**EECS 182 Final Project SP2023**

Kenneth Wang, Kevin Cai, Qingyuan Liu, Sewon Sohn

Multimodal Chain of Thought

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](https://arxiv.org/abs/2302.00923).

This homework assignment will use the [Science Question Answering (ScienceQA) dataset](https://github.com/lupantech/ScienceQA), comprising a total of 21,000+ multiple-choice science questions sourced from elementary and high school curricula. Through the [attached notebook](https://colab.research.google.com/drive/19FdymmEqGXbvtW8A5mAN_iPx_IU1VvRm#scrollTo=LqDbUDfSZ6zW), 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?

ANSWER: The solution text should output a guide that walks through the reasoning for how one might get to the answer. For an index of 7 (the bottom feeder question), the solution text first describes the sturgeon and its mouth shape, then it describes looking at the two choices (“discus” and “armored catfish”). Then, the solution text describes how the armored catfish has a mouth on the underside of its head that points downwards (and is thus adapted for bottom feeding) while the discus doesn’t have a mouth on the underside of its head, thus it is not adapted for bottom feeding.

As such, even without the images, the solution provides reasoning (a synonym of reasoning is rationale!) that can guide a reader to the answer. Similarly, the second model may use this reasoning (which is generated by the first model, the rationale generation model) to figure out the correct answer to “Which animal’s mouth is also adapted for bottom feeding?”.

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.*

ANSWER: Question text, newline, Context text (otherwise known as the hint in the ScienceQA dataset), newline, Choices, newline, “Answer: ”

Question, newline, Context, newline, Choices, newline, “Answer: ”

For instance, the datapoint at index 7 (the bottom feeder question) has the following components:

'question': "Which animal's mouth is also adapted for bottom feeding?"

'hint': "Sturgeons eat invertebrates, plants, and small fish. They are bottom feeders. Bottom feeders find their food at the bottom of rivers, lakes, and the ocean.\nThe 's mouth is located on the underside of its head and points downward. Its mouth is adapted for bottom feeding.\nFigure: sturgeon."

'choices': ['discus', 'armored catfish']

Thus, the prompt built would look something like this:

Question: Which animal's mouth is also adapted for bottom feeding?\nContext: Sturgeons eat invertebrates, plants, and small fish. They are bottom feeders. Bottom feeders find their food at the bottom of rivers, lakes, and the ocean.\nThe 's mouth is located on the underside of its head and points downward. Its mouth is adapted for bottom feeding.\nFigure: sturgeon.\nOptions: (A) discus (B) armored catfish\nAnswer:

Note: If the problem has a corresponding image, then the context text can also append a text caption that describes the image. However, there is very minimal empirical difference, so this notebook leaves out the caption text.

Additional note: Depending on the question being evaluated, the model might also append the ‘lecture’ text with the ‘solution’ text when creating the target for the rationale generation model. This doesn’t affect the prompt building (for the input).

**(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?

ANSWER: Yes. For question 7 as an example, the question with the text itself is not intuitive, especially for people that are not familiar with fish names. However, the image added provided visual cues that were not available in the text of the question alone. It allowed for a more educated guess of the answer according to the physical appearances of the fish. Similarly, incorporating pictures into the model can improve its performance and accuracy, by allowing it to learn from both textual and visual cues. For example, a multimodal model might use a combination of convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to process both the image and text data, and then use attention mechanisms to integrate the two modalities and generate an answer.

Extra: Something to note is that although multi-modality seems to help improve model performance on the ScienceQA benchmark (as well as many other similar reasoning benchmarks), it’s not clear that multi-modality always improves performance, or why it seems to improve performance (beyond the intuitive idea that adding multi-modality helps human reasoning). For instance, this paper (Wu et. al. 2021, [linked here](https://aclanthology.org/2021.acl-long.480.pdf)) finds that some multi-modal models actually learn to *ignore* the multimodal information. They argue that multi-modality is akin to regularization, and thus can improve performance in the same way that regularization can.

**(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.

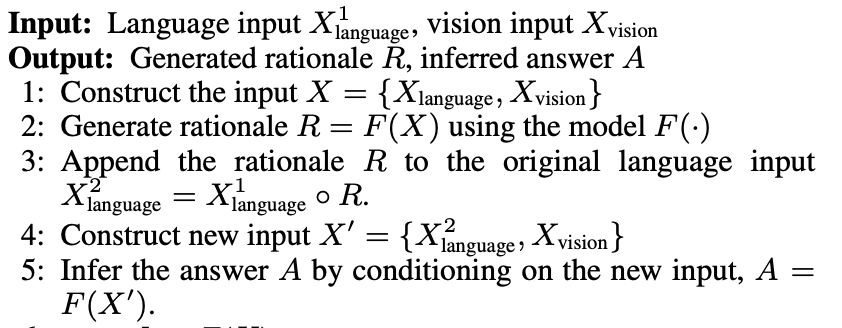
i. Implement part (c) in the notebook. Initialize and tokenize the prompt and the target. Run the “Dataloader Test” code cell to check your answer.

