Imagery in AI
Assignment 1: Geometric Analogies

Deirdre Scully and Caitlin Snyder

Introspective Observations

When attempting to solve the geometric analogy problems ourselves, the identification and isolation of different shapes in the image is the first step in our solution process. Identifying shapes is one of the skills we commonly learn as children. We are able to identify distinct shapes even if they are overlapping, as in Problem 11. After identification, we determined that the next step in our problem solving process is the creation of generic logical rules such as "the shape inside of the bigger shape is removed." The rules applied to get from image A to B are then internally applied to our mental image of C. The matching result is then chosen as the correct answer. From these rules we can determine the different operations manipulating the images include: rotation, flipping, moving, adding, and subtracting shapes. Therefore, each of the problems can be divided into subsets of the operations applied to specific or relative shapes. Overall, the process we use to solve these problems involves more abstraction than the mostly visual process we noticeably used during the Raven's test, as discussed in Dawson et al. (2007).

While the isolation of different shapes within one image is easy for us to visually segment, it is a nontrivial task for computers. An algorithm to do so, such as edge detection, must explore every pixel neighboring a non-zero one in order to find the entire boundary of the shape. This will fail when shapes are overlapping, as they will be determined to be one shape together. Due to this complication, the design of our system attempts to find the solution without segmenting the image into individual shapes. This allows for certain operations such as rotation and flipping to occur in our algorithm only when the image has just one individual shape (e.g. Problems 2, 6, 12, 14). Other operations like moving, adding, or subtracting shape(s) are treated differently in our system. As we might visualize the moving of a specific shape from one location to another, to a computer it can be simply identified as subtracting then adding the same shape.

System Description

Our system operates under the theory that whatever changes happen between images A and B, they will be congruent to the changes between C and the correct answer. Congruence in our system is measured in terms of overall pixel changes. As a new pixel is added it is valued as +1 and as a pixel is removed it is valued as -1. As an example, if the image rule is that a dot is removed, and the same dot is removed from C to the correct answer, then the overall change in pixels between A-B and C-correct answer will be the same. This is shown in Problem 15, where the correct answer is 2, as seen in Figure 1. In many problems the exact same shape is not part of the rule. An example of this is Problem 9 with answer 3, shown in Figure 2. The general rule of change for this problem is that a smaller version of the image's current shape is added. In A to B this is a circle, but in C to the correct answer this is a square. Our theory of congruence here also applies, because the circle and square being a similar size mean they will have a relatively similar number of pixels added, as opposed to other answers such as 1, where two larger squares are deleted which includes a significantly larger number of pixels than the small square. It is also important to know that additions and deletions are recorded differently in our system as positive and negative pixels respectfully, so the correct change must also occur in the correct overall direction. Our system will represent the moving, translation or rotation, of a shape as a zero difference between the two images, since the same amount of pixels are added and subtracted. Therefore if a square is moved in A to B and a triangle is moved in C to the correct answer, both changes will have an overall change of zero pixels, even though a different number of pixels were updated.

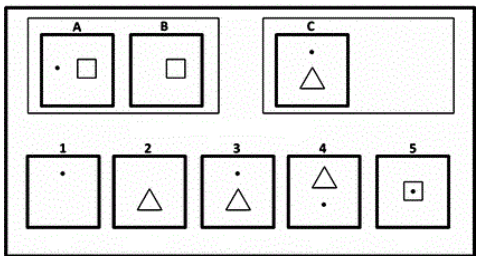

Figure 1: Problem 15

Our algorithm represents this congruence as difference maps between the images. This is done by "subtracting" B from A, i.e. removing the similar portions or pixels between

