Viveka 2.0

Project Member Application | AI Club, CFI, IIT Madras

Instructions

- The app is quite long, but we've focused on breaking down each question into smaller pieces that you can spend time on, think and piece together to get the final picture. Ideally there is also a learning outcome associated with each question that we hope will get through to you[a]. The philosophy is to point towards a certain direction, help you discover something interesting about the concept/question, you come up with an explanation, and in the process of which, you end up learning the concept.
- You are allowed to use any inanimate power on earth/ universe to solve the app, including LLMs. If you happen to use LLMs[b], cite any conversation/ interaction along with resources.
- For everything you answer, cite references you used. This is absolutely critical and not doing it will not be taken lightly.
- We will prefer people who do DFS on the questions and complete them in-depth than doing BFS and attending all the questions.
- Submit the application in format "cfi-ai-club-project-viveka-2.0-name-roll-number.pdf" and add your grade card as "roll-number-grade-card.pdf" in the submission link.
- Submit here by **3rd June EOD**.
- Refer to FAQs and Starting resources here. If you feel any question to be included in FAQs or have a doubt in resources/ FAQs, feel free to comment in the doc as well.
- For any queries, feel free to contact-
  - Saahil Faraaz Shaikh ee24b057@smail.iitm.ac.in, +91 7619627660
  - Pakshal Nagda da24b045@smail.iitm.ac.in, +91 9819008528
  - Smitali Bhandari ce24b119@smail.iitm.ac.in, +91 8767441611

[a] The app in itself is open ended. You might find something interesting which is blog-able. Regardless of whether or not you are selected, if you think you have interesting observations, we would highly encourage you to blog them out. Feel free to reach out to us if needed.

[b] No one will not use lol

Before you start, take Sriram's blessings to do well in app (and acads next year) :)

Cook well, all the best

1. General Questionnaire

1.1. Essentials

1. Tell us something about yourself. (Be informal, be creative.)

We encourage you to read up articles/ listen to good resources to articulate your answers with more maturity. Explain in your own words, without necessarily relying on formal definitions.

2. What is AI Safety? Why is it needed?
3. What is your definition of Artificial General Intelligence?
4. Name 3 of your favourite articles, blog posts or videos on ARC AGI or similar. What did you like the most about each?

1.2. Project Goals and Expectation

1. Describe the goals of the project in as much detail as possible. What do you expect working on it to look like? (For any clarification, you may ask in the aspiring Project Members' group.)
2. Imagine yourself a year later, looking back at the work you did. What achievements or progress would make you feel that the experience was a tremendous success?
3. Considering your definition of tremendous success, how much effort do you think it would take to reach that level?

1.3. Motivation and Drive

1. What is your primary motivation for joining this project? What excites you the most?
2. Is there another opportunity that is more aligned with this motivation than Viveka 2.0 PM?
3. What could dampen your enthusiasm or make you disengage?
4. This project would require you to put in a lot of work. As always there is a tradeoff between exploring multiple domains in a shallow manner vs doing one in great detail. Considering that you are in early UG, do you think this would a be a good opportunity for you? Are you sure the positives outweigh the challenges?

1.4. Commitments/PoRs

1. What other commitments are you planning to take on/have for this academic year? Clearly outline the peaks of both the Project Member tenure and the PoR, and explain how you will prevent clashes between both. In case they do clash, how will you prioritize your time?

2. Technical Questionnaire

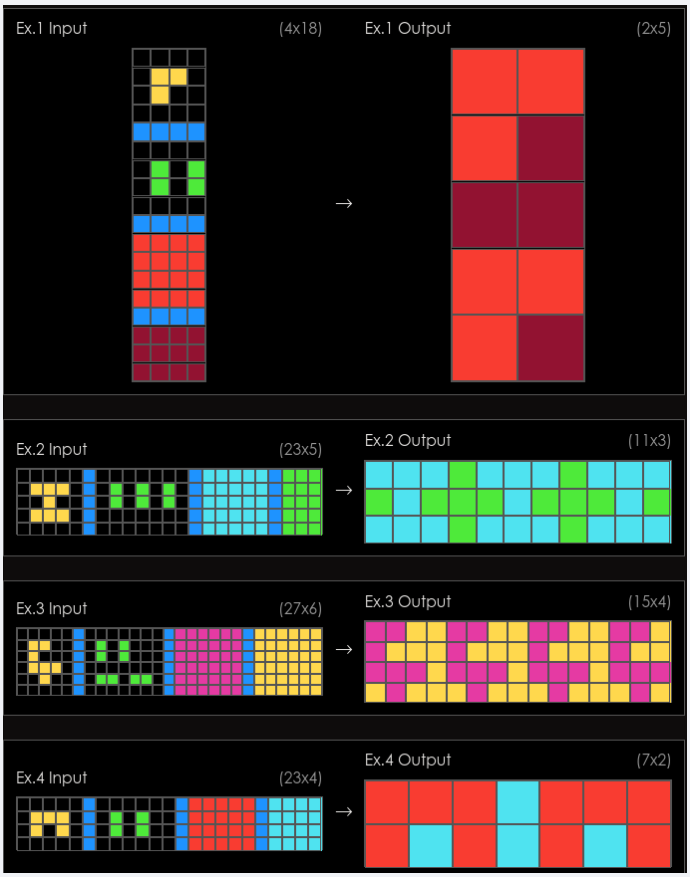
2.1. Puzzle Time!

