Analogy with Qualitative Spatial Representations Can Simulate Solving Raven's Progressive Matrices

Andrew Lovett (andrew-lovett@northwestern.edu)
Kenneth Forbus (forbus@northwestern.edu)
Jeffrey Usher (usher@cs.northwestern.edu)
Qualitative Reasoning Group, Northwestern University
2133 Sheridan Road, Evanston, IL 60201 USA

Abstract

We use SME, a domain-general model of analogy, to solve a set of problems from the Raven's Progressive Matrices. SME is used in a two-stage mapping process which we have previously shown to be effective for solving geometric analogy problems. Each problem is drawn in PowerPoint and input into sKEA, our sketch understanding system. sKEA automatically computes qualitative representations of the drawings, using a spatial representation scheme motivated by research on human perception. We show that the representations generated by sKEA can be used with SME to solve the Raven's Progressive Matrices problems, without using any processes specifically designed for the task.

Keywords: Analogy, Qualitative Reasoning, Spatial Reasoning.

1. Introduction

How we represent our visual world and use these representations in spatial reasoning tasks is a deep issue in Cognitive Science. One way to explore this issue is to investigate sketch understanding, since it enables some early visual processing problems, particularly edge detection, to be sidestepped in order to focus instead on the problem of computing mid-level spatial representations. Our hypothesis is that many human spatial representations are qualitative (Forbus, Ferguson, & Usher, 2001). To explore this hypothesis, we have developed sKEA, a sketch understanding system (Forbus *et al*., 2004). sKEA computes a qualitative representation of a sketch drawn by a user. This representation can then be used in models of reasoning tasks. For example, in (Tomai *et al*., 2005), we used sKEA representations to solve geometric analogy problems, i.e., problems of the form, A : B :: C : ?, or "A is to B as C is to _?", where A, B, C, and the set of possible answers were sketches drawn by a user. We solved these problems using SME (Falkenhainer, Forbus, & Gentner, 1989), a computational model of analogy based on Gentner's (1983) structure-mapping theory of analogy in humans. We demonstrated that our domain-general model of analogy, when used with the spatial information provided by sKEA, was sufficient for solving a set of 20 analogy problems used in Evans' (1968) classic work.

While an important step, the problems used by Evans are not calibrated in detail against human performance at different developmental levels. By contrast, the Raven's Progressive Matrices (RPM) has been heavily studied and used in evaluations in recent years. RPM is a nonverbal intelligence test which measures individuals' eductive ability, i.e., their ability to find patterns in the apparent chaos of a set of visual scenes (Raven *et al*., 1998). The matrices come in two forms: 2x2 and 3x3 (see Figure 1; the authors have made up these examples to protect the security of the test). In each problem, individuals are presented with a matrix in which the bottom right entry has been left blank. Test-takers must pick the image that correctly finishes the matrix from a set of 6 possible answers for the 2x2 matrices, or a set of 8 possible answers for the 3x3 matrices.

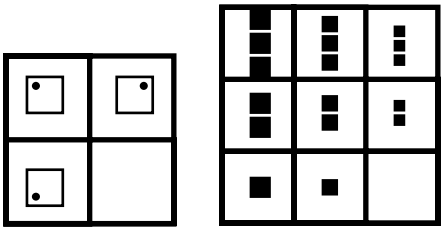

Figure 1: Examples of a 2x2 and 3x3 matrix problem.

We are particularly interested in the Standard Progressive Matrices (SPM) (Raven *et al*., 2000), which is divided into five sections, each progressively harder than the last. The A and B sections each contain 12 2x2 matrices, while the C, D, and E sections each contain 12 3x3 matrices. Section A involves simply filling in the missing part of an image. Later sections require more abstract reasoning.

Carpenter *et al*. (1990) modeled the performance of college students on the Advanced Progressive Matrices. Their system was able to match the performance of the most advanced students on these problems. However, their focus was on how students organized knowledge and applied problem-specific strategies. Among the shortcomings of their study, they mentioned: (1) they hand-coded the stimuli, rather than using an automatic perception system; (2) they identified the necessary rules beforehand and hand-coded them into the system, rather than having it discover the rules; and (3) their model might not capture the techniques and strategies used by younger students.