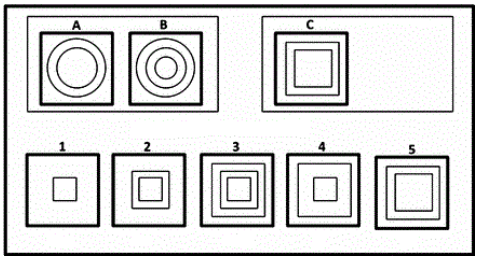

Figure 2: Problem 9

A and B so only the difference of pixels remain. An example of this can be seen in the section below. Next, the system performs the same subtracting process between C and all of the possible answers. This isolates all of the changes made between C and the answer options. The congruence idea is that the changes made from A to B and from C to the correct answer will be similar. In order to compare the changes, the system splits the difference maps into quadrants and averages the number of pixels that are changed within each. Finally, the number of pixels changed within each quadrant is compared between the A-B and C-options difference maps, i.e. the number of pixels changed in quadrant 1 of difference map A-B is compared to the number of pixels changed in quadrant 1 of difference map C-option. Whichever C-option image has the closest number of quadrant differences on average to the A-B quadrants, is chosen as the correct answer.

Our algorithm will also first attempt to find the correct answer if the only change is a simple rotation or flip of the only shape present. This is done by trying out flips and rotations on image A, then subtracting B from each, and if any difference map has a proportion of pixels greater than a set threshold, this is determined to be the correct and only rule. The threshold was found through experimentation and was set to 0.94 for rotations and 0.93 for flips. In other words, 94% and 93% of the pixels had to be the same in the translated image A and B. The found flip or rotation will then be applied to C, and again the answer options are subtracted from C. The option with the highest proportion of similar pixels is chosen as the correct answer. Note that due to the imperfection of the test pictures and rotation/flip functions, images that should match perfectly after a set degree of rotation or flip do not match exactly. Consider images A and B from Problem 6, as seen in Figure 3. Rotating image A counterclockwise 270 degrees would give the same image as B. However, as seen in Figure 4, the rotated image of A is a little higher than B which leads the program to concluding that this is not the correct rotation. To account for these problems, the rotation and flipping functions try different shifts in both the x and y direction of pixels before evaluating.

To generalize our algorithm into steps:

1. Try 45, 90, 135, 180, 225, 270, 315 degree rotations on image A:
   - For each rotation calculate the proportion of similar pixels between A and B,
   - If the proportion $>$ threshold, apply same rotation to C, calculate proportion of similar pixels for each answer option, rank answers from highest to lowest proportion value and skip all remaining steps.
   - If the proportion $\leq$ threshold, continue to next step
2. Try left/right and up/down flips on image A:
   - For each flip calculate the proportion of similar pixels between A and B,
   - If the proportion $>$ threshold, apply same flip to C, calculate proportion of similar pixels for each answer option, rank answers from highest to lowest proportion value and skip all remaining steps.
   - If the proportion $\leq$ threshold, continue to next step
3. Look at the difference maps:
   - Calculate difference map between A and B: (A - B).
   - Calculate difference maps between C and answer options: (C - option).
   - Calculate quadrant differences between A-B difference map and C-options difference maps: ((A-B) - (C-options)) for each option and image quadrant
   - Take the average over the quadrants for this final difference map(s).
   - Rank the answers based on the smallest final values, with the best answer having the minimum value.

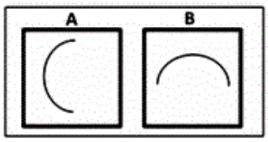

Figure 3: Problem 6

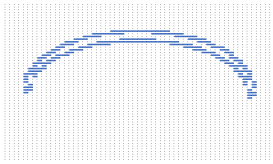

Figure 4: A rotated 270 degrees clockwise and B

Example Problem

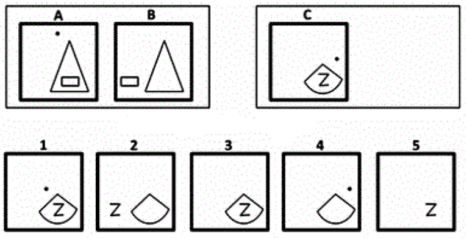

Figure 5: Problem 1

When faced with Problem 1, as seen in Figure 5, the system starts by trying 7 different rotations on A each 45 degrees apart, not including degree 360. Image B will be subtracted from each rotated A image, to generate the difference

