Agent design to solve Raven's Progressive Matrices

Nilesh More (nileshmore@gatech.edu)
GTID – 903081902

Contents

Reasoning used by the Agent

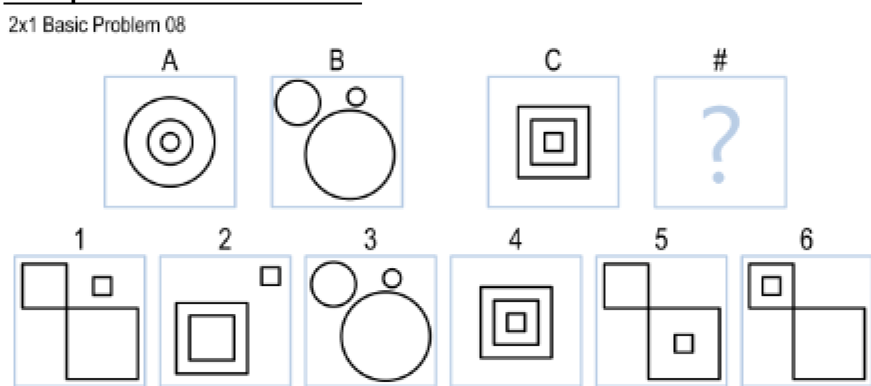
This agent is less intelligent but smarter in a sense that it does not really have too much knowledge about the geometrical properties of Raven's problems. Rather, it uses complete image matching/difference and combined with some logical filters. This helps agent to get correct answers for 35/60 basic problems.

Image matching scheme used by agent is structural similarity index (SSIM). When comparing images, the mean squared error (MSE)–while simple to implement–is not highly indicative of perceived similarity. Structural similarity aims to address this shortcoming by taking texture into account.

Simply calculating difference between corresponding images and matching those differences against other pair does not result into high precision. To overcome this, it uses filters like – If Shape has changed from fig. A to fig. B then shape must also change from fig. C to correct answer choice.

I will describe the reasoning with examples in more details below. Please note that I decided to use this similarity matching scheme based on my experiments. Experimental data was encouraging enough to use this strategy.

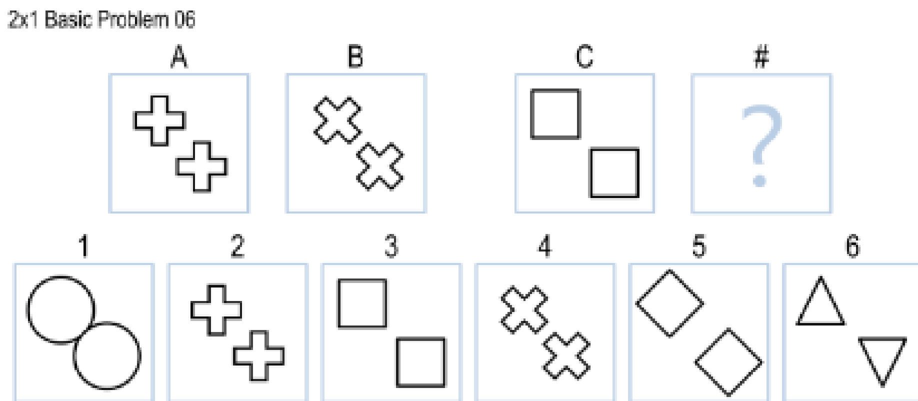
Examples for correct answers



Generic Image matching – By taking the numerical difference of SSIM for A and B and looking at SSIM difference between C and each choice, and then matching those differences to find the closest match which would have closest SSIM difference value to that of difference of A-B.

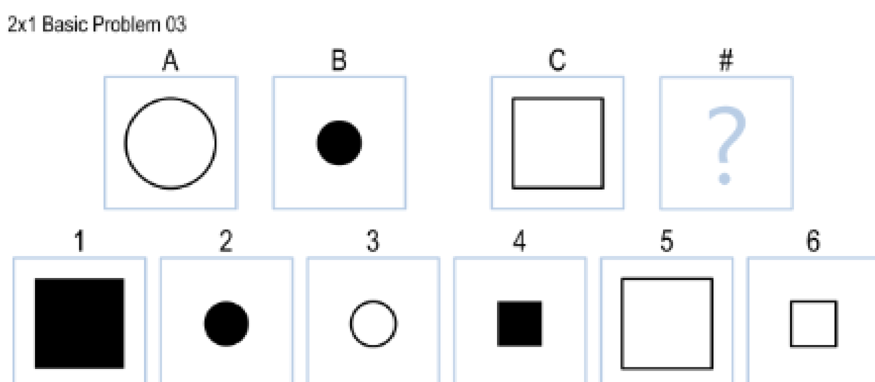
It was very surprising for me that simple using SSIM based difference matching scheme, agent was able to arrive at answer choice '1' with complete confidence (A->B and C->choice AND A->C and B->choice both pointed to choice #1).

Here it did not need to use any filters.



Using only the generic image matching scheme, agent thinks that answer choice '6' is the correct answer. In this case, filtering comes to rescue.