Our model addresses all three of these shortcomings. (1) We use sKEA to automatically compute visual representations. (2) We use only general processes to solve the Raven's problems; our system does not possess any prior knowledge about particular patterns or rules. (3) We are modeling performance by less advanced individuals;

specifically, we want to see if task-general processes suffice for modeling performance on sections B and C, two relatively easy sections in the SPM.

We start by briefly reviewing SME and sKEA. The spatial representations automatically constructed by sKEA are discussed next, including some specifically motivated by the RPM task. We then describe how SME is used in a two-stage mapping process to solve these problems. Finally, we describe the model's performance and related work.

2. The Structure Mapping Engine

Structure-mapping defines analogy and similarity in terms of a comparison process operating over structured representations, i.e., entities, attributes, and relations, including higher-order relations between relations. This process is governed by several principles (Gentner, 1983). A key bias is *systematicity*, i.e., mappings involving systems of relations, including higher-order relations, are preferred by people. SME implements this comparison process (Falkenhainer, *et al*., 1989). Given base and target descriptions, SME computes one or more *mappings*. A mapping consists of *correspondences* that describe how items (entities, statements) in the base description align with items in the target, *candidate inferences* representing conjectures suggested by the correspondences, and a *structural evaluation score* indicting the structural quality of the match. Candidate inferences are based on non-aligned structure in the base or target which is rooted in the correspondences of the mapping.

3. Sketch Understanding

sKEA, the *sketching Knowledge Entry Associate*, is the first open-domain sketch understanding system (Forbus *et al*., 2004). Most sketch understanding systems are limited to a narrow domain and require extensive per-user training. sKEA works differently, by splitting the perceptual load with the user. Users manually segment a sketch into *glyphs*, pieces of digital ink that represent the different objects in the sketch. Users can label each glyph with one or more conceptual categories from sKEA's knowledge base. sKEA automatically computes various qualitative spatial relations between the glyphs in the sketch, including relative position & size, and topological relations, such as whether one glyph is located inside another. Sketches can be segmented into *layers*. Relations are only automatically computed between glyphs on the same layer. sKEA combines the conceptual knowledge provided by the user with the automatically computed spatial knowledge to produce a qualitative, structural representation of a sketch.

In cases where complex precise spatial arrangements are required, hand drawings can be too difficult. Moreover, certain types of figures simply cannot be sketched. For example, consider the solid black squares in Figure 1. While the four sides of a square can be sketched, there is no good way to sketch a solidly black object. Consequently, we have added a second method for adding digital ink to sKEA. Users can copy lines and polygons drawn in Microsoft applications such as PowerPoint and paste them into sKEA. sKEA interprets the Windows Metafile format, which consists of a set of draw commands, to create either lines or polygons in sKEA's digital ink format.

4. Spatial Representation

Our spatial representation scheme is designed to be general-purpose. Many of the components have already been used in modeling other spatial reasoning tasks. The requirements of the RPM problems have led us to develop some new components, which we believe will prove useful in modeling future spatial reasoning tasks. We motivate the components via psychological evidence whenever possible.

Basic Elements

What objects should make up the basic elements in a representation of visual structure? One possibility would be to create an entity for every edge in an image. However, Treisman and Patterson (1984) found evidence that humans detect closure at an early, pre-attentive stage in the visual pipeline, allowing them to quickly reason about triangles at the attentive level without, apparently, having to resort to analyzing the triangles' individual edges. Consequently, we use closed shapes as entities in our spatial representation. Edges that do not form a closed shape are treated as separate entities. For example, a right triangle is represented as a single entity, while two edges forming a right angle are represented as two distinct entities.

We also include two basic attributes of closed shapes: their fill color, for solidly colored shapes, and their outline color. Ordinarily, every shape will have an outline color; a solidly black square will also have the outline color **black**.