*Hint: to initialize the prompt and the target, use the given prompt building utils.*

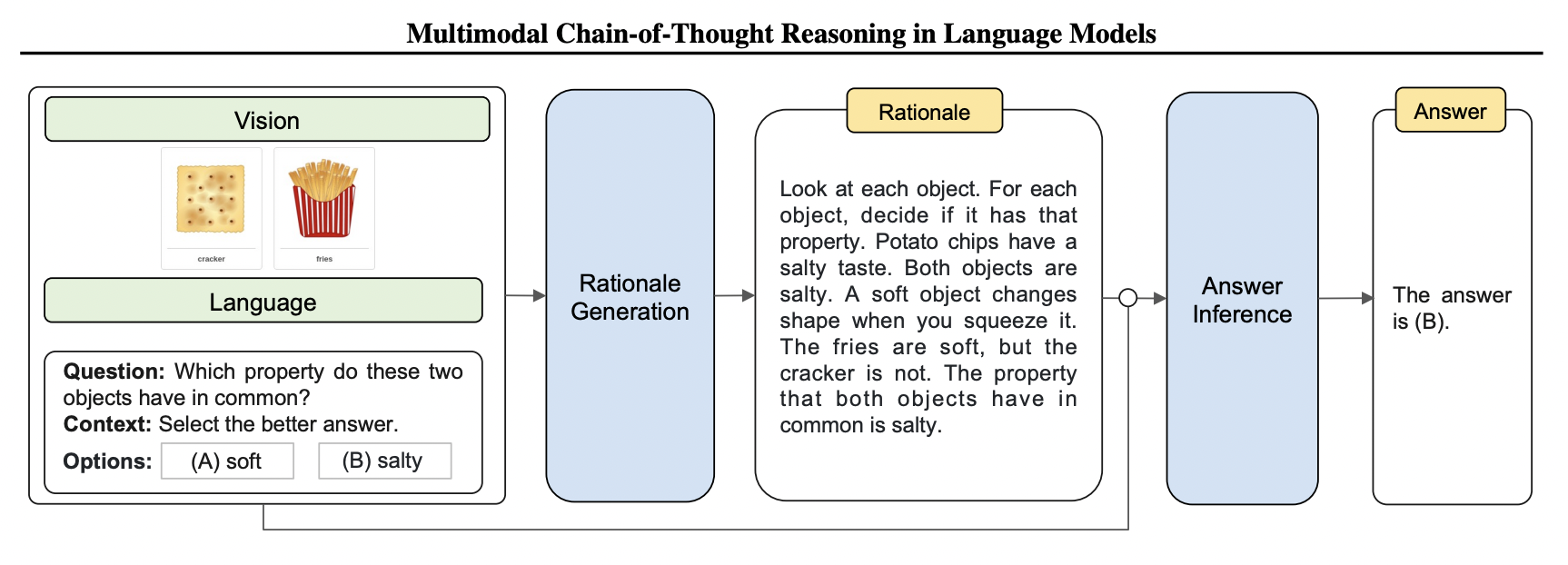
ANSWER: See notebook.

**(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.

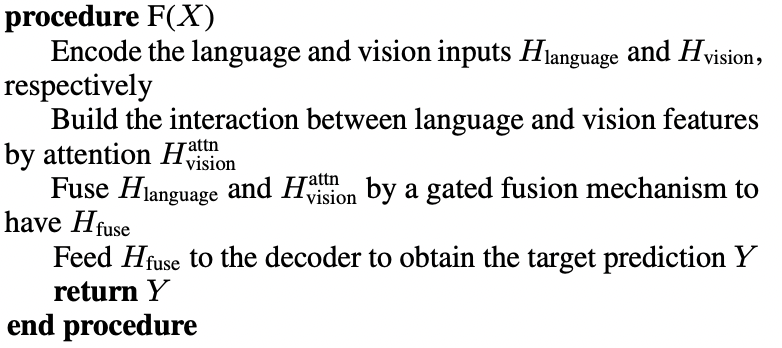


*Figure 1. Multimodal CoT high-level pseudocode (*[*Zhang et al. 2023*](https://arxiv.org/pdf/2302.00923.pdf)*)*



*Figure 2. Multimodal CoT (*[*Zhang et al. 2023*](https://arxiv.org/pdf/2302.00923.pdf)*)*

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.