Recently LLMs solved IMO problems, disproved conjecture in discrete geometry. But are they capable of fluid intelligence? The Abstract Reasoning Corpus - Artificial General Intelligence (ARC-AGI) is a benchmark which tests this.

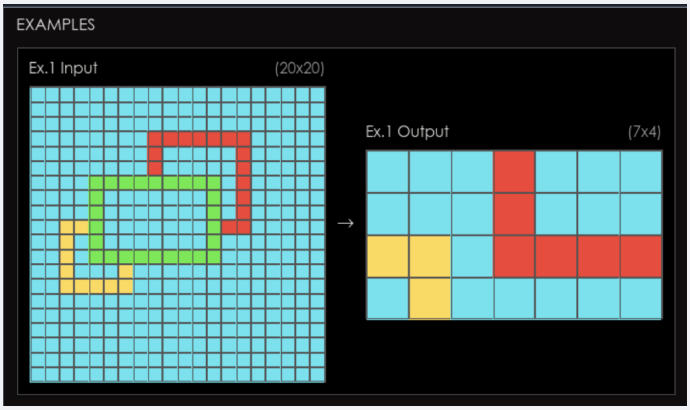
Nearly every frontier model, except for GPT 5.2 pro high - which required ~ 13 *USD/ task, cannot solve these problems.*

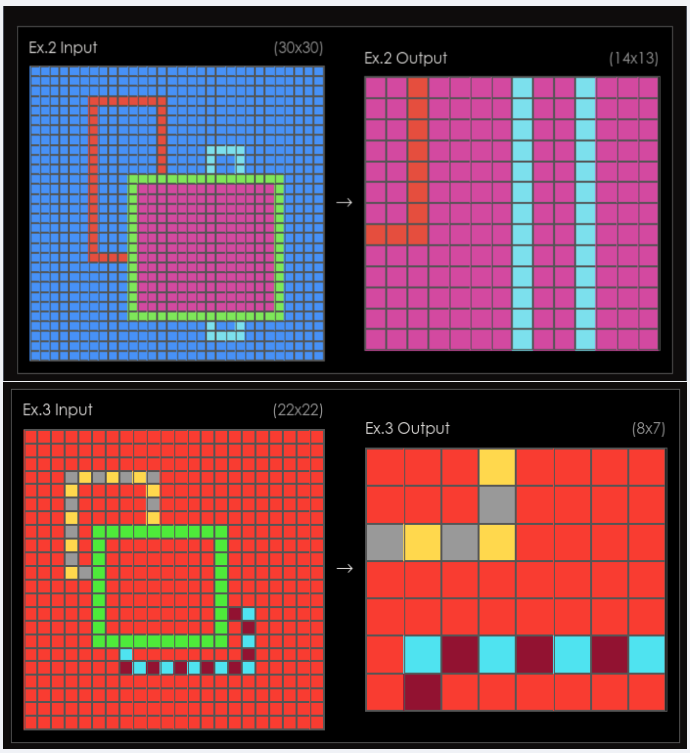
Write a Python script to solve both questions - it should identify which problem the given test case falls under and correctly solve it.