Spatial Relations

Our representation contains two basic positional relations between elements: **left-of** and **above**. sKEA computes these relationships under certain conditions for pairs of glyphs that are disconnected and adjacent, i.e., there is no third glyph between them. Adjacency is determined by using a Voronoi diagram (Forbus *et al.,* 2003). Positional relations are only asserted when one glyph is directly above or directly beside an adjacent glyph; mixtures of positional relations between a pair of glyphs are not allowed.

When one object is spatially located within another, a different set of relationships must be applied. In studying subjects' memories for visual scenes in which a dot was located somewhere inside a circle, Huttenlocher *et al*. (1991) found that the memory contained two components. One was a quantitative component consisting of the dot's actual location inside the circle. The second was a qualitative component which encoded which of the circle's four quadrants the dot had been located in. When asked to recreate the scenes, subjects tended to allow their memory of the qualitative component to bias their memory of the exact location, moving the dot closer to the center of the quadrant in which it had been located. We interpret Huttenlocher *et al*.'s results (1991) as indicating that closed

shapes are capable of setting up a frame of reference. This frame consists of x- and y-axes running through the center of the shape. For simplicity, we currently assume that these axes align with the axes of the global reference frame, although this is not always true. When one visual object is located inside a closed shape, we first assert a topological relation stating that one glyph is inside the other. Then, the interior glyph's position relative to the axes of the exterior glyph's frame of reference is computed and encoded.

Shape Comparison

Clearly people are capable of distinguishing between different shapes based on detailed properties even when they are the same general type (e.g., two triangles). So while we treat closed shapes as entities, properties of their edges are still used in comparisons. People can compare novel complex shapes that are presented at different orientations. For tasks like the RPM, detecting what differences (rotation, reflection, and/or scale change) there are between two instances of the same shape is crucial.

Previous work on mental rotation provides valuable constraints. Shepard and Metzler (1971) demonstrated that when subjects were shown drawings of arbitrary shapes and asked to determine whether one shape was a rotation of the other, the time required to identify the shapes as identical was proportional to the angle between them. This result seems to suggest that shape representations are orientation-specific, and that subjects cannot compare two shapes to determine whether they are the same without first rotating their representation of one of the shapes to align it with the other. However, in the many studies that followed the original (cf. Shepard & Cooper, 1982), one fairly consistent finding was that, whether the shapes being compared were 2D or 3D, the time required tended to be proportionate to the degrees of rotation along the shortest possible axis of rotation. How could subjects know the shortest axis of rotation before they knew whether the shapes were the same? Shepard and Cooper (1982) suggested that, before mental rotating one of the representations, subjects identify corresponding parts on the two images and these parts guide the rotation. If this is true, then subjects must have access to some orientation-invariant representation that can be used to identify corresponding parts.

Thus, we propose that subjects use two representations for objects being compared, whether the objects are simple closed shapes or complex, three-dimensional structures. The first representation is a qualitative, rotation-invariant representation that describes how an object's parts relate to each other. If the object is a closed shape, this representation describes the shape's edges. It may include relations which specify the types of angles that exist between connected edges, as well as the relative lengths of edges, etc.

When comparing two shapes, we claim that subjects begin by using a structure-mapping process to align the shapes' qualitative representations, identifying the corresponding edges in each shape. Once corresponding edges have been identified, subjects cannot immediately conclude that the shapes are the same. The qualitative representations are relatively sparse, lacking specific information about the length and orientation of each edge. To determine that the shapes are identical, their quantitative representations must be compared. These representations are orientation-specific, so they cannot be compared without first mental rotating one of them to line up its edges with the corresponding edges in the other representation. Because corresponding edges are known, subjects can quickly compare one pair of corresponding edges to determine the shortest axis of rotation between them. They must then rotate all the other edges together along this axis, using some mental process that can take linear time.