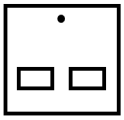

Figure 6: Absolute value difference between A and B

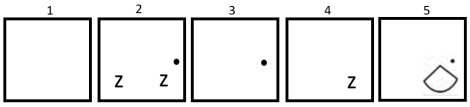

Figure 7: Absolute value differences between C and all the answer options

maps. In this problem no rotations are used, so no rotated A will match up with B and no difference map will be zero or close to. Therefore, the algorithm moves on to the next step 2. The rotation steps are repeated for 4 flips, over the x and y axis. Again here, no one flip can generate the change from A to B, so no difference map will be zero or close to and the algorithm moves on to the next step 3.

In step 3 the system isolates the changes between A and B which results in a difference image representation as seen in Figure 6. Next, the system isolates the changes between each of the options and C as seen in Figure 7. These C-option difference maps are all subtracted from the A-B difference map. The average pixel change is calculated for each answer across difference map quadrants. When comparing Figure 6 and the options in Figure 7, it can be seen that option 2 has the most similar pixel distribution, so will have the smallest difference when subtracted and in fact is the correct answer.

Results

When looking at the first answer ranked our system solves 8 of the problems correctly for 53% accuracy: 1, 2, 5, 6, 8, 9, 11 and 12. When considering the top two answers our system gets an additional two answers correct for 67% accuracy: 7 and 15. Finally, with top three answers, our system gets problems 3, 10 and 14 correct for a total of 13 out of 15 correct for 87% accuracy. Therefore the only problems that our system does not come close to the correct answers are 4 and 13.

Discussion

From our results and many different method attempts to solve this problem generally using only images, it is apparent how versatile humans are at this task compared to computers. Since computers process images on a by pixel basis, the smallest difference in two images can be recognized, whereas most humans do not even see individual pixels. An example of this is when a rotation or flip is applied to an image, if the shape is not perfectly centered the new shape can be offset by a few pixels as shown in Figure 4. To a human if two shapes are rotations or flips of each other, the exact pixel difference will make no impact on being able to recognize the shapes are the same.

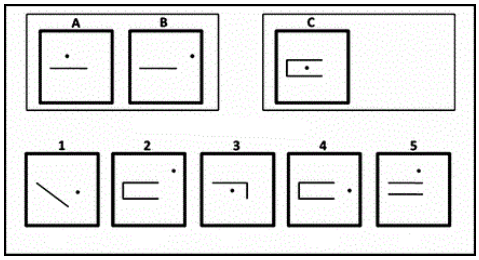

Figure 8: Problem 4

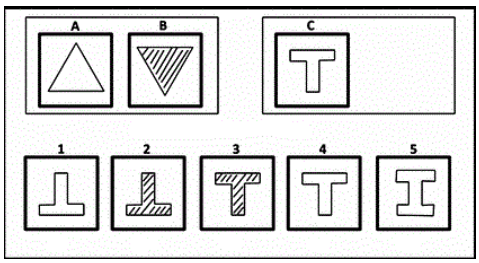

Figure 9: Problem 13

One of the main strengths of our system is that the algorithm is based on calculating difference maps, using subtraction. This method is very simple to implement and right away shows results that are correct over half the time. Other common methods, such as edge detection, can be very difficult to implement and still will not be guaranteed to work 100% of the time. Edge detection would not work when two shapes are overlapping, for example, whereas humans can easily differentiate between a circle and square even if they share a few pixels in common. The efficiency of a simple system with a descent baseline of correct answers, can be added to for improvements.

