CS 7637 Project 3 Reflection:

Abhijeet Gaurav
agaurav@gatech.edu

Introduction

The goal of KBAI course is to learn to build human like intelligence. In this journal we'll discuss my approach to building an AI agent that will solve Raven's Progressive Matrices (RPM) which were used in the past to test abstract reasoning and human intelligence. This journal covers the entire design and implementation of my AI agent that will solve the 3x3 RPM problems and will build up (to some extent) on my AI agent that solved 2x2 RPM problems for Project 1 and 3x3 RPM problems for project 2. 3x3 PRM problems are similar to 2x2 RPM problems where there is some pattern/relationships among a set of given 8 figures (fig A to fig H). Using the same observed relationship between these 8 figures, the AI agent has to identify the 9th figure that best follows a similar relationship. This optimal figure that completes the relationship is given as one of the 8 options (fig 1 to fig 8) to the AI agent to pick from.

Building Project 3 to solve on top of Project 1 and Project 2:

I decided to scale my 3x3 RPM problem solver to solve project 3's 3x3 RPM problems as I thought my 3x3 RPM problem solver was very robust (although it overfitted to some extent on the Set C basic problems. My 3x3 AI agent from project 3 used to use scoring mechanism based on dark pixel density to determine how dark pixels increased/decreased in horizontal/vertical 3x3 relationships and scored options based on this metric, diff image scorer which subtracts one image from another which helps in identifying shape change in horizontal/vertical relationships, reflection scorer which checks if the figures follow any reflection relationship, slice image and compare scoring mechanism which slices the image horizontally/vertically into two parts and then left slice of each image is compared to right slice (and vice-versa) of the other images to determine if they are equal. Each of these scoring mechanisms was explored in project 2. This resulting performance of my AI agent was: Set C: Basic problems: 11/12 ; Test problems: 10/12; Ravens problems: 9/12. On the other hand, my 2x2 AI agent used to do transformations on A, compare it with B/C to see if there is a horizontal/vertical relationship. If horizontal/vertical relationship was found, same transformation on B/C was applied and then compared to 6 options.

Similarly, the 2x2 AI agent also used to do transformations on B, compare it with C to determine if there is a diagonal relationship. If diagonal relationship was found, similar transformation was done to A and transformed A was compared with all the options. Each option was assigned a score. Option with the highest score was my agent's answer. Applied transformations were either trivial (rotation/identical/reflection) or non-trivial (compare difference of difference image/fill image with color transformation). Using this approach my AI agent solved Set B: 12/12 basic 2x2 problems, 9/12 test 2x2 problems, 8/12 raven 2x2 problems; 7/12 2x2 challenge problems. However, as you can see the underlying logic for my 2x2 RPM problem solver is quite different from 3x3 RPM problem solver for project 2. This was because the 3x3 RPM problems in project 2 were quite different from 2x2 RPM problems. 3x3 RPM problems given in the Project2 did not have rotation (and a direct reflection) relationship. Also, there are no 3x3 problems of filling the entire figure with black color.

History has repeated again with the Project 3. Again, I found out that Set D and Set E 3x3 RPM problems are quite different from Set C 3x3 RPM problems. Therefore, I had to almost again start from scratch for Project 3. However, in my final AI agent logic, I still kept all the scorer mechanisms that solved Set C 3x3 RPM problems and Set B 2x2 problems. This allows my Project 3's AI agent to work across all the Set B, Set C, Set D and Set E problems.

In my Project 3, I had to add multiple new scoring mechanisms to my AI agent to make it work on Set D and Set E problems. The majority of new scoring mechanism is built on top of dark pixel density percentage logic. These new scoring mechanisms are based on 'all pattern exists scorer', 'intersection of two figures and compare with the third figure scorer', 'union of two figures and compare with the third figure scorer', 'subtract figure in terms of dark pixel density and compare scorer', 'Remove intersection pixels and sum remaining dark pixels scorer' and 'split/slice and compare scorer'. Each of these scoring mechanisms will be explored in the submission section of this journal.