Filter used: Shape should not change between fig. A and fig. B. This filter eliminates all choices but choice #3 and #6. Agent is not left with easy task of choosing between 3 and 6 which it does with complete confidence by using image matching scheme.



Filters used – 1. Shape should not change between C and option.

2. Fill should change between C and option.

3. Number of contours should decrease between C and option. I have computed number of contours as number of corners in a given figure. Smaller circle would have less number of corners than bigger one.

Using these filters, agent is left with only 1 option and it does not have essentially use the image matching scheme.

As could be imagined, the agent's performance is miserable in 3x3 since there the filtering schemes are not so clearly defined and it has to rely on weak filters and its inherent image matching which again is not clearly defined. Examples to follow –

3x3 Basic Problem 13

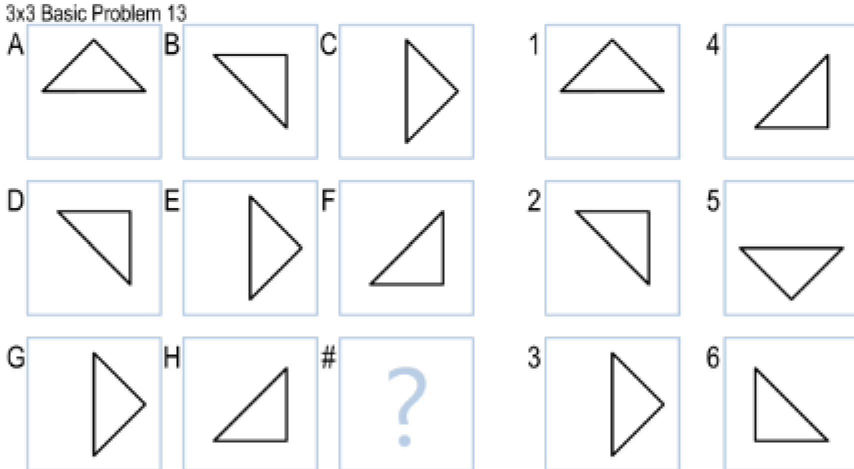
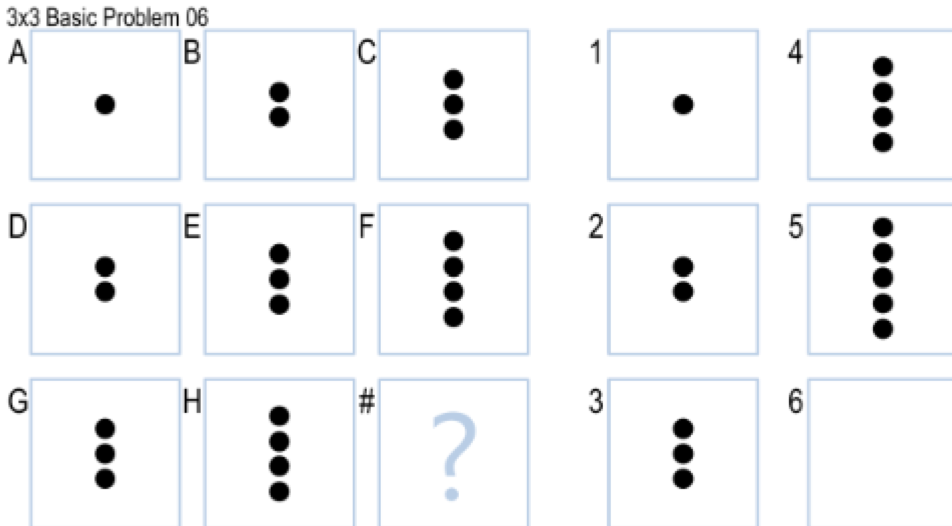


Image matching strategy –

Unlike 2x1 and 2x2, the SSIM difference taken is between A+B and A+C, D+E and D+F and finally G+H and G+Option.

You might be wondering as to difference between A+B and A+C would be same as difference between B and C. NO. Please remember though we are adding images pixel by pixel, the difference is not pixel by pixel. SSIM models differences by taking into account their structures and hence to add more precision for image matching first column is added to second and third column before taking difference of second and third column.

Image matching scheme again surprised me in a sense that it was able to identify angle changes with precision. Here this scheme identifies '5' as the correct answer without any hesitation.

