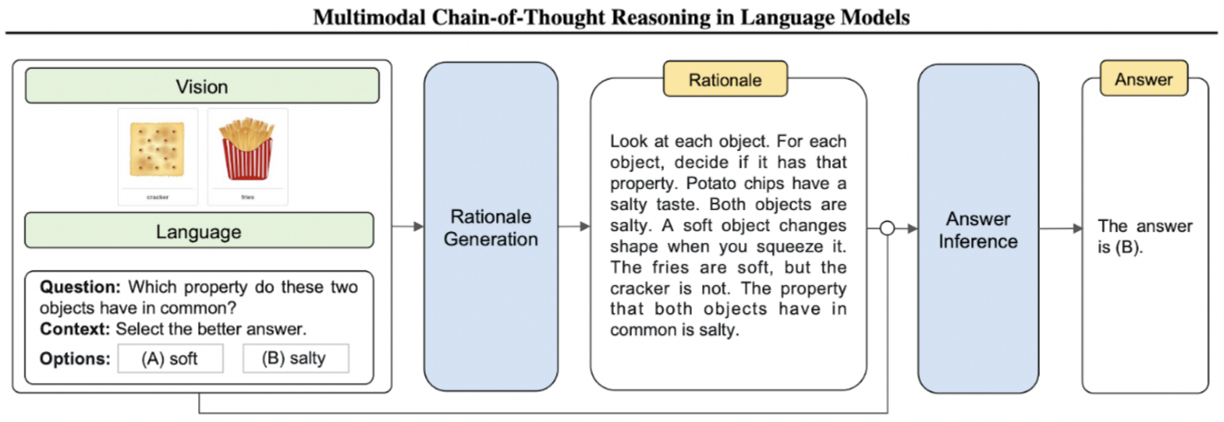
EECS 182 Deep Neural Networks **Spring 2023**

EECS 182 Final Project: Multimodal Chain of Thought (Option 1)

Contributors: Kenneth Wang, Kevin Cai, Qingyuan Liu, Sewon Sohn

Introduction


Multimodal Chain-of-Thought Reasoning in Language Models
Vision
cracker
fries
Language
Question: Which property do these two objects have in common?
Context: Select the better answer.
Options: (A) soft (B) salty
Rationale Generation
Rationale
Look at each object. For each object, decide if it has that property. Potato chips have a salty taste. Both objects are salty. A soft object changes shape when you squeeze it. The fries are soft, but the cracker is not. The property that both objects have in common is salty.
Answer Inference
Answer
The answer is (B).

Chain of Thought Reasoning on ScienceQA (Zhang et al. 2023)

Chain of Thought (CoT) refers to the mental process of reasoning and inference that humans use to arrive at an answer or solution to a problem. It involves synthesizing information from multiple sources, making logical connections between ideas, and integrating them into a coherent line of reasoning. In the context of natural language processing and deep learning models, CoT prompting involves generating intermediate reasoning steps to arrive at the final answer to a question. This process allows the model to break down complex questions into simpler, more manageable steps, and to leverage information from multiple sources, such as text and images, to arrive at a more accurate answer. This homework assignment is heavily adapted from this paper.

This homework assignment will use the Science Question Answering (ScienceQA) dataset , comprising a total of 21,000+ multiple-choice science questions sourced from elementary and high school curricula. Through the attached notebook, you'll explore a subset of this dataset, which consists of questions that only contain a text context as well as questions that have both text and image contexts.

This homework assignment will walk you through the sequential steps towards building a multimodal chain-of-thought model that utilizes both text and image inputs and chain-of-thought reasoning to solve ScienceQA problems.

(a) Prompt Building

Prompt building is commonly used in tasks such as text completion, translation, and question answering, where the model is required to generate output that is consistent with a given input prompt. Prompt building can be a highly effective technique for improving the accuracy and performance of language models, as it allows us to fine-tune the model's behavior for specific tasks and domains. Additionally, prompt building can help to mitigate the problem of bias in language models, as it provides a way to explicitly specify the desired output and constrain the model's behavior.