First Submission

The first submission was sent on 2019-12-01 00:32:25 UTC. This first submission code is the same as Project 2's final AI agent code except I removed the reflection property scorer, which determines if the 3x3 RPM problems have reflection relationship and slice image and compare scoring mechanism which slices the image horizontally/vertically into two parts and then left slice of each image is compared to right slice (and vice-versa) of the other images to determine if they are equal. I removed these two scoring mechanisms because none of the Set D and Set E problems have a reflection property or sliced image equality property. Also, if my agent is not able to determine answers to an RPM problem, it by default chooses the first option. This submission was sent for ‘trial and error’ to determine if the Set D and Set E: Ravens and Test problems are similar to Set C problem and to out the number of Set D and Set E problems the AI agent with almost the same code as Project 2 can solve.

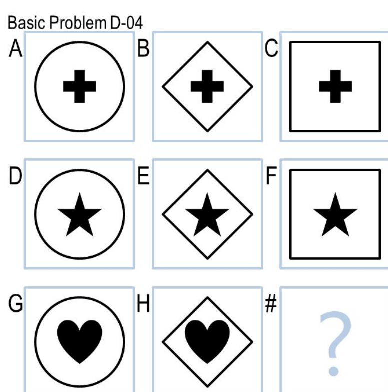
My AI agent is not similar to humans at all. This version of AI agent uses multiple scoring mechanisms which is not intuitive at all compared to human way of solving the 3x3 RPM problems. My AI agent solves every problem by doing some pixel counting, manipulation, subtraction, dark pixel increment, dark pixel decrement which is not like humans at all.

Performance:

Problem Set D: Basic problems: 6/12 ; Test problems: 2/12; Ravens problems: 1/12; Challenge problems: 3/12. Problem Set E: Basic problems: 3/12 ; Test problems: 2/12; Ravens problems: 2/12; Challenge problems: 3/12. This version of AI agent performed miserably on both Problem Set D and E. An okayish score on Set D basic problems is observed because this version of AI agent defaults on choosing the option 1 when it is not able to determine an answer. Basic problems D-02, D-04, D-06, D-07, D-10, E-01, E-08 all have correct answer of 1. In fact, the only basic problem this version of AI agent was able to determine correctly was D-01 (which has an identical horizontal relation) and E-11 (which has row1 ∩ row2 = row3 and col1 ∩ col2 = col3 but is still solved because the problem has decreasing pixel density order both horizontally and vertically). In terms of **efficiency**, my agent took 31.657 seconds to execute all the problems. The algorithmic complexity of my AI agent is O (n) where n is the number of options.

Second Submission:

The second submission was sent on 2019-12-01 09:42:22 UTC. I almost started from scratch for this submission as the Project 2’s AI agent was not able to solve much in submission 1. In this submission, I mainly targeted Set D basic problems. I developed 3 scoring mechanisms all based on the dark pixel density. (1) All patterns exist scorer: Contrary to the name, this scorer just checks if the sum total of all the dark pixels across all the figures in rows/columns are equal. If the sum of dark pixel density is equal for 2 rows/columns, option which makes the sum of dark pixel density of the 3rd row/column equal to the other rows/columns are given a score of 1, 0 otherwise. (2) Remove intersection pixels and sum remaining dark pixels scorer: This scorer first calculates the intersection pixel density in each row/column, and then subtracts this intersection pixel density from each figure in the same row/column to get the remaining pixel density of each figure. It then sums the remaining pixel density of all the figures in rows/columns and checks if they are equal. If the sum of remaining dark pixel density is equal for 2 rows/columns, similar intersection pixel density is taken for option and the 3rd row/column figures and the resulting intersection pixel density is subtracted from each image in the 3rd row/column and the option. The option that makes the sum of remaining pixel density equal to the other rows/columns are given a score of 1, 0 otherwise. (3) Union Pixel density scorer: This scorer takes the union of all the figures in a row/column and checks if the dark pixel density of the union figure is the same between rows/columns. If the union pixel density between 2 rows/columns are the same, the option which makes the union of the third row/column same as the other 2 rows/columns are given a score of 1, 0 otherwise. The problems such as D-02, D-03 and D-11 were solved using 'all patterns exist scorer' as the sum of raw dark pixel density is constant across rows/columns; problems such as D-04, D-06 were solved using 'Remove intersection pixels and sum remaining dark pixels scorer'. Problem D-10 was solved using 'Union Pixel density scorer'. Problem D-01 was solved using the logic from submission 1. Other remaining problems such as D-09 and D-05 were solved using the scoring combination of (2) and (3). An example, in problem D-04, if we remove the intersection of pixel density in a row (+ first row, * in second, ♥ in third) from each of the rows, the problem now becomes a much simpler problem of checking if each of the remaining figures do exist across row/column. My agent does this by checking if the sum of pixel density of the resulting figure is constant across rows/columns.



