Agent design to solve Raven's 3x3 Matrices

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

Contents

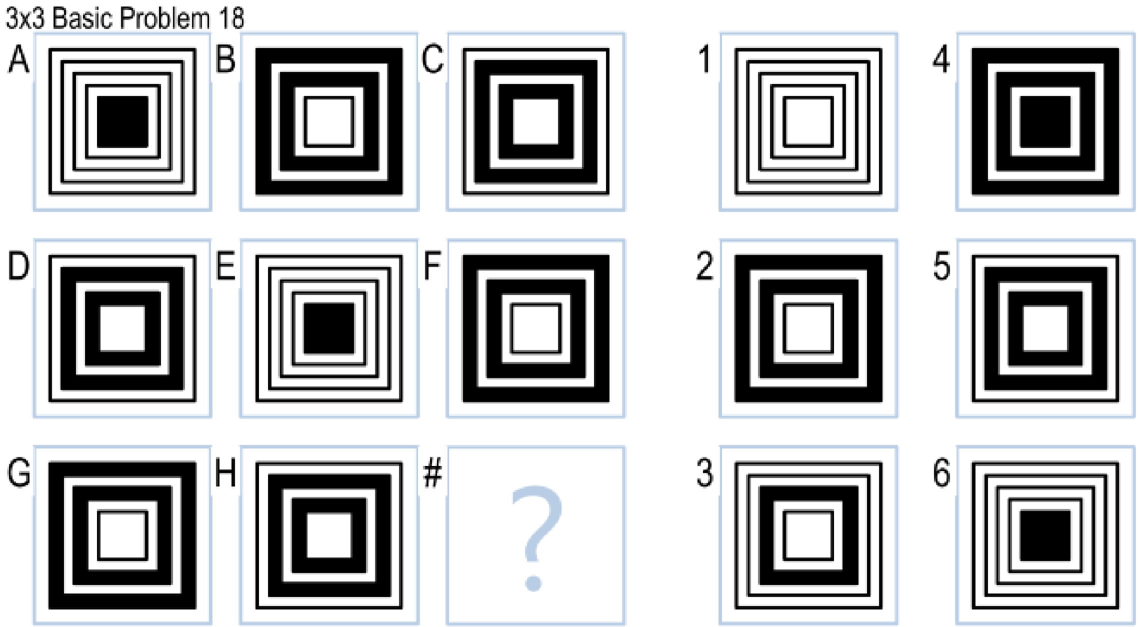
Reasoning used by Agent

This agent's reasoning is interesting in a sense that it does build on the previous implementations since by looking at test data for 3x3, I could see that it can use a very simple and yet combination of strategies that would enable agent to solve ALL the test problems correctly.

Agent uses mainly three strategies to solve the problem. Given a problem, agent first classifies the given problem from its features into three categories. Each category is solved by respective reasoning.

Strategy #1: 3-3-2 Strategy –

Here the agent looks for a pattern of 3-3-2. I will explain 3-3-2 pattern with following examples -



If the agent applies traditional strategy of finding corresponding objects and then changes between them, this problem looks intimidating.