Run part (a) of the notebook to answer the following questions:

i. This chain-of-thought model uses a two-stage framework. The first stage generates the rationale, which is trained with the "solution" text as the target. Run the block to see what the "solution" corresponds to for the question you saw above. How might this solution help the second model to get the answer?

ii. What are the components of the input prompt to the first stage? In other words, which pieces of a specific datapoint in the ScienceQA dataset do you concatenate together to generate the input prompt? Maintain the order that each component is attached. Hint: Look at build_train_pair.

(b) Add Images

Note that not all questions are associated with an image - the ScienceQA dataset comprises 10,332 (48.7%) questions with an image context, 10,220 (48.2%) with a text context, and 6,532 (30.8%) with both modalities.

i. Run part (b) of the notebook. You can try other indices and see the images as well as the questions they correspond to. To save space, this notebook only downloads certain images (question indices that produce an image are 7, 28, 45, 60). Notice that some indices produce only one image corresponding to the overall question, while other questions also produce images that correspond to the answers. Does adding the picture(s) make the question easier to solve? How might this inform our model?

(c) Dataloading

Dataloading typically involves reading data from one or more sources, such as a file or a database, and performing preprocessing steps such as normalization, transformation, or augmentation, in order to prepare the data for use in the model.This process determines how efficiently and effectively the model can learn from the data.

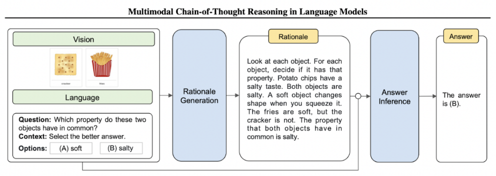
i. Implement part (c) in the notebook. Initialize and tokenize the prompt and the target. Run the "Dataloader Test" code cell to check your answer. Hint: to initialize the prompt and the target, use the given prompt building utils.

(d) Model Architecture

At a high level, the following figure shows the two stage framework of the Multimodal CoT model. The first stage involves the rationale generation, where the input is the text and image data, and the target is the "solution" (which is redefined as rationale) text you saw above. This is the first model. The second model involves answer inference, which takes as input the original text and image data with the corresponding rationale appended, and outputs the multiple choice answer (A, B, C, D, or E). Something to note is the architecture you are implementing (in the notebook, the class is called T5ForMultimodalGeneration) will be used twice, once for the first model (rationale generation) and once for the second model (answer inference) The following figure (Figure 1) shows the end-to-end two stage framework, which was just described in this paragraph, in pseudo-code form. Figure 2 shows a more high-level view of this two stage framework.

```
Input: Language input $X^1_{\text{language}}$, vision input $X_{\text{vision}}$
Output: Generated rationale $R$, inferred answer $A$
1: Construct the input $X = \{X_{\text{language}}, X_{\text{vision}}\}$
2: Generate rationale $R = F(X)$ using the model $F(\cdot)$
3: Append the rationale $R$ to the original language input
   $X^2_{\text{language}} = X^1_{\text{language}} \circ R$.
4: Construct new input $X' = \{X^2_{\text{language}}, X_{\text{vision}}\}$
5: Infer the answer $A$ by conditioning on the new input, $A =$
   $F(X')$.
```

Figure 1. Multimodal CoT high-level pseudocode (Zhang et al. 2023)



Question: Which property do these two objects have in common?
Context: Select the better answer.
Options: (A) soft (B) salty

Look at each object. For each object, decide if it has that property. Potato chips have a salty taste. Both objects are salty. A soft object changes shape when you squeeze it. The fries are soft, but the cracker is not. The property that both objects have in common is salty.

The answer is (B).

