Dustin Jones

Ashok Goel

CS-7637

2016-10-30

Project 2 Reflection

One of the primary goals of Knowledge-Based Artificial Intelligence: Cognitive Systems (KBAI) is to learn about human behavior and to be able to translate that behavior into artificial processes. Just as humans have many different problem solving strategies, so too do those strategies translate to problem solving within problem solving agents.

Translating Project 1 into Project 2

I initially had decided upon building a verbal-only agent for both project 1 and project 2 as the verbal method seemed to most closely tie in with KBAI concepts. Project 1, with just 4 total relationship sets proved to be relatively trivial to model within the logic of the agent, with those relationship sets defined as ‘A-B’, ‘A-C’, ‘B-?’, and ‘C-?’.

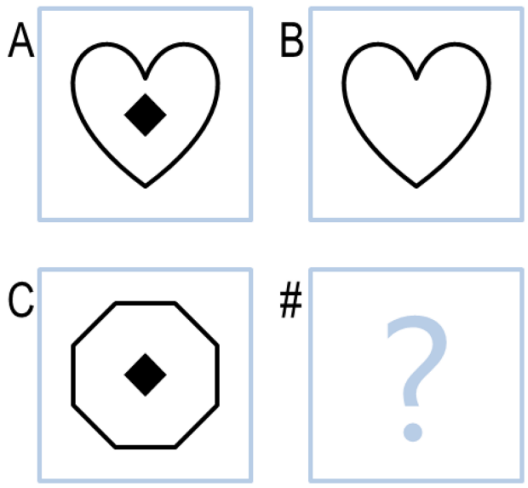

Fig. 1 Basic Problem B-11

My verbal algorithm relied upon 1-to-1 shape matching, generating a total set of combinations of object pairs, and selecting the appropriate pairs based upon metrics scoring the number and types of transformations to translate from one object to its match. Special placeholder objects were used to pair with shapes that were either added or deleted within each relationship set. For example, the set of combinations of object to object pairs within Fig. 1 for figure A to figure B would exist as it does in Table 1. First Pair 1 would be selected as there were no changes or transformations, Pair 2 would be skipped as the object 'Heart-A' was already claimed by Figure A in Pair 1, Pair 3 would be skipped as the object 'Heart-B' was already claimed by Figure B in Pair 1, Pair 4 would be selected as there are no remaining objects with which it could be paired, and Pair 5 would be skipped as 'Heart-B' was already claimed by Figure B in Pair 1.

	Figure A Object	**Figure B Object**
Pair 1	Heart-A	Heart-B
Pair 2	Heart-A	*Deleted*
Pair 3	Diamond	Heart-B
Pair 4	Diamond	*Deleted*
Pair 5	*Added*	Heart-B

Table 1 Object to Object matching for Fig. 1 A-B Relationship Set

While this approach worked well for 2x2 Raven's Progressive Matrices (RPM) scoring 12/12 correct for Basic problems and 11/12 for Test problems, it ultimately proved to be unscalable when translating to 3x3 RPM problems. Two different approaches were attempted when translating to 3x3 RPM problems: quadrant-based and row/column-based. Both of these approaches eventually ran into difficulties due to the inability to track relationships beyond simple object to object pairs and not being able to adequately build relationships between object to object pairs. This issue can most clearly be demonstrated by examining Fig. 2, where the

relationship between rows and columns relies upon realizing the additive nature of multiple objects.