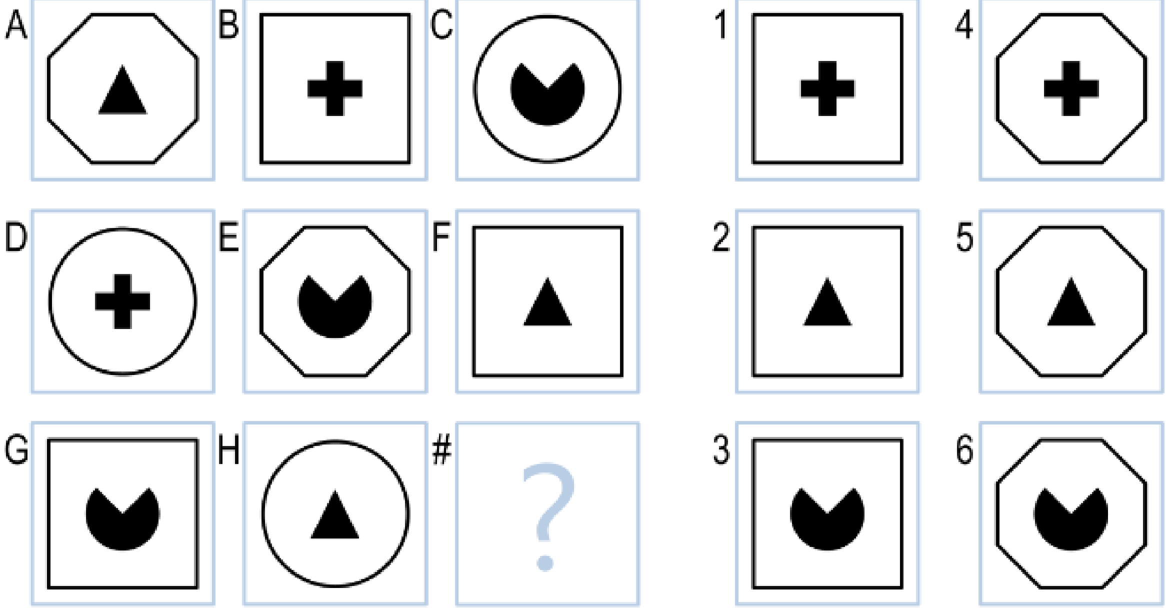
Now a different perspective could be -
1. Fig. B appears three times (fig. B, fig. F and fig. G)
2. Fig. C appears three times (fig. C, fig. D and fig. H)
3. Fig. A appears two times and hence answer should be the one which would make this count as three.

Agent makes use of this strategy and comes up with the correct answer.

The implementation for this is in terms of simple hash table where each figure is represented in terms of a string having all attributes and their values. Two figures having same attribute values would be kept in a single bucket with count incremented.

I want to illustrate this strategy with one more example since this operates at the level of Raven's object instead of Raven's figure.

3x3 Basic Problem 05



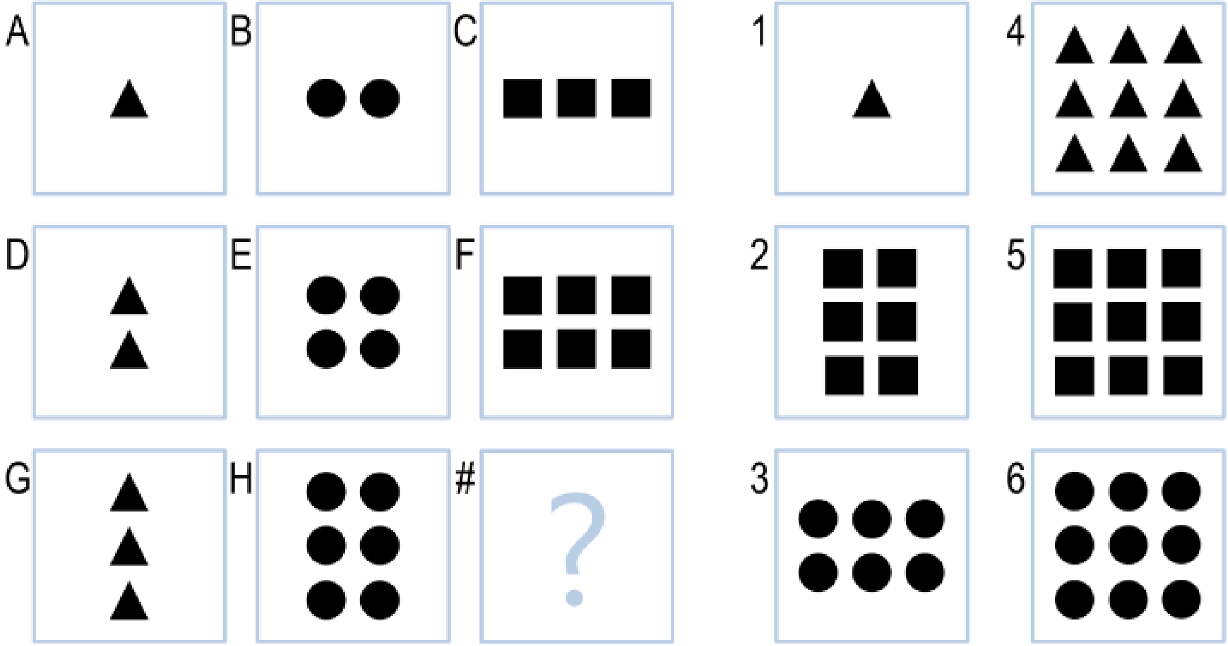
Let us make a table of counts of each object.