Problem 1:



Problem 2:



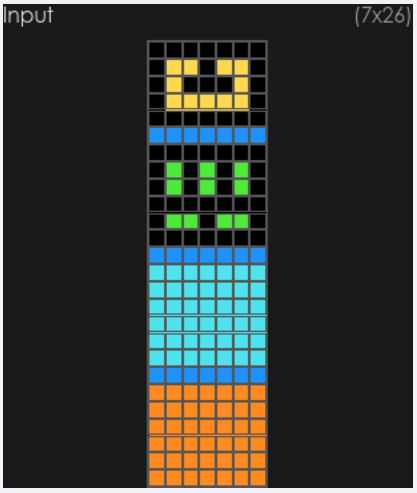


Constraints:

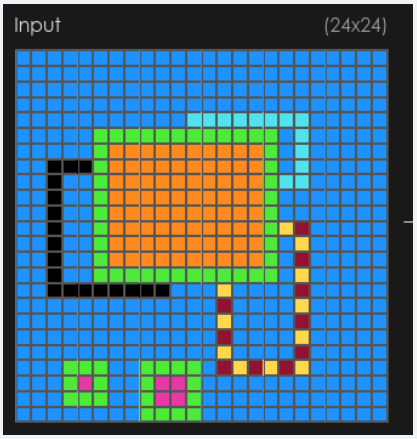
1. Take input and output as a JSON - the same format as the ARC AGI dataset (Hint: Check the GitHub repository)
2. Keep the solution readable.

Test cases:

Problem 1



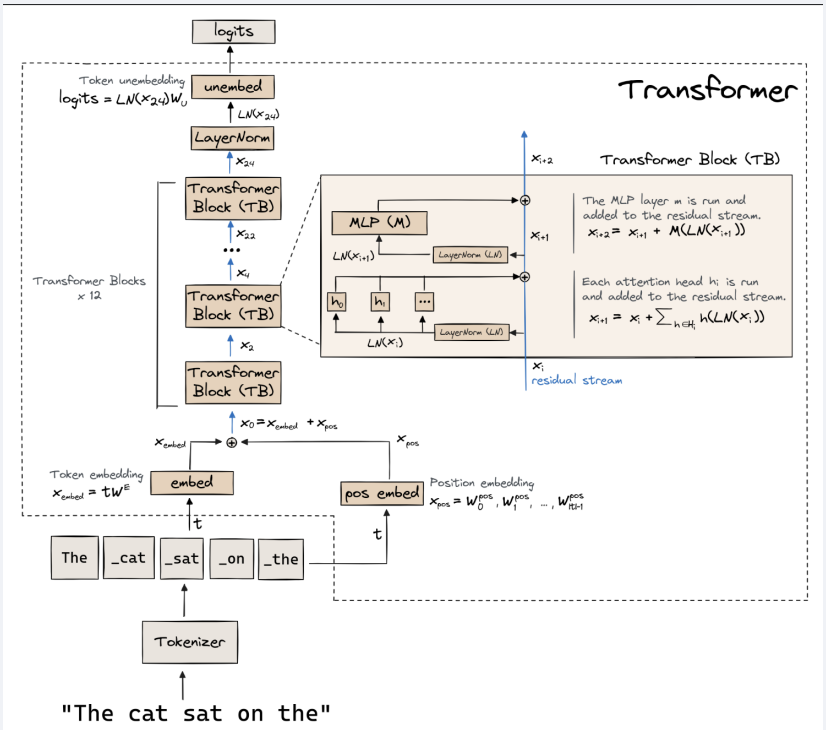
Problem 2



Every LLM that you use, at its heart, has a transformer. *Most* approaches used to solve ARC tasks so far are either LLM based or small model approaches which use transformer blocks with smart tricks. Hence, it is important to understand what transformers are.

2.2. Transformers or Power Rangers?

▶ Transfomers



1. What are the components of transformer blocks? What is the purpose of each part?
2. Explain how attention works. Why does attention exist? What does it do?
3. What is a computational complexity class? Which class does a transformer belong to? What does this mean for transformers?
4. Then how were the LLMs able to solve ARC tasks? What is the common inference time trick used? List the major shortcomings of this trick.

▶ Sudoku! (part 1)

Use this starter notebook to train transformer on Sudoku (use this dataset).

1. "Transformer depth allows the model to capture increasingly complex, hierarchical and abstract representations."
   (a) In context of language models, how is a one layer transformer different to a two layer/ multi layer transformer. Research enough and remark cleanly relevant to the context.
   (b) Explore depth effect for Sudoku transformers.[a]
2. A transformer is typically used for sequence modeling.
   (a) Learn about Positional embeddings. What changes can you make in this module of transformer to make it more apt for Sudoku? [b]
   (b) What is autoregressive generation? Does it make sense to use this for Sudoku transformer? If not, suggest some alternatives and train them out. What happens?