Figure 2. Multimodal CoT High-Level Two-Stage Framework (Zhang et al. 2023)

This following figure (Figure 3) shows the model architecture (T5ForMultimodalGeneration) that you'll be implementing in part (d) of the notebook. Remember that F(X) is the model architecture (described in Figure 3 and below), which is used for both the Rationale Generation and the Answer Inference. As such, you're essentially creating two models (which have the same underlying architecture), where the first model (Rationale Generation) produces output that is appended to the original input that is then the new input that goes into the second model (Answer Inference). Below Figure 3 is a textual description of what you should implement for the code. This textual description is also depicted visually (you might want to zoom in to see this better) in Figure 4.

```
procedure F($X$)
    Encode the language and vision inputs $H_{\text{language}}$ and $H_{\text{vision}}$, respectively
    Build the interaction between language and vision features by attention $H^{\text{attn}}_{\text{vision}}$
    Fuse $H_{\text{language}}$ and $H^{\text{attn}}_{\text{vision}}$ by a gated fusion mechanism to have $H_{\text{fuse}}$
    Feed $H_{\text{fuse}}$ to the decoder to obtain the target prediction $Y$
    return $Y$
end procedure
```

Figure 3. Pseudocode of the individual model architecture (both rationale generation and answer inference models use same architecture) (Zhang et al. 2023)

Encoder: By utilizing LanguageEncoder and VisionExtractor functions, the model can take in both language and vision inputs and generate text representation ($H_{language}$) and image feature (H_{vision}). The LanguageEncoder function is implemented using a Transformer model and uses the hidden states of the last layer as the language representation. We use a pre-trained model of T5Stack from HuggingFace (documentation link here) for the LanguageEncoder. On the other hand, the VisionExtractor function vectorized the input image into vision features by extracting patch-level features. In the notebook, this is already done to save space (ie. the raw image data is not imported in, rather the ScienceQA dataset comes with the image features VisionExtractor(X_{Vision})). As such, you do not need to implement this vision extractor step.

$$H_{language} = LanguageEncoder(X_{language})$$
$$H_{vision} = W_h VisionExtractor(X_{vision})$$

Interaction: To correlate text tokens with image patches, we use a single-head attention network after obtaining the language and vision representations. In this network, text representation is used as query (Q) and image feature is used as key (K) and value (V). After this step, gated fusion mechanism is applied, combining the text representation and image feature.

Gated Fusion Mechanism This mechanism combines information from multiple modalities in a controlled way. Each modality is represented by a separate set of features which are combined by weighing the contributions of each modality based on its relevance to the task at hand. It typically consists of a sigmoidal gating function that takes as input a weighted sum of the features from each modality, and outputs a gating vector that controls the contribution of each modality to the final output. The gating vector is then multiplied element-wise with the features from each modality, and the resulting features are summed or concatenated to produce the final output. For example, in a question where an image context is not available, it can be deduced that more weight would be put on the language modality.

$$H^{attn}_{vision} = Softmax(QK^T/\sqrt{d_k})V$$
$$\lambda = Sigmoid(W_l H_{language} + W_v H^{attn}_{vision})$$
$$H_{fuse} = (1-\lambda) * H_{language} + \lambda * H^{attn}_{vision}$$

Decoder: The resulting fused output, H_{fuse}, is then passed through the Transformer decoder to predict the target. We use the pre-trained model of T5Stack to implement the final decoder.

i. The two-stage framework takes the input (text and image), generates rationale, and then appends the rationale to the original input to create a modified input. This modified input is then passed into the second model, the inference model. What architectural structure (covered in the course) is this reminiscent of or analogous to? Hint: If the rationale is empty or zero, what does the modified input devolve to?

ii. Implement part (d) in the notebook. The following figure has been provided to assist you with this:

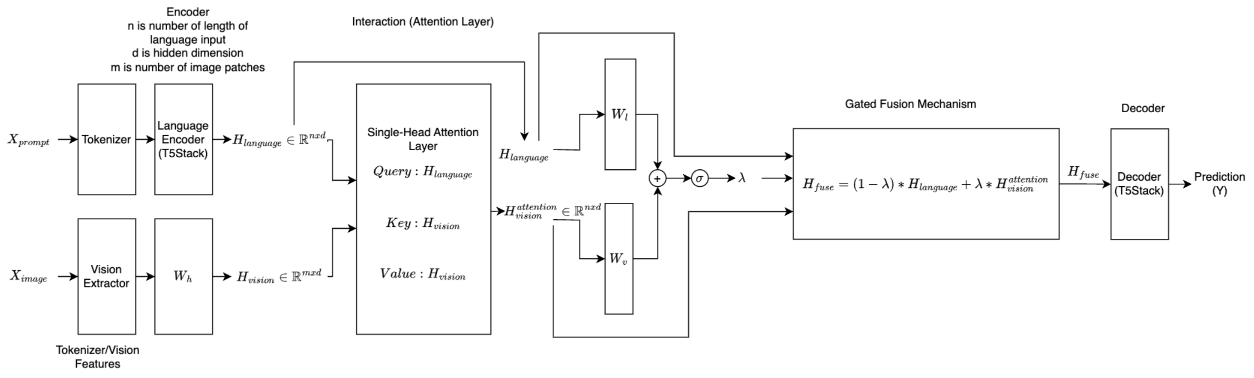

Encoder
n is number of length of language input
d is hidden dimension
m is number of image patches
Interaction (Attention Layer)
X_{prompt}
Tokenizer
Language Encoder (T5Stack)
$H_{language} \in \mathbb{R}^{nxd}$
Single-Head Attention Layer
$Query: H_{language}$
$Key: H_{vision}$
$Value: H_{vision}$
$H_{language}$
W_l
$H_{vision}^{attention} \in \mathbb{R}^{nxd}$
W_v
$+$
σ
λ
Gated Fusion Mechanism
$H_{fuse} = (1-\lambda) * H_{language} + \lambda * H_{vision}^{attention}$
H_{fuse}
Decoder
Decoder (T5Stack)
Prediction (Y)
X_{image}
Vision Extractor
W_h
$H_{vision} \in \mathbb{R}^{mxd}$
Tokenizer/Vision Features

Figure 4. Multimodal CoT – End-to-End Model Architecture Graph

(e) Visualization of rationales on examples

i. What pattern do you see in the rationales generated from the model that hasn't been trained yet? Why do you think this pattern exists?

Run the training cells. This model is loaded with pretrained weights and you will finish running the last two epochs of the model. This should take about 10 minutes.

ii. With your now fully trained model, run inference on some data points! Do the rationales make sense? Do they help answer the question? Compare them to the rationales generated before the training.

(f) Image robustness experimentation and visualization

Congratulations! You (hopefully) have a working multimodal CoT model! In this part of the assignment, you will explore the robustness of the multimodality of this model and see what happens when you feed incorrect data in.

i. Run part (f) of the notebook. In this part, you will experiment on a single example to better visualize and understand how the model you trained may or may not be robust to incorrect inputs. This first block simply displays the example discussed beforehand (the bottom feeder question), along with the text inputs and the image inputs.

ii. What happens when you swap the image of the bottomfeeder that serves as an example for the question (see Figure 5) with a completely unrelated image? What do you notice between the original inference (where the question has the correct image) versus the "wrong" inference (where the question has an incorrect image)? What does this suggest about this model's reliance on the image modality for performance? Write your answer in the text box below.

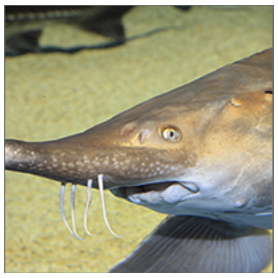


Figure 5. Bottomfeeder) (Zhang et al. 2023)