Octagon – 2 times
filled triangle – 3 times
Circle – 3 times
Square - 3 times
filled plus – 2 times
filles pacman – 3 times

To complete this problem, we need a filled plus and an Octagon. Hence agent chooses 4 as the correct answer.

Strategy #2 Pattern identification

3x3 Basic Problem 11



In this method, agent makes a 8x5 table where each objects attributes are stored column-vise.

Agent then identifies pattern using row 1-2-3 and 5-6-7. Identified pattern is then applied to fig. H to obtain target attributes.

These target attributes are then compared with choices given to get to the correct answer.

In the above example, five columns in a table would be – shape, size, angle, fill and #objects.

From #objects analysis, agent knows that the number is in multiple of A/D and hence applies same to G to get target #objects = 9. Similarly, it obtains target shape as square.

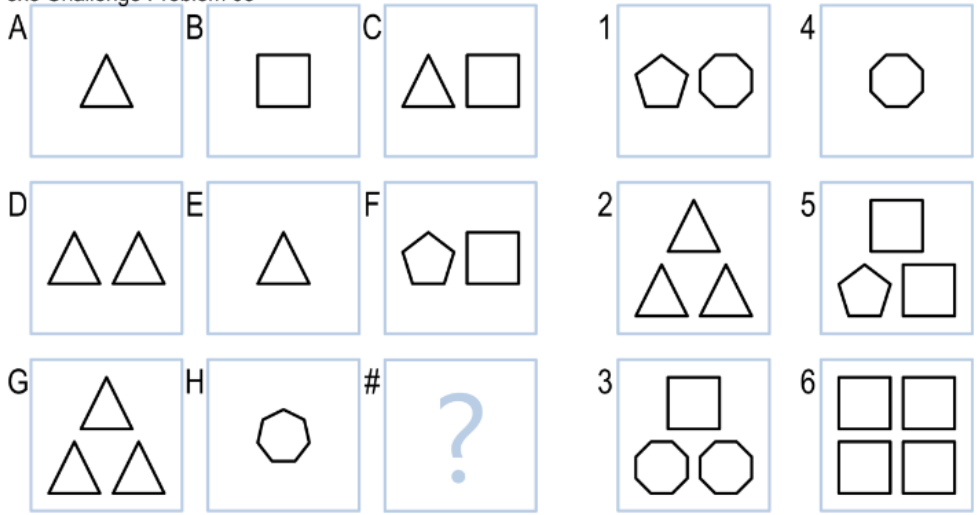
Interesting heuristic that agent applies here is to ignore structural attributes since in these class of problems, none of them requires structural attributes to be used.

Strategy #3 Single object based reasoning

For problems like #16, where each Raven's figure has only one object, agent takes this simplistic strategy where a similar table as described above is formed but without # objects column.
Similar target attributes are formed and used to come-up with the correct answer.