*Figure 3. Pseudocode of the model architecture* ([Zhang et al. 2023](https://arxiv.org/pdf/2302.00923.pdf))

**Encoder:**

By utilizing LanguageEncoder and VisionExtractor functions, the model can take in both language and vision inputs and generate text representation (Hlanguage) and image feature (Hvision). 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](https://huggingface.co/docs/transformers/model_doc/t5)) 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(XVision)). As such, you do not need to implement this vision extractor step.

**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 a query (Q) and image feature is used as key (K) and value (V). After this step, a 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.

)

**Decoder:**

The resulting fused output, , 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?*

ANSWER: ResNet skip connections.

ii. Implement part (d) in the notebook.

ANSWER: See notebook.

**(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?

ANSWER: Some of the rationales have “zero-padding”, ie. repeated words that fill the last part of the rationale. This is because the text input is zero-padded.

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.

ANSWER: The rationale makes sense now and they provide additional context for answering the questions. The rationales generated after training do not have the repeated words that we could see before the training.

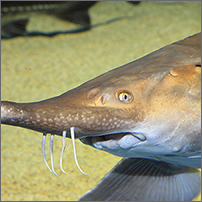
**(f) Image robustness experimentation and visualization**

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 4) with a completely unrelated image?

Figure 4. Bottomfeeder Barry



ii. In this part,

ANSWER:

**(g) Error analysis of rationale generation**

A natural question that may arise: what happens when the rationale generation model outputs “incorrect” rationale for a given datapoint? This part of the assignment will explore some of those cases and see how the model may or may not be robust to incorrect rationale.

Run part (g) of the notebook and answer the following questions below.

i. Does the final model prediction (A, B, C, D, E) always match up with the answer suggested by the rationale? What does this suggest about the model’s weights, the two-stage framework, and the inference model’s overall robustness?

ii. What part of the model architecture can enable the model to be robust to incorrect rationale generated? *HINT: How did you answer part (d)?*

ANSWER: The ResNet skip connection, or the feeding of the original inputs back into the answer inference model along with the rationales generated. If the model didn’t have this connection, then the overall model would likely be less robust to incorrect rationales.

ANSWER:

The inference model learns to ignore some of the rationales, suggesting some parts of the learned weights are dedicated towards ignoring the rationales.

In other words, the inference model is somewhat robust to incorrect rationales. This seems to suggest that in some cases, the chain of thought is ignored.

* + 1. Question: Given that incorrect rationales tend to (but not always) result in incorrect final predictions, what is one direction of improvement for the model?
       1. ANSWER: Add some filtering mechanism on the rationale generation such that “bad” rationales are ignored.

(h) Ablation studies

i. Run evaluation metrics on model with vision and without vision

ii. Question: Which model performs better? Why does one model seem to perform better (intuition)?

SAVE FOR THE FINAL SUBMISSION

(f) Side-by-side comparison of the rationales generated

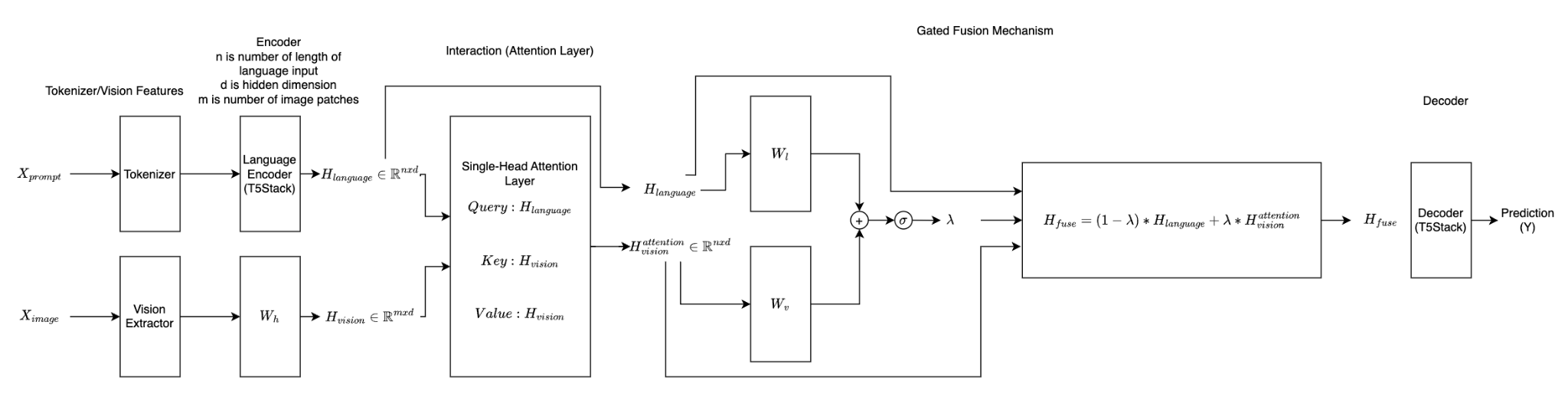
* 1. Attach the side-by-side comparison to your submission.