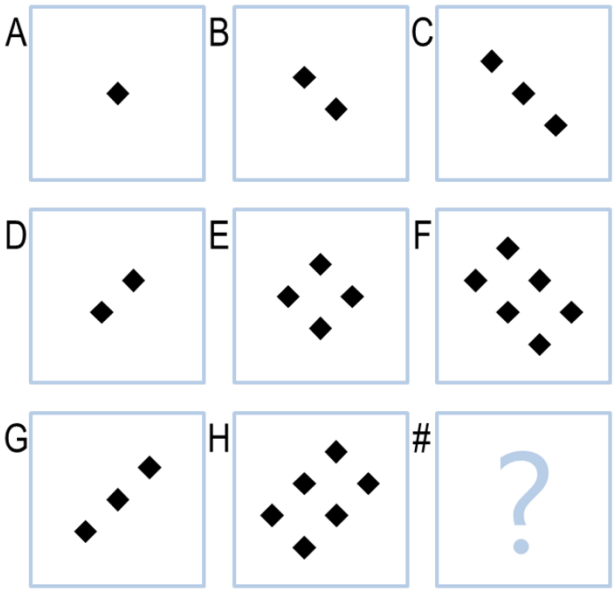

Fig. 2 Basic Problem C-03

The goal of the quadrant-based approach was to be able to reuse all of the existing 2x2 RPM logic, and build sets of transformations for each quadrant of the 3x3 RPM problems. Each quadrant was defined as a separate 2x2 RPM: 'A-B-D-E', 'B-C-E-F', 'D-E-G-H', 'E-F-H-?'. Unfortunately, this approach was not able to adequately capture the relationships across rows and columns, with relationships like A + B = C.

The goal of the row/column-based approach was to expand the relationships from just the object to object pairs to object triplets in order to track relationships as a total set. Each object triplet defined as 'A-B-C', 'D-E-F', 'G-H-?', 'A-D-G', 'B-E-H', and 'C-F-?'. While this approach certainly handled the main issue that the quadrant-based approach ran into, the challenge became how to not only identify and match object pairs across triplets, so objects in figures A, B, and C, but then mapping the figures and the transformations between each to corresponding objects in D, E, and F, and then subsequently identifying triplet-based

transformations. Eventually trying to match, identify transformations, and score and rank transformations proved to be exponentially more complicated with the selected data structures and design. At the end of my attempts to build a verbal 3x3 agent, I was able to achieve only 4/12 correct Basic problems with little chance of being able to expand the agent to the more complex problems without significant redesign and time investment.

Transitioning to a Visual Agent

With the experience gained from building a verbal shape-matching algorithm, I did not want to repeat the same mistakes. The agent needed to be far less granular, forgoing the process of recognizing shapes and transformations individually, and needed to be able to characterize figures from a macroscopic perspective. Without a background in image processing or computer vision, I did not want to rely upon defining a set of transformations (e.g. rotate, fill, duplicate, mirror, etc.) and then generating a set of comparative set of images to test and see if they matched a goal image. After some research, direct comparison of figures to discern relationships appeared to fit my initial criteria of a macroscopic solution. In particular, there are 2 metrics which have been demonstrated to provide the best results (Joyner et al, 2015) for this purpose: dark pixel ratio and pixel intersection ratio.

The agent begins by building a set of pixels for each figure that are not completely white (RGB values 255, 255, 255) and are touching at least one other non-white pixel. By including pixels that are gray and not just completely black (RGB values 0, 0, 0), it better allows the agent to match shapes where sharp angles are involved or where edge blurring may have occurred. By including only pixels that are touch at least one other pixel, the agent is better able to account for pixels that are compression artifacts or otherwise irrelevant to the problem. Once the agent has

broken the figures down into their component pixels, it is then able to quantitatively compare one figure to another. These sets of pixels are as dark pixels.

The agent calculates the dark pixel ratio (R_D) by comparing the sets of dark pixels (P) between two figures, calculating which figure has more pixels, and by how exactly how much more.

$$R_D = \frac{|P_B|}{|P_A \cup P_B|} - \frac{|P_A|}{|P_A \cup P_B|}$$

This allows for comparison from figure to figure pairs, and holds transitive properties to allow for comparison across rows and columns. The dark pixel ratio is most crucial for identifying patterns of progression, and establishing bounds for potential answers. As seen in Fig. 3, using the dark pixel ratio will allow the agent to reduce the set of potential answers to just Figures 1, 2 and 6 as they are the only figures that also preserve the pattern of ratios seen across all non-central (excluding 'E') figure pairs.