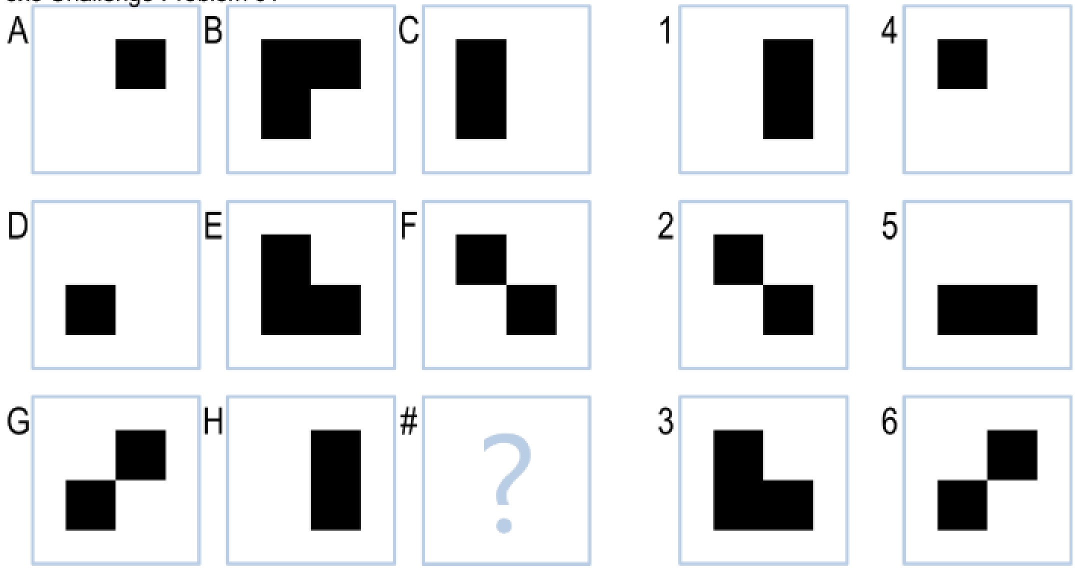
Agent’s mistakes

3x3 Challenge Problem 03



For this problem, agent requires to understand the number of sides in each Raven’s figure. With the current implementation agent simply is not to able to classify the given problem and hence is in no position to come-up the correct answer.

3x3 Challenge Problem 01



Since this agent does not understand the structural properties explicitly, it fails to deduce that fig.C/fig.F is obtained by subtracting fig. A from fig. B.
Hence fails to answer correctly.

Further Improvements

There is lot of scope from improvements since agent only used three to four concrete strategies. This agent could be made to understand structural relationships, number of sides in each figure.

Further, even now agent does not have the confidence level for each choice. If it chooses the answer then that is with 100% confidence and 0% for others. This does not mimic how humans think. Given more time, I would like to incorporate these improvements.

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

Because agent skips the part where corresponding objects were found in earlier projects, the runtime is reduced from $O(n^2)$ to $O(n)$ where n is number of Raven's objects in given figure.

As such agent does not take any significant amount of time to run. Agent makes a single pass over all the objects to build required data structures – it does not backtrack to understand why it made mistake. It does not take long times for any of the problems.

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 tried to mimic human reasoning in a way that it uses multi-layered reasoning each at different abstraction level.

First naïve reasoning where agent simply counts the number of objects is at highest level of abstraction whereas lowest level is where agent builds data tables and analyzes them to find out the patterns.

This agent tells us that we humans apply many number of heuristics to get to the answer. This agent also tells us that when humans are not able to come-up the answer, they use a rudimentary yet powerful approach of refutal-heuristics.

For problem #17 and #20, agent is not able to come-up with the reasoning to solve those. It then uses two heuristics –
1. Answer figure should not be the one which is already present in question.
2. Number of objects in answer figure should remain same - generally speaking.
With these two heuristics, it is able to solve both of these problems.

Please note that the above two heuristics are not accurate and mimics human cognition which does not understand problem completely.

I really enjoyed building this agent from the scratch and looking forward to image processing agent!