My AI agent is not similar to humans at all. This version of AI agent uses three scoring mechanisms which are not very intuitive to human way of solving the

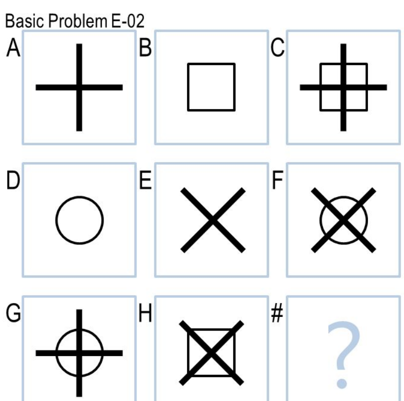
3x3 RPM problems. Example: 'All patterns exist scorer' of my AI agents adds up all the pixels in all the figures in row/column and then compares the sum of pixel density across rows/columns. Instead as a human, we see a set of figures, and would know instantly the figure from the options which is missing by figure comparison and not pixel density addition and comparison.

Performance:

Problem Set D: Basic problems: 9/12 ; Test problems: 9/12; Ravens problems: 6/12; Challenge problems: 1/12. Problem Set E: Basic problems: 3/12; Test problems: 3/12; Ravens problems: 4/12; Challenge problems: 3/12. The AI agent improved a lot after I added the above 3 scorers. It didn't do very well on Ravens problems though and I don't have access to those problems to figure out the exact reason for this. The AI agent wasn't able to solve Basic: D-07, D-08 and D-12 as it doesn't have logic to solve such kind of problems. In terms of **efficiency**, my agent took 46.06 seconds to execute all the problems. The algorithmic complexity of my AI agent is O (n) where n is the number of options.

Third Submission:

The third submission was sent on 2019-12-01 21:20:14 UTC.



In this submission, I mainly focused on Set E Basic problems. I implemented a new scoring mechanism which checks if the dark pixel density of the union of two figures in row/column is equal to the dark pixel density of the third figure in the same row/column. If the dark pixel densities are equal, the option which satisfies the same property is given a score of 1, 0 otherwise. Eg: In the problem E-02, we can see that Row1 ∪ Row2 = Row 3 and Col1 ∪ Col2 = Col3. Based on the scorer, in this case, the agent's would score the option which is equal to C ∪ F and G ∪ H.

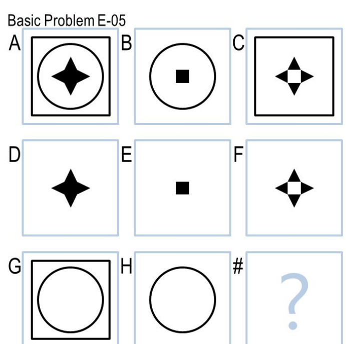
My AI agent is not very similar to humans. This version of AI agent uses scoring mechanisms which deals with union of two figures, but the final comparison between the resulting union figure and the third figure is done using dark pixel density. Therefore, I don't think it is very human. However, if you remove the dark pixel density comparison at the end, the AI agent does union of figures which is very like human technique to solve RPM problems.