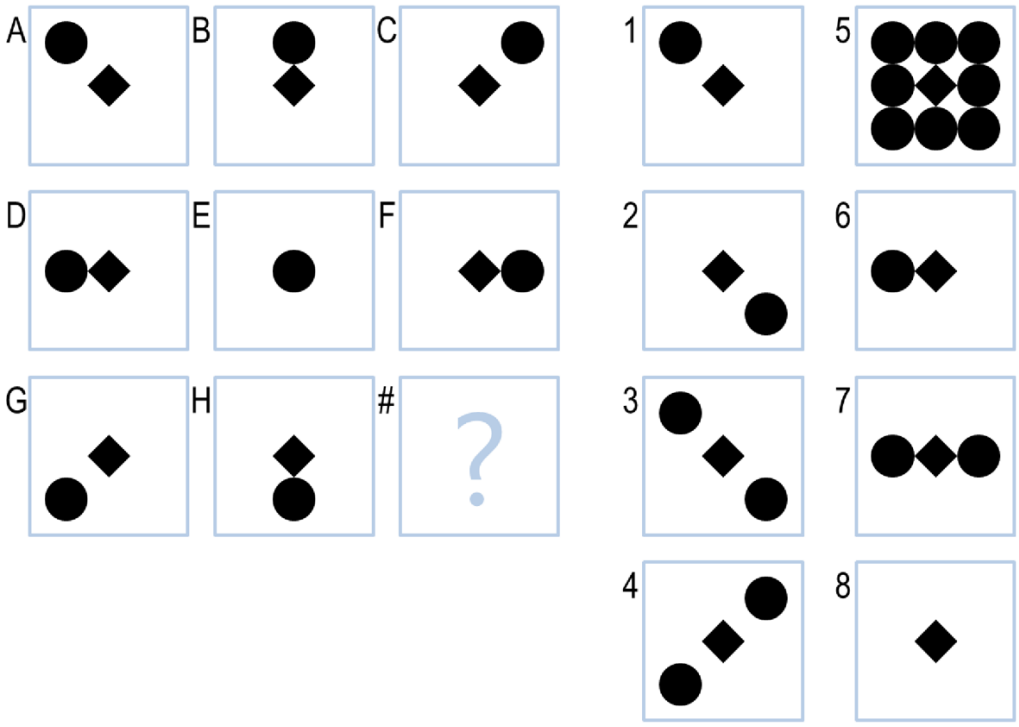

Fig. 3 Basic Problem C-07

The agent calculates the intersection pixel ratio (R_I) by identifying the total set of pixels (P) that exist in both of each of figure in a pair. This particular metric is better known as the Jaccard index.

$$R_I = \frac{|P_A \cap P_B|}{|P_A \cup P_B|}$$

This allows for the agent to compare figures by both similarity and dissimilarity, and can be especially useful for handling RPMs where there is either positional-shape repetition, or RPMs where there are distinct positions for objects across all figures such as in Fig. 3 where the filled circle is never located in the same position in any pair of figures.

Selecting an answer

Through trial and error, I determined that mathematical variance and mean are the most useful statistical methods for identifying patterns with both R_D and R_I. Using these methods, the agent is able to calculate key traits across rows and columns, identifying traits such as progressively increasing, decreasing, or static dark pixel ratio which would indicate transformations like growing a shape or adding shapes, shrinking a shape or removing shapes, and non-change of a shape or mirroring or rotation of a shape. Both sets of algorithms make heavy use of logic to identify figures which are suitable answers for a particular RPM. The logic for these problems are broken into two main pieces: identifying and classifying relationships between pairs of figures, and identifying and classifying relationships between rows and columns. All of these relationships have been identified incrementally, building and adjusting as new problems are encountered.

The agent uses both the dark pixel ratio and intersection pixel ratio separately, each using its own set of statistical comparisons for pattern recognition, each iterating through every available answer figure for a particular RPM. Each algorithm will select all answers that satisfy its respective conditions and tolerances and produce two resultant sets.

If there is an intersection of the resultant sets, the agent will sort the intersecting answer figures by combining and weighing the tolerance criteria for each algorithm, the weight determined by the cardinality of each of the resultant sets.

Given the below example, where for a given RPM the Dark Pixel Ratio (DPR) returns 4 figures and the Intersection Pixel Ratio (IPR) returns 3 figures:

- Dark Pixel Ratio Set (*D*)
    - Figure 1 – Distance from expected value = 3
    - Figure 2 – Distance from expected value = 5
    - Figure 3 – Distance from expected value = 7
    - Figure 4 – Distance from expected value = 9
- Intersection Pixel Ratio Set (*I*)
    - Figure 2 – Distance from expected value = 8
    - Figure 3 – Distance from expected value = 6
    - Figure 5 – Distance from expected value = 4
- Intersection of both sets
    - Figure 2
    - Figure 3
- Equations to calculate weights

$$P_D = \frac{|D \cap I|}{|D|}, \qquad P_I = \frac{|D \cap I|}{|I|}, \qquad W_D = \frac{P_D}{P_D + P_I}, \qquad W_I = \frac{P_I}{P_D + P_I}$$

$$P_D = \frac{2}{4}, \qquad P_I = \frac{2}{3}, \qquad W_D = 0.43, \qquad W_I = 0.57$$

- Equation to calculate weight (E_W) each figure using expected values (E)

$$E_W = (E_D \times W_D) + (E_I \times W_I)$$

- Calculated weighted distances from expected values
    - Figure 2 – $E_W = 6.71$
    - Figure 3 - $E_W = 6.43$

With this example, Figure 3 would be selected and returned as the answer to the RPM even though more than one answer satisfied the necessary conditions for each metric.