Implementation Our shape comparison process starts by decomposing the shape into its component edges by identifying significant discontinuities in the curvature of its outline. We then build a qualitative representation where each component edge becomes an entity. We compare the qualitative representations using SME. The correspondences for the mappings it found are used to compute the quantitative difference. That is, we iterate over every pair of corresponding edges to ascertain whether the rotational difference between every pair is the same. This comparison process is an approximation of the mental rotation process described above, since we compare each pair of edges in isolation, rather than rotating the entire set of edges together over a common axis. When SME returns multiple mappings (representing multiple possible rotations between the shapes), we pick the shortest possible rotation.

Our comparison process can detect reflections and scale changes as well as rotations. We detect reflections over the x- or y-axis by reversing the order of edges in one of the representations, finding the corresponding edges, and checking whether a reflection over the appropriate axis would explain the orientation for every pair of corresponding edges. We also check whether one shape is longer or taller than the other along the x and y axes. This size comparison is facilitated by the previous orientation comparison because, if one shape has been rotated about 90 or 270 degrees, we know to switch that shape's x and y dimensions before comparing their sizes.

Encoding Given an RPM problem, the system begins by comparing shapes across all figures in the problem, creating a shape equivalency class for each shape. Membership in this equivalence class is used as an attribute that is encoded for every member of that class. For example, all rectangles in a sketched problem would be placed in the same shape class. (The attribute used is arbitrary, having no meaning outside that problem.) This enables objects to be aligned based on having similar shapes.

Transformations (i.e., rotations, reflections, and scale changes) are computed when images are compared. In RPM, this includes comparing two entries in the matrix or an entry with a possible answer. When comparing two entities, it begins by iterating over all shape equivalency

classes. For each class, it selects the first instance of that class it finds and uses it as a reference shape. It compares all other instances of the class in both entries to the reference shape to identify any transformations. All transformations are included in the representation of the appropriate entry as an attribute of that object. This process is performed on edges which are stand-alone entities, as well as closed shapes. Firstly, all such edges are assigned to one of the following shape classes: **CurvedEdge** and **StraightEdge**. Then, one straight edge is chosen as a reference, and all other edges are compared to its orientation and length.

Textures

Several RPM problems require distinguishing textures, so we implemented a rudimentary representation of textures in our system. This is the least psychologically constrained component of our model. Textures are detected by looking for parallel lines that are not part of a closed shape. When enough such lines are found, they are grouped together to form a *texture patch*. The outline of a texture patch is the total area covered by the set of parallel lines. This outline is scrutinized to see if there is a closed shape whose outline matches it. If such a shape is found, it is added to the texture patch to create a border for it; otherwise the patch is marked as a closed shape without a border. Texture patches can play the same roles as any other closed shape in the spatial relationships described above.

Encoding In RPM problems, there are cases where two shapes should align simply because they both possess a texture, and other cases where two shapes should align specifically because they possess the same texture, i.e., the lines that make up their textures are parallel. We captured these distinctions by including two attributes for any closed shape possessing a texture. The first is the **TexturedObject** attribute, assigned to all shapes with a texture. The second attribute is a texture class attribute, similar to the shape class attribute described above. That is, for a given RPM problem, all shapes possessing textures with parallel lines are placed into a texture equivalency class and assigned an arbitrary texture name for that class. Thus, in a comparison between entries, any two textured shapes will share at least one common attribute, but two shapes with the same texture will share two common attributes.

5. The Two-Stage Mapping Process

We use a variation of the two-stage mapping process from (Tomai *et al.*, 2005) to solve RPM problems. We first describe the process for 2x2 problems and then show how it is generalized to solve the more complex 3x3 problems.

Solving 2x2 Matrices

Recall that geometric analogy problems take the form A : B :: C : ?. This form can easily be applied to a 2x2 Raven's matrix by focusing on either the rows or columns. For example, consider the columns in the 2x2 matrix in Figure 1. We could solve the problem posed by this matrix by asking "The top-left entry is to the bottom-left entry as the top-right entry is to _?". By posing the question in this way, we are ignoring some of the information provided in the matrix, specifically the relationship between the two entries in the top row. However, one of our key insights about both the 2x2 matrix problems and the 3x3 matrix problems is that they include a great deal of redundant information. Much of the time, these problems can be solved while ignoring some, or even most of the information provided in the matrices.