Performance:

Problem Set D: Basic problems: 9/12 ; Test problems: 9/12; Ravens problems: 6/12; Challenge problems: 1/12. Problem Set E: Basic problems: 7/12 ; Test problems: 6/12; Ravens problems: 4/12; Challenge problems: 3/12. Adding this logic significantly improved the agent's score substantially on Set E problems. Using this logic, AI agent was able to solve basic problems: E-01, E-02, E-03, E-06 as all of these problems show this union relationship. The agent also solved E-10, E-11 although it should have been solved by more robust logic. The problems E-10, E-11 also show union property in terms of dark pixel ratio and thus my agent was able to solve them. Eg: In E-10 problem, if we do Row2 U Row3, the resulting pixel density is equal to Row1 pixel density. The agent couldn't solve problem E-08 but since the default option the agent chooses is 1, the AI agent answered it correctly. The AI agent wasn't able to solve remaining Set D basic problems as it doesn't have logic to solve such kind of problems. In terms of **efficiency**, my agent took 54.604 seconds to execute all the problems. The algorithmic complexity of my AI agent is O (n) where n is the number of options.

Fourth Submission:

The fourth submission was sent on 2019-12-01 21:58:02 UTC. I found that third Submission didn't solve Basic E-05 which is also a union problem. In this submission, I made my union logic more robust by adding logic to also consider Row2 ∪ Row3 = Row1 and Col2 ∪ Col3 = Col1 (instead of just considering first and second row/column). I also made some black pixel density threshold adjustment. These changes were done so that AI agent could solve problem E-05.



My AI agent is not very similar to humans. The reasoning for this is exactly the same as the one explained in Third submission.

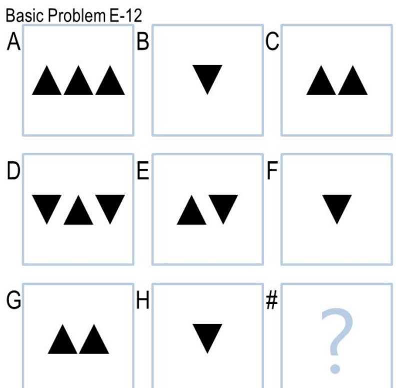
Performance:

Problem Set D: Basic problems: 9/12 ; Test problems: 9/12; Ravens problems: 6/12; Challenge problems: 3/12. Problem Set E: Basic problems: 8/12 ; Test problems:

6/12; Ravens problems: 4/12; Challenge problems: 3/12. With this change, my AI agent is now able to solve Basic problem E-05. No performance changes in any other set. In terms of **efficiency**, my agent took 54.736 seconds to execute all the problems. The algorithmic complexity of my AI agent is O (n) where n is the number of options.

Fifth Submission:

The fifth submission was sent on 2019-12-01 23:21:27 UTC. In this submission, I targeted problems Basic E-04 and Basic E-12. Both of these problems have image subtraction. However, the major problem in both of these problems is that the resulting figure has change of position (E-04)/orientation(Basic E-12). I implemented dark pixel subtract scorer which subtracts a figure in 1 row/column from another figure in the same row/column and checks if the resulting subtracted dark pixel density is equal to the figure in third row/column. However, just subtracting dark pixel density of two figures didn't solve this kind of problems perfectly as both of these problems have two options that match the subtracted dark pixel density. Therefore, we need to do positioning changes (for E-04) and keep track of orientation (for E-12). I'll address this later if I get time. For now, I hardcoded in my solver to always choose the option which has lower value in case of this kind of race condition. This means that, if option 4 and option 5 are possible answer, my agent picks 4 because 4 < 5.



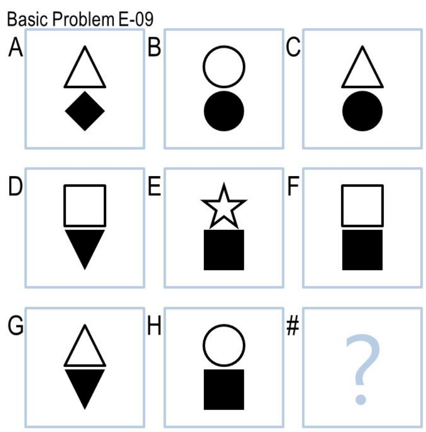
My AI agent is not similar to humans. A human won't solve this problem by calculating dark pixel density of each figure, subtract them with one another and compare with the third figure to figure out the relationship. We as humans, look at the overall figure and don't think in terms of dark pixel density. Also, we are very good with tracking position/orientation which my agent is very poor at.