(g) Error analysis

* 1. Showing cases where rationale generated was incorrect
     1. Question: Does the final model prediction (A, B, C, D, E) always match up with the answer suggested by the rationale? What does this suggest about the model’s weights, the two-stage framework, and the inference model’s overall robustness?
        1. ANSWER: The inference model learns to ignore some of the rationales, suggesting some parts of the learned weights are dedicated towards ignoring the rationales.
        2. In other words, the inference model is somewhat robust to incorrect rationales. This seems to suggest that in some cases, the chain of thought is ignored.
     2. Question: Given that incorrect rationales tend to (but not always) result in incorrect final predictions, what is one direction of improvement for the model?
        1. ANSWER: Add some filtering mechanism on the rationale generation such that “bad” rationales are ignored.

(h) Ablation studies

* 1. Run evaluation metrics on model with vision and without vision
  2. Question: Which model performs better? Why does one model seem to perform better (intuition)?



EECS 182: Multi-Modal Chain-of-Thought Reasoning Model Homework Assignment (Option 1): Commentary for Initial Submission

Key Ideas:

-> we slowly ease them through sequentially

* Multimodality is a natural step and improves language model due to adding more information
  + Part b) Goes through this
* Managing multimodality is difficult
  + Model does this through having cross attention between text (queries) to image patches (keys and values)
    - Selective attention: correlates text with image patches
    - They implement this and that’s how they learn about how to deal with multi modal
  + Use [gated fusion mechanism](https://openreview.net/pdf?id=Byl8hhNYPS)  to merge the vision and text model together
    - The matrices are used to adjust the
  + Multimodality also as regularizer?
    - Not intuitive, briefly touch on this in regularization
* Chain of Thought Reasoning also help improves
  + Two-stage framework acts as a form of inductive bias, particularly suited for reasoning tasks such as ScienceQA
    - Having the two stage framework serves as a inductive bias as to what the right way to reason is, as opposed to just having a bigger model and assuming the model will learn the best way on its own

Some other smaller ideas highlighted:

* ResNet Skip connection idea used in the chain of thought model
* Encoder-Decoder architecture
  + We explain and siomplify the model, then make them actually code it up in part d)
  + Fine tuning / pre-training
    - They use pre-trained T5Stack. We ask thema question on it? ??
  + Kind-of. They engage with in the code lmao
* Tokenizing
  + What does tokenizing actually do?
    - We show them the before and after of tokenizing (before is the question prompt)
    - TODO: would be to go through and add more information / scaffolding on exactly what tokenizing (ie. explain byte-pair encoding, something a scribe note mentioned)
  + Part c)
* Prompt building
  + Part a) exercise to show how prompts get built.
    - Basically part of the sequential showing of what actually happens to the data through each step of the language model for a complex, transformer based model that has multi parts

TODOS:

1. ~~Fix notebook code to make sure it runs consistently~~
   1. ~~No RAM crashing~~
   2. ~~Dependencies solved~~
   3. ~~GDrive solved~~
      1. ~~We have to use GDown because the dataset is too big to download~~
2. Clean notebook titles, variable names
3. Add more comment scaffolding around the coding questions
4. Add computational graph of the architecture
   1. Add the derivative question of the loss function
5. Explain fusion of language and vision (basically explain the fusion gate issue)
6. Address reviewer comments and do point-by-point response (write this out)
7. Add parts f (image robustness experimentation), g (error analysis), and h (ablation studies)
   1. Add to notebook
   2. Add to written PDF
   3. Add to commentary (append)
8. Import to LaTeX
9. Write abstract
10. Fully write out each part of this PDF
    1. Importantly, need to write out the intro context - **Sewon**
    2. Move this into a LaTeX file - **Kenneth**
11. Transfer weights over - **Kevin**
12. On notebook, code up the second part of the model - **Qingyuan**
    1. If time, add a part to the error analysis portion (ie. show an example where the model ignored the rationale, ie. was robust to incorrect rationales)
13. Clean up notebook
    1. Add context and mask out coding parts - **Sewon**
    2. Test fresh (QA) before submission - **Kevin**
    3. Make a copy of the notebooks - **Sewon**
       1. One is solution
       2. One is the blank one
14. Short 2-3 page commentary on the HW that explains the key concepts in the paper(s) and how the HW engages with them in a systematic way - **Kenneth**
15. Extra (after the peer review):
    1. Ablation studies
    2. Error analysis parts

RATIONALE GENERATION model

1. NO TRAIN< SEE HOW SHIT IT IS
2. Train last 2 epochs on the pre-trained

ANSWER INFERENCE model

1. Train last 2 epochs on the pre-trained model.