Our strategy is to run two stages of comparisons. The first stage compares individual entries in the matrix. SME is used to compare the top-left and bottom-left entries in the matrix. Its mapping contains candidate inferences for expressions in the base that fail to align with the target and reverse candidate inferences for expressions in the target that fail to align with the base. In the current example, it produces one candidate inference saying that the dot is in the upper half of the square and one reverse candidate inference saying that the dot is in the lower half of the square. These inferences are used to construct a new representation, a representation of the differences between the two entries in the sketch. We refer to sets of differences as $\Delta(e_1,e_2)$, where e_1, e_2 are matrix entries or possible answers. Similarly, we use SME to compare the upper-right entry in the matrix to each of the six possible answers. Each of these comparisons also produces a Δ.

The second stage mapping process uses SME to compare the Δs found in the first stage. Thus Δ(*upper-left*, *lower-left*) is compared to the Δ(*upper-right*, *answer*) for each of the six possible answers. The correct answer should be the one whose Δ is most similar to Δ(*upper-left*, *lower-left*).

Scoring the similarity of Δs requires taking into account both those elements that align and those that fail to align. In our example, the correct answer would be a square with a dot located in its bottom right corner. Thus, both the Δs would involve a dot being in the upper half of the square versus the lower half. An answer that contained additional differences, e.g. where the dot also differed in its size or shape, would be less correct. An answer that contained fewer differences, such as the dot being in the same location, would also be less correct. Consequently, we measure similarity by calculating both the percentage of expressions in the base case (the differences between two matrix entries) that align with the target case and the percentage of expressions in the target case (the differences between a matrix entry and a possible answer) that align with the base case. We use the average of these two percentages as our similarity measure.

The entire process described above can also be computed based on the rows of the matrix. Because of redundancy, scoring the answers based on either rows or columns is usually sufficient. However, to ensure maximum accuracy, the system picks an answer based on the average of the scores computed based on rows and columns.

Solving 3x3 Matrices

It may initially appear to be the case that the two-stage mapping process described above is insufficient for solving 3x3 matrices. After all, the 3x3 matrices involve understanding a row of three entries, so a system based on comparing only pairs of entries should be unable to solve it. However, as noted above, Raven's matrices contain a high degree of redundancy. We have found that 3x3 problems can be solved by the same two-stage mapping process.

We solve 3x3 matrices by dividing them into four separate geometric analogy problems. As before, each of these problems involves finding differences between two matrix entries and comparing them to the differences between one matrix entry and each of the possible answers. The problems can be formulated as follows:

1) Row: middle : middle-right :: bottom-middle : ?
2) Column: middle : bottom-middle :: middle-right : ?
3) R-Prog: bottom-left : bottom-middle :: bottom-middle : ?
4) C-Prog: top-right : middle-right :: middle-right : ?

The first two questions are identical to the two used in the 2x2 matrix. They ignore the first entry in their respective rows or columns and consider only the mapping between the second and third entries. The last two questions focus on a single row or column. They look at the change between the first and second entries and compare that to the changes between the second entry and the eight possible answers.

We have concluded that all of the problems in section C of the RPM can be solved by asking one of the four questions given above. Many of them can be solved by asking more than one; our example 3x3 matrix in Figure 1 could actually be solved by asking any of these questions. However, most problems are not as easy as this one. Often, the correct answer will receive a poor score using the R-Prog and C-Prog questions. Therefore, it does not make sense to score each answer based on its average score across the four. Instead, we take the maximum score across the questions, assuming that the correct answer should receive a perfect or near-perfect score on at least one of the questions.