Performance:

Problem Set D: Basic problems: 9/12 ; Test problems: 9/12; Ravens problems: 6/12; Challenge problems: 1/12. Problem Set E: Basic problems: 9/12 ; Test problems: 6/12; Ravens problems: 4/12; Challenge problems: 3/12. With this change, my agent was able to solve Basic E-12 but it failed to solve problem E-04. This is because of the race condition between two options that have the same dark pixel density as the subtracted dark pixel density (Col1 -Col2). In case of E-12, the correct answer is 6 and the two options that have the ((pixel density of Col1) - (pixel density of Col2)) is 6 and 7. Similarly, for problem E-04, the correct option is 8 but the two options that have the same pixel density is 2 and 4. Hardcoding this to choose the lower option means that my agent picks correct option 6 for E-12 but wrong option 2 in case of E-04. In terms of **efficiency**, my agent took 80.996 seconds to execute all the problems. The algorithmic complexity of my AI agent is O (n) where n is the number of options.

Sixth Submission:

The sixth submission was sent on 2019-12-02 00:23:21 UTC. In this submission, I was able to solve Basic problem E-09. To solve this problem, I added split/slice and compare scorer to my AI agent. My agent slices figures horizontally in the middle. The lower part of figure in Column2 is compared with the lower part of figure in Column3 within the same row and similarly upper half of figure in Column1 is compared with the upper half of figure in Column3 within the same row. If there is an exact match in top 2 rows, similar comparison is done with all the options by slicing them with row3. The option that satisfies the relationship is given a score of 1, 0 otherwise. Similarly, this comparison is also done across columns.



My AI agent is similar in this case to humans way of solving problems. I would tackle this problem similarly by looking for upper half comparison and lower half comparison. The AI agent follows the exact same steps too.

Performance:

Problem Set D: Basic problems: 9/12 ; Test problems: 9/12; Ravens problems: 6/12; Challenge problems: 1/12. Problem Set E: Basic problems: 10/12 ; Test problems: 6/12; Ravens problems: 6/12; Challenge problems: 5/12. With this change, my AI agent was able to solve Basic problem E-09. No performance changes in any other set. In terms of **efficiency**, my agent took 73.757 seconds to execute all the problems. The algorithmic complexity of my AI agent is O (n) where n is the number of options.

Seventh Submission:

The seventh submission was sent on 2019-12-02 01:55:10 UTC. In this submission, I mainly tried to improve the efficiency of my code as it was taking too much time to run on all the problems.

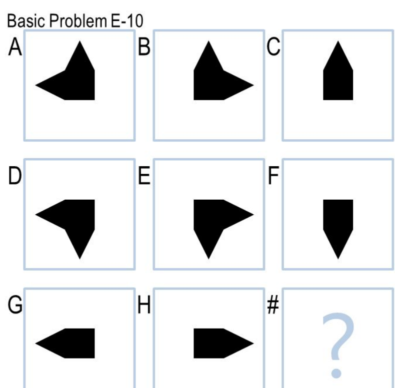
My AI agent is not similar to humans. The code efficiency of solving problems can be increased but the same is not true for humans.

Performance:

Problem Set D: Basic problems: 9/12 ; Test problems: 9/12; Ravens problems: 6/12; Challenge problems: 1/12. Problem Set E: Basic problems: 10/12 ; Test problems: 6/12; Ravens problems: 7/12; Challenge problems: 4/12. No performance changes in any set. In terms of **efficiency**, my agent took 57.534 seconds as opposed to 73.757 seconds in sixth submission to execute on all the problems. The algorithmic complexity of my AI agent is O (n) where n is the number of options.

Eighth Submission:

The eighth submission was sent on 2019-12-02 02:35:53 UTC. In the last 4 submissions, I wasn't able to increase my AI agent's score on Set E test problems. To fix this, I decided to solve Basic E-10 and E-11 using more robust way. Although, I found that these problems also show union property in terms of dark pixel ratio and thus my agent was able to solve them in the third submission, more robust way to solve these two problems will be to consider relation Row1 ∩ Row2 = Row3 and Col1 ∩ Col2 = Col3. Therefore, in my AI agent's logic, I added intersection of two figures and compare scorer. The idea here is to take any two rows/columns perform an intersection on them and check if the resulting figure is equal to the third figure in the same row/column. If such a relationship is there, do a similar intersection/comparison



with all the options. The option that is the exact match gets a score of 1. 0 otherwise.

