (QLoRA)**. LoRA involves freezing the pre-trained model weights and learning small low-rank weight deltas, drastically reducing the number of trainable parameters ([2106.09685] LoRA: Low-Rank Adaptation of Large Language Models). This can cut down GPU memory requirements by 3-4× while achieving performance on par with full fine-tuning ([2106.09685] LoRA: Low-Rank Adaptation of Large Language Models). QLoRA goes further by fine-tuning a model that is loaded in 4-bit precision, allowing models as large as 65B to be fine-tuned on a single 48GB GPU ([2305.14314] QLoRA: Efficient Finetuning of Quantized LLMs - arXiv). We can leverage QLoRA to fine-tune a 7B or 13B model on a reasonable GPU (e.g. a 24 GB card) by quantizing it to 4-bit for forward passes and using LoRA adapters for updates. By using LoRA/QLoRA, we maintain the benefits of a large model's knowledge and capacity, without needing multiple GPUs or absurd memory. In our future experiments, we will fine-tune LLaMA2 or Mistral with LoRA on the VQA dataset, possibly also injecting chain-of-thought style reasoning during training. This approach should let us scale up model size and complexity of reasoning significantly, which we expect to yield major improvements in VQA accuracy.

- **Incorporating Vision Encoder (Multimodal Training)**: Finally, to make the system truly effective, we need to give the model actual visual understanding. This was a known limitation of our current pipeline – GPT-2 was blind to images. In future work, we plan to integrate an image encoder so that the model can base its chain-of-thought on real visual features. One path is to use a pre-trained vision transformer or CNN (like ResNet or CLIP ViT) to encode images into embeddings, and then feed those embeddings into the language model (for example, as prefix tokens or via a multimodal transformer design). Another approach is to start with an existing multimodal model such as **BLIP-2** or **LLaVA** that already combine a vision encoder with a language model (LoRA_Design_LAB.docx). We could fine-tune such a model on our chain-of-thought augmented data – effectively teaching it to generate reasoning and answers given an image. In fact, InstructBLIP (an instruction-tuned BLIP-2) has been shown to handle reasoning-type prompts relatively well. Incorporating one of these advanced VLMs with our CoT approach could greatly boost performance. Even using a mid-level approach like feeding image captions or detected object labels (obtained via an external vision system) into GPT-2 as part of the prompt could be explored, to simulate visual input. In summary, adding proper visual input processing is a critical future step to move from a toy text-only system to a genuine VQA system. With a strong vision-language model and the enhancements outlined above, we expect the CoT-enhanced VQA system to reach

competitive accuracy and demonstrate the benefits of explicit reasoning in visual question answering.

Conclusion

In this project, we designed and implemented a prototype VQA system that integrates **chain-of-thought** prompting with a GPT-2 language model. We have detailed the motivation for this approach – namely, improving logical reasoning in visual question answering – and demonstrated a complete pipeline from data preparation through model training to evaluation. Due to the extremely limited scope of the fine-tuning (tiny dataset and compute), the quantitative results were modest: the GPT-2 model fine-tuned on a small CoT-augmented dataset achieved near 0 scores on standard metrics. However, the exercise successfully proved the viability of the *design* and *workflow*. We showed that GPT-2 can be fine-tuned (within Colab's constraints) to produce a reasoning + answer format, and we established evaluation procedures for direct vs. chain-of-thought prompted answering. The slight differences observed between direct and CoT modes hint that with more training, CoT prompting could start to yield improvements even for smaller models.

Crucially, this project has illuminated the path forward to reach a truly effective chain-of-thought VQA system. The lessons learned from the challenges (e.g. model size trade-offs, Colab limitations) have informed a clear plan for scaling up. By using a **much larger training regime** and **more powerful models** – for example, fine-tuning modern 7B+ parameter models with efficient techniques like QLoRA – we anticipate a dramatic jump in performance. With proper image inputs and a state-of-the-art vision-language architecture, the system would no longer be guessing, but actually understanding and reasoning about images. We expect that such a scaled-up system could achieve results on par with contemporary VQA benchmarks, while providing the added benefit of explainability through its chain-of-thought outputs.

In conclusion, the project served as a successful proof-of-concept for designing a Chain-of-Thought enhanced VQA system using GPT-2. While the current implementation is limited in accuracy, it establishes the framework and demonstrates each component of the solution. With the planned future improvements – larger data, better models (e.g. LLaMA2, Mistral), efficient fine-tuning, and integration of vision features – the CoT-enhanced approach has the potential to significantly improve VQA performance. We reaffirm that logical reasoning is an important aspect of VQA, and our work shows a pathway to imbue models with this capability. With more resources and refinement, we are confident that a chain-of-thought driven VQA system can reach competitive accuracy and provide interpretable reasoning for its answers, advancing the state-of-the-art in explainable visual question answering.

References: The concepts and tools discussed are supported by the following: VQA task definition (What is Visual Question Answering? - Hugging Face), Chain-of-Thought prompting and its efficacy in large language models ([2201.11903] Chain-of-Thought Prompting Elicits Reasoning in Large Language Models) ([2201.11903] Chain-of-Thought Prompting Elicits

Reasoning in Large Language Models), descriptions of TinyLlama (TinyLlama/TinyLlama-1.1B-Chat-v1.0 · Hugging Face) and Moondream2 (Moondream2: Tiny Visual Language Model For Document Understanding | by Yogendra Sisodia | Medium) (Moondream2: Tiny Visual Language Model For Document Understanding | by Yogendra Sisodia | Medium) models, the open release of GPT-2 (GPT-2 - Wikipedia), state-of-art VQA accuracy from BLIP-2 models, and advancements like LoRA ([2106.09685] LoRA: Low-Rank Adaptation of Large Language Models) and QLoRA ([2305.14314] QLoRA: Efficient Finetuning of Quantized LLMs - arXiv) for efficient fine-tuning of large models, as well as the performance of the Mistral 7B model (Mistral 7B | Mistral AI) (Mistral 7B | Mistral AI). These sources collectively underpin the methodology and justification for our system's design.