In the event that one of the algorithms (either the dark pixel ratio or intersection pixel ratio) returns a non-empty set, and the other returns an empty set, if the cardinality of the non-empty set is below a predetermined threshold, the set will be sorted using the only the distance from expected values of that particular algorithm. In the event that both algorithms return a non-empty set, but there is no intersection between those sets, the problem will be skipped.

Risks and Mistakes

The main risk and potential source of returning incorrect answers lies with the ability for the agent to guess at an answer when multiple answers satisfy the necessary metrics of the respective algorithms. While this may on the surface appear to be a problem, it allows the agent to behave in a more human-like manor. When humans encounter a problem they are unable to answer with absolute certainty, they will frequently follow a set script to determine if answering the problem incorrectly is a reasonable risk. Firstly, likely answers are identified out of all potential answers, reducing the probability of selecting an incorrect answer. Secondly, if they were able to reduce the set of answers enough to sufficiently minimize the probability of

selecting a correct answer, and that probability outweighs the risk of selecting incorrectly, then the human will select a guess from the likely answer set. This agent in fact follows that script exactly, and makes allowable mistakes.

At the time of this writing, the agent is correctly answering 12/12 Basic Problems C, and 9/12 Test Problems C with 3 incorrect in just over 13 seconds. Given unlimited time to improve the agent there are additional algorithms/metrics that can be added that would reduce the chance for incorrect answers. One of the potential methods for improving the agent would be to utilize a chunking strategy to slice each of the figures into pieces such that additional relationships could be discerned. The current algorithms only look at figures from a single perspective, the whole figure, which can result in erroneous results in certain situations such as Fig. 4, where neither the dark pixel ratio nor the intersection pixel ratio are able to identify clear patterns. Another method would be to add an algorithm to handle shape counting by identifying discrete sets of connected pixels that allows identification of counting relationships, and inter- and intra- figure relationships.

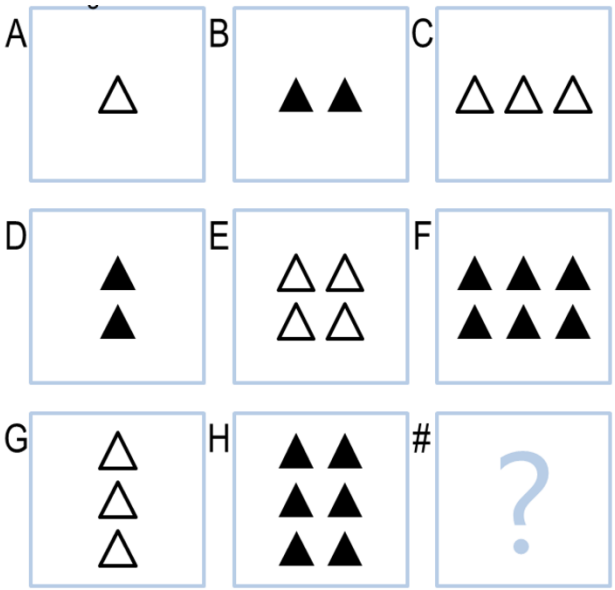

Fig. 4 Challenge Problem C-03

Without being able to access the Test Problems, the results would seem to indicate that the solution is not specific enough in certain circumstances, and either additional metrics/algorithms would need to be added, or there are better (but unknown) relationships that can be identified between figures.

Works Cited

Joyner, David A., Darren Bedwell, Chris Graham, Warren Lemmon, Oscar Martinez, and Ashok K. Goel. "Using Human Computation to Acquire Novel Methods for Addressing Visual Analogy Problems on Intelligence Tests." *Proceedings of the Sixth International Conference on Computational Creativity*. Provo, Utah. 2015. Web.