Fardad Hajirostami
GTID: 903011588

Project 2: Solving 3x3 RPM Problems

Previous Project Overview:

In the previous project, my agent was responsible for solving strictly 2x2 Raven's Progressive Matrices Problems. The agent was quite successful in generating correct solutions by solely relying on verbal representations. Specifically, the agent mapped semantic network relationships based on the transformations that took place from A -> B and A -> C and then used these mappings to generate 2 solutions that would later be tested against the 6 provided candidates in order to find the candidate with the highest confidence score.

Changes in Problem Set:

For the current project, my agent was given the task of handling 3x3 RPM problems. Additionally, the number of total candidates to choose from increased by 2 from last time. Due to the added complexity of the problem set, I decided that verbal representations alone weren't going to suffice. In order to increase the accuracy of my agent when it came to tackling 3x3 RPMs, I decided to incorporate visual representations of transformations along with previous methodologies of verbal semantic network mappings and testing of generated solutions. While the general approach for finding semantic network relationships remained the same, there were several tweaks to the input feeds and to the generating of solutions in order to accommodate to the nature of 3x3 RPM problems. At a high level, pattern relationships of 3x3 RPM problems can be boiled down in the following ways:

Two Horizontals: (A -> B -> C) and (D -> E -> F)
Two Verticals: (A -> D -> G) and (B -> E -> H)
One Diagonal: (A -> E)

To allow for backward compatibility of my agent's previous utilization of verbal representations, I had to find a way to translate the 2-step transformations of the verticals and horizontals into one-to-one mappings. When processing the semantic webs of row 1 for example, the agent first creates two one-to-one mappings: (A -> C) and (B -> C). These mappings are later translated on the G and H respectively in the last row to give us two generated solutions. The confidence of these two generated solution is then calculated based on their similarities and then later tested against the possible candidates. This process is repeated for the remaining 2-step transformations whenever the agent is prompted utilize verbal representations.

Agent's Reasoning and Relation to Human Cognition:

The main motivation behind my agent's design in solving 3x3 RPM's originates from parallels that can be made in the cognitive processes of humans when it comes to tackling these more

challenging problems. One specific cognitive approach that we have discussed frequently in class is the top-down approach of problem solving. This approach allows us to first attack a problem at the highest level of abstraction. If a convincing solution is not found, we can work our way down the levels of abstraction to extract more detail. This process is repeated until we arrive a conclusive solution that is supported by a substantial amount of evidence. Furthermore, the power of top-down approach was re-emphasized in our discussions regarding the properties of Frames and their rich representative powers in allowing for advanced-sense making. In our discussion regarding the cognitive connection of Frames, we have seen how frames "help to make sense of data. It has been generating expectations of the world. So then the processing becomes not just bottom-up, but also top-down" (P. 91, Goel, Ashok; Joyner,David, 2017) By utilizing a top-down approach, not only are we effectively giving our agent more human like qualities, but also making our agent more powerful as a result. By giving our agent the capability to work in different layers of abstraction, it is able to avoid doing extraneous work and only extrapolate more details about its world after intelligently evaluating whether or not the current information it has about its world suffices.

Agent Design:

To implement the top-down approach of problem-solving that we discussed earlier, my agent first looks at different visual transformations that take place horizontally and vertically. These transformations include addition (3^{rd} image is the sum of the pixels of the first and second image), deletion (3^{rd} image is the absolute difference between the 2^{nd} and 1^{st} image), incrementation (3^{rd} image is the sum of the pixels of the 2^{nd} image and the difference between the 2^{nd} and the 1^{st} image), and static. The confidence of these transformations is calculated by calculating the ratios of the proposed pixel transformations and the actual pixels of the 3^{rd} image.

After the confidence of the pixel transformations for each row and column is calculated, we look and see whether or not they are worthy of being tested against the candidates by thresholding each vertical and horizontal pixel transformation confidence. If there are no pixel transformations with a high enough confidence, the agent moves on to evaluating the verbal transformations. If the confidence of a certain pixel transformation is high enough, the transformation is mapped on the either on the last row, last column, or both to generate the size of the pixels of the proposed solution image.

Next, the agent runs the generated pixel size against the pixel sizes of the candidates and calculates the confidence of the candidates by evaluating the pixel ratios. This process is repeated for remaining transformations with a high enough confidence and the confidence of each candidate is cumulatively multiplied. After mapping and analyzing all the solutions, if the final confidence of one of the candidates is significantly higher than the rest, then the agent returns the solution candidate.

If there are 2 or more candidates with high levels of confidence, the agent must then look for further evidence by analyzing the verbal representations of our RPM. To get a better visual

understanding of the agent's cognitive processes, figure 1 delineates the decision flows of my agent at each step.