3. *Bonus- FFT –*
   - What are some differences in Sudoku and ARC tasks? Do they have any constraints in common? *Can you take advantage of this in the ARC AGI/Sudoku task? Cite resources (if any) which utilized smart tricks as in 2(a), 2(b) on ARC tasks.*

[a] Train multiple models to be able to make conclusive remarks on how does transformer depth(number of layers), width(number of neurons per layer) affect Sudoku transformer accuracy. Justify your experiment set-up. Are you sure there is no other factor but relative depth/width that influence your results. Do the results make sense? Why or why not?

[b] Train it out. What was the impact? Was it expected, why/ why not? Hint: think of sequence v/s grid

2.3. How to make them reason? - Part 1

Using "thinking", transformers are capable of solving a lot of different kinds of tasks.

▶ Change the iteration

Instead of autoregressively making them reason using tokens, we iterate on the "representations". What if we kept the same set of token representations and passed them through the same transformer block multiple times, updating the representations in place.

1. Understand and explain Looped transformers. How is this different from simply adding more layers?
2. From what you understood about transformer depth- what assumption must break for the Looped transformer to work? What does the Looped transformer implicitly assume about what a "good reasoning step" looks like?

2.4. Part 2- Something still not so much like brain

You now have some understanding of how transformers and looped transformers work, is this similar to how you believe your brain works?

1. Think of all the possible ways, your brain thinks differently than what any of the architectures so far covered optimally do. List all the key structural differences.
2. A hierarchy of intrinsic timescales across primate cortex[a] talks about how brain organizes computation hierarchically across cortical regions operating at different timescales, enabling deep, multi-stage reasoning. Explain this idea with some examples in (daily life) your choice.
3. Hierarchical Reasoning Model consists of two recurrent modules. Understand and explain the architecture and inference of HRMs. Also, explain using an example, take an ARC task.[b] Write mathematically what is the input to the HRM. What is the output. Give details such as dimensions of the embeddings, etc. (You are obviously not at all expected to hand write the vectors!)
4. Why is the H-module's update necessary in L-module's update? What would otherwise the architecture be expected to behave like?

▶ Some Reflection

Vanilla HRM reasons entirely in latent space - no natural language intermediate steps. The final answer is read off from the H-module's hidden state after all the steps.

1. What are the advantages of latent reasoning over token-level reasoning?
2. What does latent reasoning give up?
3. Given these trade-offs, do you think latent reasoning and natural language are competing paradigms?

[a] You are not at all expected to read this paper _/_. Just adding for motivation.

[b] You might find the HRM paper and the Github repository useful.

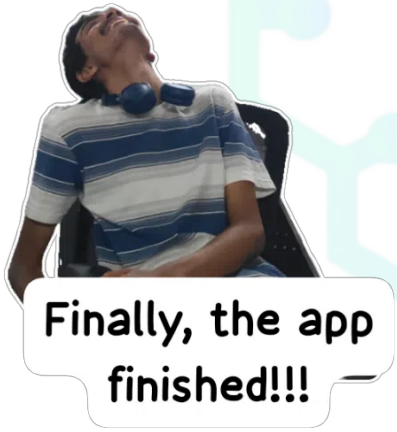
2.5. Inside Out 3! – Sudoku Interp - Part 2

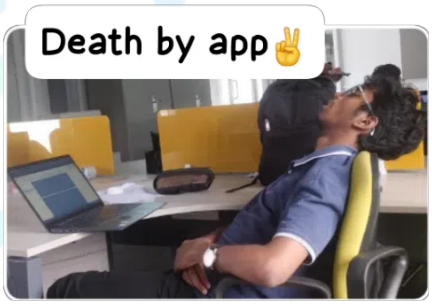
Pick the best Sudoku transformer you trained, let us look inside it.[a]

1. You might have noticed that the architecture in the starter notebook isn't the same as the Transformer architecture you saw in 2.2.1. What are the structural differences?
2. Examine the behaviour of the output probability distribution of the transformer at each position in the grid, do you see any patterns?[b] Explain what you see.
3. What is Logit Lens? Use this after various transformer blocks and sub-blocks in the residual stream to get heatmaps as in previous question. Which block is doing the heavy lifting?[c] Does it make sense for this block to do so, justify.
4. Has the transformer actually learned Sudoku rules? Based on your answer from previous question, look at that particular module.[d]

[a] Starter code is in the Sudoku starter notebook itself.
[b] You can use metrics like Entropy, KL divergence with the ground truth
[c] *Bonus*– Try a different architecture based on this result to verify your claim.
[d] Giving away partial answer- look at attention heat maps.

All the best, enjoy yall!





After this app, I can do anything