6. Performance on the SPM

Because the problems in the Standard Progressive Matrices are precisely rendered, we drew the figures in PowerPoint and pasted them into sKEA. The figures in each entry of the matrices were pasted into separate layers, and the layers were named so that sKEA could easily retrieve the set of glyphs and relations for each entry in the matrix. Each answer was also given its own layer.

We followed the normal sKEA strategy of relying on the user to segment a sketch into shapes. Each of the closed shapes was drawn in PowerPoint as a separate polygon. Lines that were not part of a closed shape were each drawn separately. Thus, when they were pasted into sKEA, they had already been segmented into the appropriate entities. sKEA automatically segmented closed shapes into edges, so that they could be compared to other shapes, and grouped parallel lines together to form textured closed shapes.

Note that we do not consider the problem of segmenting a scene into objects to be a trivial part of visual processing. In fact, we have previously explored automatic methods for decomposing the ink in a single glyph into edges and closed shapes (Lovett *et al*., 2007). However, for the present study, we are focusing on other perception problems.

Results

Our system was tested on the 12 problems in section B and 12 problems in section C of the SPM. Chance performance would be 2 correct answers in section B and 1.5 correct answers in section C. Our system correctly answered all 12 problems in section B, and 10 out of 12 problems in section C. Using the norms available on the SPM, we can compare our system's performance to human test-takers. According to the 1979 norms found in Table SPM2 (Raven *et al*., 2000), subjects who scored a perfect score on section B generally scored 52 or higher on the overall test (out of a total score of 60). Subjects who scored a 10 on section C generally scored between 49 and 52. This suggests that, within those sections, our system is performing at the level of test-takers who scored around 52 on the overall test, though we are not claiming our system could score as high on the other sections. According to tables SPM 9 and SPM 10 (Raven *et al*., 2000), a score of 52 is in the 50th percentile for individuals from the United States between the ages of 18 and 45. Thus, our system's performance on sections B and C appears to match the performance of the average American adult.

Discussion

While our system's performance was similar to the performance of typical adults on the SPM, the two problems that it missed were not the most difficult problems in section C, by human standards. The problems were from the middle third of the section, whereas the test is designed to steadily increase in difficulty throughout each section (Raven *et al.*, 2000). Thus, we believe the mistakes made by the system are based on limitations in our current model of perception. By considering the cause of these mistakes, we can gain insights into how the model may be improved.

The first mistake involves a single closed shape that differs in number of parts between entries. Unfortunately, our current shape comparison algorithm does not handle partial shape matches: it only identifies shapes with the exact same number of edges at approximately the same relative orientations. In the future we plan to use a more forgiving comparison algorithm, which can align some of the edges in two shapes and, based on that alignment, identify differences such as the addition or removal of other edges. The other mistake involves textures; here, the general **TexturedObject** attribute playes a part in misleading the system into choosing the incorrect answer. This result indicates that the system's simplified texture component needs to be fleshed out to better model human perception.

7. Related Work

Other current models of analogical matching include Mitchell's (1993) Copycat program and French's (1995) TableTop. However, these systems were designed primarily to work in a single domain, letter-strings for Copycat and table settings for TableTop. SME, in contrast, works on a general vocabulary that allows it to be used in variety of different domains (Forbus *et al*., 1997). LISA (Hummel & Holoyak, 1997) is a general model of analogy, but to our knowledge it has not been used with automatically generated representations.

8. Conclusion

This simulation provides evidence for two important points. First, it demonstrates that our spatial representation scheme encodes sufficient information for solving 22 out of 24 SPM problems that constitute two sections of the entire exam. This, along with independent evidence motivating them, suggests that our representations capture some important properties of human visual representations. Most aspects of these representations have been used in prior simulations (e.g., Tomai et al., 2005), with only two (frames of references inside objects and texture patches) added for this simulation. While further development is needed, especially in texture patches, this result suggests that the representations are on the right track. Second, we have shown that SME, in a two-stage mapping process, can be used to solve easy to mid-level SPM problems. No special-purpose mechanisms are required, lending support to the claim that SME models general processes of structural alignment in human cognition.