Another strength of our system is that is takes into account the different quadrants of the images. Many rules involve adding, deleting, or moving a shape from a particular location to another. Comparing quadrant differences can be valuable when, for example a square is moved from quadrant 1 to quadrant 4 in A to B. If C to the correct answer moves a triangle from quadrant 1 to quadrant 4, the pixel values added and deleted will be different since they are different shapes. However, comparing quadrants both difference maps will have many pixels added to quadrant 4 and subtracted from quadrant 1, so the correct answer is more easily found. This strength can also be a weakness if the rule does not involve a specific location. For example, if a square is inserted in quadrant 1 from A to B and the same square is inserted in quadrant 4 from C to the correct answer, the overall pixel changes will be equal, but the by quadrant changes will not be equal. This weakness is address to an extent in our system by taking the average over all the quadrants before final answer rankings.

The only problems that our system does not come close

to the correct answers are 4 and 13. For Problem 4 one of the reasons our system is always wrong is because of the by quadrant analysis discussed earlier. Problem 4, shown in Figure 8, moves a dot from its current location to the right in A to B and C to the correct answer. The dot originates in different locations within the A and C images, so also ends up in different locations within the B and correct answer images. These locations appear to be in different quadrants between A-B and C-correct answers, therefore the difference maps will not match up when analyzing by quadrant changes. The other reason this question is always wrong is because the change happening is a small dot, possibly represented by only 1 pixel value. If the image itself is imperfect, such as one of the lines is a couple pixels longer than the other, the change from moving such a small dot could be completely over powered by other small pixel differences. This is another good example at how most humans cannot even see such by pixel differences, but to a computer the pixels are all it can "see".

The other problem our system gets consistently wrong is Problem 13, shown in Figure 9. This illustrates perfectly one of the weakness of our system. Currently, rotation, flipping and the calculation of difference maps are all separate. In Problem 13, the image is rotated and added to. Since our system handles these two operations completely separately, this is answered incorrectly. An improvement to the system should include both rotating the image and calculating the difference maps.

Additional Methods

There were quite a few methods or small changes we attempted to implement into our system, but did not have improved results. There were implementations as simple as instead of averaging over the difference map quadrants, summing over them. These results were the same, so averaging was chosen to keep the quadrants a factor in the analysis. We also implemented percent change between the images instead of just the difference, but this method did not perform well. Because percent change involves dividing by the original map two issues arose. One is most pixels were divided by zero. The other is that when the rule involves adding and deleting shapes, those specific pixels get discounted since they are either valued as zero or are divided by zero, so no change is registered in any quadrant.

Another couple changes implemented involved removing the quadrants or analyzing with more than 4 quadrants. The strength of having the quadrants outweighed not using them, as discussed above, because it adds the extra information layer of not just overall pixel change, but also overall pixel change within a specific location of of the image. Using more than 4 quadrants was very sensitive to pixel changes, so certain shapes could have rather different difference maps from others. For example, if a large square was added into the bottom left corner in A to B, but a large triangle was added into the bottom left corner in C to correct answer. In this case, with 4 quadrants, most of the change would likely all be represented within the same quadrant. But if instead that one quadrant was split into many, the square pixels would likely be counted in more and different quadrants than the triangle pixels because of their difference in shape.

As discussed above, future improvements to the system should address Problem 13, in Figure 9. The weakness of our system represented by this problem is that the rotations, flips, and difference mapping are all separate options. If a rotation is found to match it does not attempt to flip or create difference maps. If we could better implement the rotation and flip image outputs, by fixing pixel offsets or making the image square again, then we could combine all these methods and recognize when an image has been flipped, then rotated, and another shape added. This would solve Problem 13, Figure 9, well since it is rotated then stripes are added.

Another future improvement that could encode additional spacial information is comparing the changes that occur in the images over the left and right halves as well as the top and bottom halves. Joining this analysis with the quadrants analysis would allow for more problems to be represented, such as Problem 4, Figure 8, where the difference across the left and right halves would be the same, but not across the 4 quadrants.

In all these methods discussed, the main issue is that for a computer to solve this problem it must be told every case to handle, whereas as humans we are learned to be extremely versatile at easily translating images to rules and logic and vice versa.

References

Dawson, M.; Soulires, I.; Ann Gernsbacher, M.; and Mottron, L. 2007. The Level and Nature of Autistic Intelligence. *Psychological Science.* 18(8): 657662.