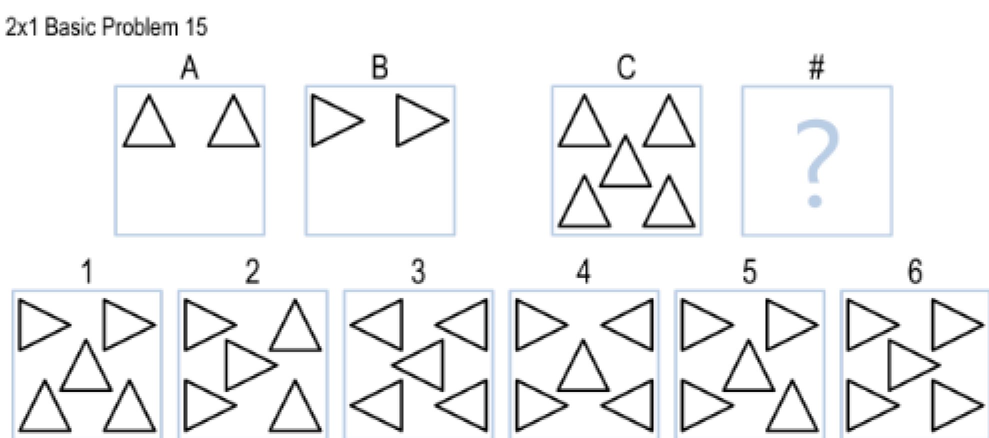


In the above example agent has to use filtering to come-up with correct answer. With image matching scheme described above, this agent turns itself into a big fool.

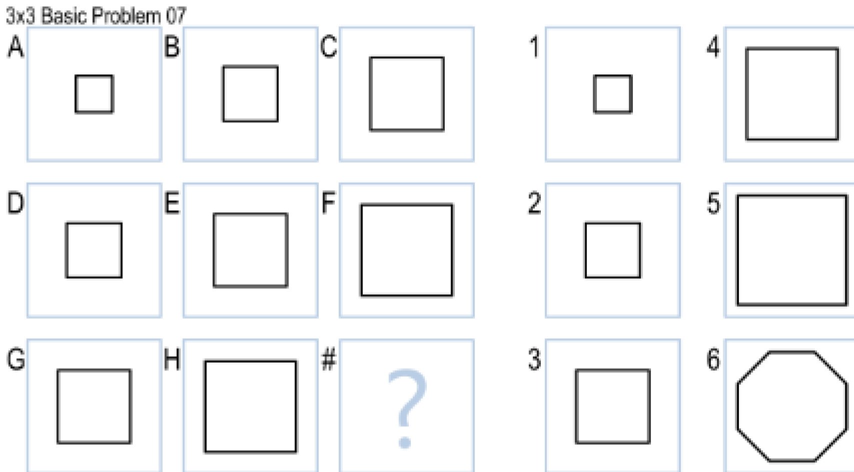
It thinks choice #4 is the correct answer. you might wonder why but remember we combined first column with second and third before taking the difference. That leads to failure in this case. Filtering to the rescue again ☺

Filtering scheme – number of contours should increase in successive columns. Once agent uses this filter, it is left with only the correct option!

Mistakes by agent



Here, agent notices that triangles with horizontal base should rotate by 90 degree. It applies this observation and comes-up with answer as choice #1. Please note that no filter described above can help agent.



In the above example, number of contours remain same since my agent models contours as number of corners and those are four for every figure. Another filter of counting objects and see if 3-3-2 pattern exists also does not help here since each object is different.

Poor agent with only reliance on image matching thinks choice #7 is the correct answer.

Further Improvements

I would have really liked to look into the details of SSIM matching scheme and based on the understanding use that only for certain number of problems. What I mean is agent could have used visual and pre-positional strategies for different set of problems depending on the effectiveness. Currently, it used naïve filtering with visual reasoning.

Efficiency of the agent: how long does it take to run? What kinds of problems will force it to take longer?

Agent does not extract individual shapes from Raven's figure, instead it uses complete image matrix to do the image comparisons, agent's running time would be order of O(n + e) where n is number of pixels since it does corresponding pixel match and 'e' for accounting changes in algorithm to model structural similarity.

It takes ~2 minutes to complete the basic set of problems. Another time consuming factor is template matching used by agent to identify fill which also has polynomial running time.

Differences perceived by the agent as against earlier pre-positional representations

It was lot easier to solve 2x1 and 2x2 problems using visual representations than pre-positional since the image matching scheme was clearly defined.

But for the 3x3, the matching scheme was not clear enough. Agent found it a lot easier to go with pre-positional representations for 3x3 problems.

Also extracting precise pre-positional representations is a known hard problem to solve and this particular agent extracts minimal information which also is not precise! Fails to identify filled shapes from non-filled ones.

Reflecting Back: What does this agent tell us about how people might solve these problems and relationship between the agent and human cognition?

This agent mimics humans with less knowledge about geometry but who are very smart. Core reasoning used by humans all the time is choice eliminations – my agent did not mimic that in earlier three projects but this time it uses it explicitly by using filters described above.

In terms of image matching, I would say it does not really mimic the humans. It still acts at superficial level of images at pixel level and fails to identify general pattern changes (which humans do very easily) - as described in mistake example #1.

References

1. Kunda, M., McGreggor, K., & Goel, A. K. (2011). "Two visual strategies for solving the Raven's Progressive Matrices intelligence test." AAAI National Conference, San Francisco, CA

2. Kunda, M., McGreggor, K., & Goel, A. K. (2010). "Taking a look (literally!) at the Raven's intelligence test: Two visual solution strategies." 32nd Annual Conference of the Cognitive Science Society, Portland, OR, pp. 1691-1696