Project 1: Raven’s Progressive Matrices Journal

Sergiy Palguyev

sergiy.palguyev@gatech.edu

Introduction

Raven’s Progressive Matrices (RPM) test is a test of intelligence most often associated with proving human-level intelligence for a computer program. The test is non-verbal and uses visual figured in order to determine levels and complexity of intelligence of a test taker. Most common RPM tests are shown as a 3x3 matrix problem where the last picture must be inferred from the other 8 pictures. A solutions table provides the possible choices for an answer, only one of which is correct. – (reference)

Strategy

The strategy implemented using visual approach only but requires multiple logical processions to occur.

Firstly, it is important to not only get the largest amount of correct values, but subsequently the least amount of Incorrect values. If the algorithm skips certain values rather than attempt to solve them and get them incorrect, then there is room in the algorithm for improvement in new strategies to detect the “correct”-ness. On the other hand if the implementation evaluates multiple answers as Incorrect, then the implementation itself must be re-evaluated since its algorithm improperly attributes a problem to the algorithm which cannot solve it.

Secondly, the visual approach will depend on the transformations that occurs form image to image across rows and columns in order to determine the answer. It is evident that RPMs are not restricted to only one transformation and thus may include multiple changes in each stage. For the first attempts, only one transformation level will be evaluated, with multiple transformation levels being implemented in further development, or if time permits.

This strategy is in line with human-level approach which would solve the problem from starting from simplistic single level transformation approach, and only after exhausting all possibilities, move to evaluate multi-transformational approaches.

Implementation

The algorithm for solving RPMs is implemented in Python with only Pillow and NumPy libraries allowed as utilities. Each image of the problem matrix is converted to a Numpy binary matrix array. This is done in order to easily evaluation shapes against a background. For the current problem set, a binary representation is enough as no gray scale transitions are present. Two global functions are implemented that help determine the solutions in the current problem set.

The function “diffStatistics” is implemented to take two matrix representations of images and compare them together. The mean of the two images against each other compared to a provided threshold will provide the similarity of the two images. A value of 1.0 means the images are exactly the same. Any value lower will imply the images differ by some extent. A default value of 0.98 as the threshold is given as a high confidence value. Additionally a Root Mean Square is calculated in addition to a Pillow function which determined the bounding box of the area where any difference in images is located. Both of these data points may be useful in future logic implementations.

The function “equate” is used to scan through the solutions and compare each one to the best guess as determined by the implemented logic. The solution with the best match gets selected as the answer, otherwise a “-1” becomes the answer, implying the problem should be Skipped.

1st Submission (Failed - Missing Libraries)

For the first attempt at a submission, multiple issues arose with connecting with Bonnie, the primary one was a lack of the “future” library, which could not be solved in an appropriate amount of time. Thus, this reflection is of the state of the code at the time running local tests, and not a reflection of the Bonnie tests. Currently, only the algorithms for equality of images, horizontal mirroring of images, and vertical mirroring of images are implemented. The “equate” function is used to run all the tests with the default similarity threshold of 0.97. Curiously, The Basic Problem B-11 is also solved correctly. This is most likely due to the fact that the “diamond” figure in the image A has a very small footprint, allowing a fair similarity between image A and B. Most importantly, the “equate” function will scan through all solutions and match the similarity of C to a solution as close as possible to the similarity of image A to B. Thus, even image 2 is the exact same image as image C allowing 1.0 similarity, it is not chosen since the similarity of image C to image 1 better matches the similarity of images A to B. In local tests, 9 Basic problems are solved and 1 Challenge problem. Only one problem is marked incorrect and all others are Skipped.

2nd Submission (Failed - 2FA)

Still unable to submit to Bonnie, the second working version of the RPM solving algorithm now contains provisions to evaluate images where a shape transforms from hollow to a solid image of the same shape. This provide a shape with high similarity, since only a sliver of the shape is missing between the two images. For this test, the “equate” function is passed two thresholds, The first threshold is for comparing A to B or A to C and is valued at 0.96. The second threshold is to compare the logical answer for D with that of the possible solution set and is set to a value of 0.90. Although this evaluation of a solid filled body is not very elegant, it does prove fruitful in solving problem B-09 without introducing any “Incorrect” answers anywhere else in the problem set.

3rd Submission (Success)

Finally, the third submission is on the day of the deadline and all Bonnie issues have been resolved with the help of Piazza and fellow classmates. At this time, rotating the figure through seven different angle values is implemented as part of the algorithm. Implementing this transformation evaluation leads to correct solution for 3 more challenge problems. At this stage, on the local problem set, 10 Basic problems and 4 Challenge problems are correct. One Challenge problem is incorrect and the rest are skipped. On Bonnie, a similar picture is evident, 10 Basic, 4 Challenge, 7 Test, and 6 Raven’s problems are correct.

In order to gain some points on random chance, if after all previous logic the return value is “-1”, then simply guess one of the 6 answers. This is ok to do since throughout the length of the algorithm, incorrect answers are wholeheartedly minimized in order to optimize each portion of the algorithm to detect the figures intended. By the end of the algorithm, if detection is not possible, a simple guess can only increase chances for the results. At this point on Bonnie 10 Basic, 6 Challenge, 8 Test and 7 Raven’s problems are marked Correct. While it is true that without randomly guessing on the skipped problems still leads to an evaluation that is better than random chance (10 Basic and 7 Test problems Correct), if we consider how a human would evaluate the same problems, a test – taker would not leave the problems blank, but guess in hopes of getting at least some points correct.

Conclusion

In conclusion, the currently implemented strategies seem to be adequate for the Basic test set, but barely good enough for the Test test-set, getting only 7 out of 12 problems correct before random chance takes over. It is beneficial to have started the RPM project immediately with the visual approach since it provides a head start into thinking through problems of visual categorization and evaluation in preparation for the next problem stages.

Looking at the project from a human perspective, it is evident that the approach undertaken follows basic human thought. The very first level of complexity is single level transformation (i.e. image equivalency, mirroring, solid/hollow fill and rotation). The next level will be two or more transformations between images and more complex image transitions between frames which involve higher human intelligence as well. Additionally, in the best of human fashion, when an answer is not easily revealed, one would guess a random answer, since Incorrect answers do not discount from the total score. It is for this reason that at the very end of the algorithm, skipped problems get to randomly pick a value as the answer.

Future

For future development, multi-level transformations will be a must-have feature as human cognition would dictate. Deciding whether certain problems involve both a rotation and solid fill, or erasure and mirroring, or other combination of factors will provide most challenging. Furthermore, certain parts of the algorithm can be refined (i.e. solid-fill detection, rotation, skew translation, edge detection, figure detection, etc). Additionally, the bounding-box feature implemented but not used in the diffStatistics function will definitely become more and more helpful.