AI Agent Decision Flowchart for solving RPMs

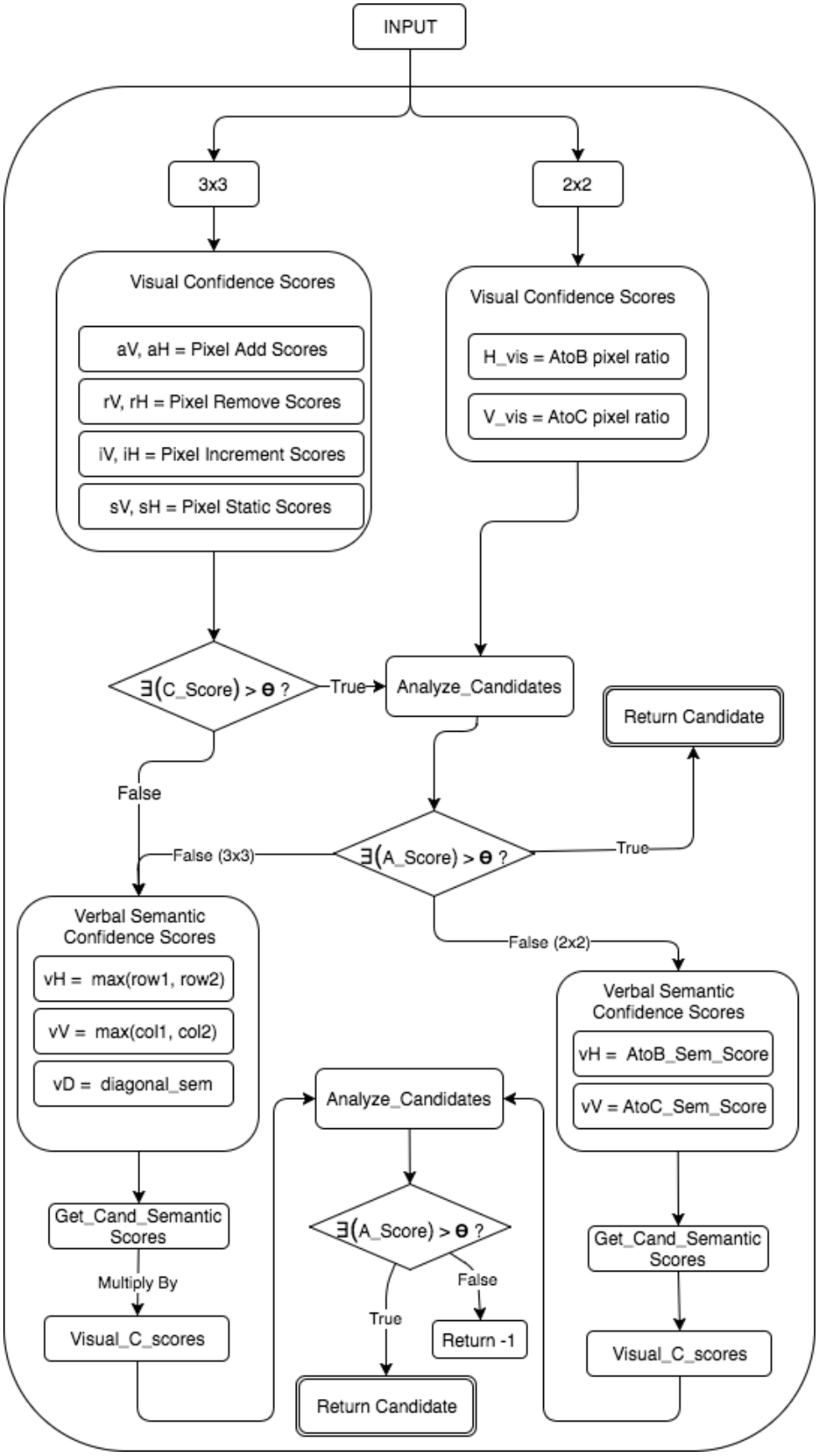

Figure 1

Agent Limitations:

When evaluating the visual transformations (addition, deletion, incrementation, and static), my agent solely relies on the generated pixel size when comparing it to the solution set. While this approach gave my agent the ability to correctly identify the correct candidate most of the time, it is a somewhat naïve approach especially for future problems that we might encounter.

There are several improvements that could be made to our current approach. Rather than just using the total size of pixels as a means of testing, future visual examinations can include layering of images on top of each other to check for pixel similarity, calculating the centers of images, performing breadth first search to find strongly connected components (this can help us calculate object counts without relying on verbal representations), and many other powerful strategies for effectively extrapolating useful information.

Other limitations that need to be addressed include specific advanced pattern recognition techniques that will be especially useful in future RPMs that will be inevitably more challenging. For example, given my agent's current implementation, it lacks the capability to detect whether or not translation patterns exist between rows or columns. To accommodate for this, my agent needs to be able to detect for shifting of elements to the left, right, up, or down by exploiting array rotation.

Agent Performance:

Compared to the runtime of my agent prior to utilizing visual representations, my agent's current runtime is significantly slower in comparison (~+6 seconds) as it must iterate through the entirety of several images when calculating pixel sizes. Careful consideration of runtime and space must be accounted for future RPMs especially since future design is going to rely heavily on pixel manipulations and examinations. The current design approach ensures that pixel data for different images and transformations is calculated just once and then stored in a dictionary for later lookup.

The agent was able to correctly solve 11/12 and 8/12 of the basic and test problems respectively. Additionally, improvements were made to the 2x2 RPM problem scores by incorporating visual transformations that geared towards 2x2 problems specifically by evaluating the ratios of the black pixels of A -> B and A -> C, mapping these ratios to C and B respectively, and finally calculating the confidence scores of the candidates based on these ratios.

Certainly, there are some deficiencies of my agent when it comes to being able to generalize holistically. Specifically, one of the flaws of generality comes from the fact that the thresholding of the visual transformations tend to over-fit to the problem sets. To accommodate for this flaw, future agent design must incorporate more complex visual transformations as discussed previously.

KBAI Ebook: Knowledge-based Artificial Intelligence ©2016 Ashok Goel and David Joyner

Goel, Ashok. Joyner, David. Udacity Video Course: Knowledge-Based AI: Cognitive Systems
Link:
https://classroom.udacity.com/courses/ud409/lessons/1778648600/concepts/1800638544092
3