While our simulation overcomes the three limitations of the Carpenter *et al* (1990) model, it has a complementary limitation: it is not clear that our model can explain the strategies used by more skilled individuals on more advanced problems. Modeling performance on the advanced problems would have the dual advantages of demonstrating the generalizability of our model and allowing us to determine whether the model can accurately predict human errors, since humans make considerably more mistakes on these problems. We believe it may be possible to approximate the task-specific problem-solving strategies laid out by Carpenter using general analogical processes, although additional comparisons and mapping stages will be needed to deal with the more abstract spatial relationships. In the future, we plan to test out this approach.

Acknowledgments

This work was supported by NSF SLC Grant SBE-0541957, the Spatial Intelligence and Learning Center (SILC).

References

Carpenter, P. A., Just, M. A., & Shell, P. (1990). What one intelligence test measures: A theoretical account of the processing in the Raven Progressive Matrices test. *Psychological Review, 97*, 404-431.

Evans, T. (1968). A program for the solution of geometric-analogy intelligence test questions. In M. Minsky (Ed.), *Semantic Information Processing*. Cambridge: MIT Press.

Forbus, K., Gentner, D., Markman, A. and Ferguson, R. (1997). Analogy just looks like high-level perception: Why a domain-general approach to analogical mapping is right. *Journal of Experimental and Theoretical Artificial Intelligence* (JETI), 4, 185-211.

Forbus, K., Lockwood, K., Klenk, M., Tomai, E., and Usher, J. (2004). Open-domain sketch understanding: The nuSketch approach. In *AAAI Fall Symposium on Making Pen-based Interaction Intelligent and Natural*.

Forbus, K., Tomai, E., and Usher, J. (2003). Qualitative spatial reasoning for visual grouping in sketches. In *Proceedings of the 17th International Workshop on Qualitative Reasoning*. Brasilia, Brazil.

Falkenhainer, B., Forbus, K. and Gentner, D. (1986). The Structure-Mapping Engine. In *Proceedings of the Fifth National Conference on Artificial Intelligence*.

Forbus, K., Ferguson, R., and Usher, J. (2001). Towards a computational model of sketching. *Proceedings of the 6th International Conference on Intelligent User Interfaces.*

French, R. (1995). *The subtlety of sameness*. Cambridge, MA: MIT Press.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science, 7*, 155-170.

Hummel, J. E., & Holyoak, K. J. (1997). Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review, 104*, 427-466.

Huttenlocher, J., Hedges, L. V., and Duncan, S. (1991). Categories and particulars: Prototype effects in estimating spatial location. *Psychological Review, 98*(3), 352-376.

Lovett, A., Dehghani, M., and Forbus, K. (2007). Incremental Learning of Perceptual Categories for Open-Domain Sketch Recognition. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*.

Mitchell, M. (1993). *Analogy-making as perception: A computer model*. Cambridge, MA: MIT Press.

Raven, J., Raven, J. C., and Court, J. H. (1998). *Manual for the Raven's Progressive Matrices and Vocabulary Scales. Section 1: General Overview*. Oxford: OPP Limited.

Raven, J., Raven, J. C., and Court, J. H. (2000). *Manual for the Raven's Progressive Matrices and Vocabulary Scales. Section 3: The Standard Progressive Matrices.* Oxford: OPP Limited.

Shepard, R. N., & Cooper, L. A. (1982). *Mental images and their transformations*. Cambridge: MIT Press.

Shepard, R. N., & Metzler, J. (1971). Mental rotation of three-dimensional objects. *Science, 171*, 701-703.

Tomai, E., Lovett, A., Forbus, K., and Usher, J. (2005). A structure mapping model for solving geometric analogy problems. In *Proceedings of the 27th Annual Conference of the Cognitive Science Society*, Stresa, Italy, 2190-2195.

Treisman, A., and Paterson, P. (1984). Emergent features, attention, and object perception. *Journal of Experimental Psychology: Human Perception and Performance, 10*(1), 12-31.