My AI agent's approach to solve these problems is very similar to humans. Taking intersection to solve this problem is the exact same thing we do as humans do to solve this problem.

Performance:

Problem Set D: Basic problems: 9/12 ; Test problems: 8/12; Ravens problems: 5/12; Challenge problems: 1/12. Problem Set E: Basic problems: 10/12 ; Test problems: 10/12; Ravens problems: 6/12; Challenge problems: 4/12. With this added new logic, my AI agent improved significantly on the Set E Test problems. However, I also see the addition of this logic caused a slight decrement in Set D Test and Ravens problem scores. I think changing the weightage of each of the scorers during final scoring calculation in my AI agent will fix this issue. The AI agent is still failing on the other remaining unsolved problems because it doesn't have appropriate logic to solve them. In terms of **efficiency**, my agent took 63.359 seconds to execute all the problems. The algorithmic complexity of my AI agent is O (n) where n is the number of options.

Conclusion:

I started this project project with a trial and error approach of submission 1 to see if the 3x3 RPM problem solver from project 2 can run well on Project 3 RPM problems. Since the first submission, did not work well on Set D and Set E problems, all the subsequent improvements in the AI agent has been done with deliberate improvements. In all of my implementation and submissions except seventh submission, I have incrementally added independent logic in form of scorers to improve my agent by targeting one or multiple similar problems at a time. In the seventh submission, I increased the efficiency of my agent which decreased the running time of the agent by more than 20%. The final version of my agent was able to achieve okayish scores w.r.t. the goal of achieving 7/12 across Set D and Set E basic and test problems. The final scores were: Set D: Basic problems: 9/12; Test problems: 9/12 and Set E: Basic problems: 10/12; Test problems: 10/12. Having the same scores for Basic and Test problem also shows that the AI agent is not overfitting on the basic problems. The agent was not able to solve 3 problems in Set D and 2 problems in Set E because it currently lacks the logic for solving those kinds of problems.

My AI agent for is not similar to humans overall. My AI agent uses multiple scoring mechanisms which are not very intuitive to human way of solving the 3x3 RPM problems. My AI agent solves every problem by doing some pixel counting, manipulation, subtraction, dark pixel increment, dark pixel decrement. It has scoring mechanisms based on 'all pattern exists scorer' which does pixel addition, 'intersection of two figures and compare with the third figure scorer', 'union of two figures and compare with the third figure scorer', 'subtract figure in terms of dark pixel density and compare scorer' which does pixel addition/subtraction, 'Remove intersection pixels and sum remaining dark pixels scorer' which does pixel addition/subtraction and 'split/slice and compare scorer'. The three scoring mechanism used by AI agent which are somewhat human like are 'intersection of two figures and compare with the third figure scorer', 'union of two figures and compare with the third figure' and 'split/slice and compare figures'. These three scoring mechanisms solves the problem overall exactly the same as how humans do (except maybe if we look this from the lower level perspective of pixel manipulation). Therefore, overall, I would say my AI agent doesn't solves RPM problems as humans/I do.

If I had more time, I would have tried to use edge detection techniques, fractal method and would have tweaked dark pixel/image equality threshold to see if those help with the Test & Ravens problems. I would have also tried to implement logic in my AI agent that could determine shapes and corresponding transformations. Ideally, if I had more time, I would have also explored analogical reasoning method to solve RPM problems. To improve my AI agent's score, I would have tried to come up with logic to solve Basic problems that are currently not solved by my AI agent such as: D-07, D-08, D-12, E-07, E-12. I would have also tried to come up with logic to solve Challenging problems. Currently, in my AI agent logic, sometimes the code is duplicated and pixel density is calculated multiple times for the same figure. I would have reduced this redundant calculations which would have decreased the run time of my code significantly and would have increased the efficiency. Also, to make my code more readable, I would have refactored all of my AI agent's scoring methods.