Implicit Analogy-Making: A Connectionist Exploration

Douglas S. Blank

Indiana University
Computer Science Department
Lindley Hall 215
Bloomington, IN 47405

blank@cs.indiana.edu

Table of Contents

Introduction

Most everyone, including the experts, would agree that analogy-making is best defined as a process that creates a mapping between items in one domain (often called the *source*) to "similar" items in another (often called the *target*). Based on this definition, many researchers have attempted to model analogy-making by creating a mapping between two sets of data structures that represent the domains (see (Gentner, 1983) and (Holyoak and Thagard, 1989), for example.) In fact, most researchers treat an analogy as if it were *equivalent to* a mapping between the two domains. Let us refer to this type of modeling as "explicit analogy-making" as it represents the analogy via a set of explicit correspondences. An explicit analogy-making model might begin with the representations of the dynamics of heat-flow and water-flow, including relevant relations and attributes (see Figure 1) (Gentner, 1983). A set of correspondences, representing a mapping between all analogous items in the two domains, would then be produced by the model.

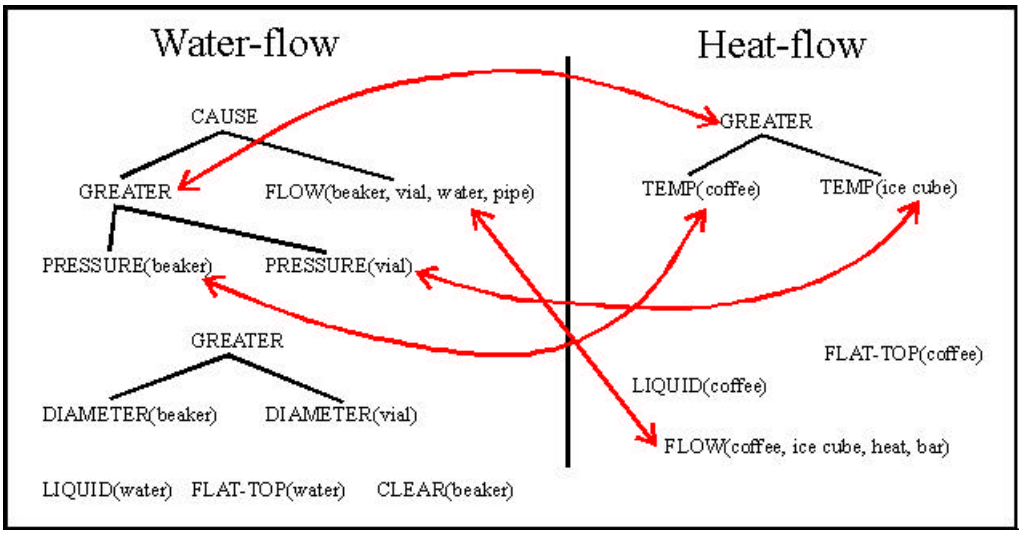

Figure 1. The representation of a typical "explicit analogy-making" problem. The red arrows indicate the explicit results of running the model (after (Gentner, 1983).)

An alternative viewpoint defines analogy-making as a type of *behavior*. In this view a system might, say, point to pairs of corresponding items one set at a time. Therefore the system is not required to produce an explicit mapping; the mapping is implied by the pointing, or more generally, by the behavior of the system. Let us refer to this type of modeling as "implicit analogy-making." Focusing on the behavior of making analogies rather than on explicit internal structures allows for the possibility of very different kinds of solutions to the analogy-making problem.

One analogy-making task that is defined completely in terms of behavior is the "Do this!" task posed by Hofstadter and French (see (Hofstadter and FARG, 1995) for much discussion of this task and other related ones.) French's model operates in a microworld representing a café table top, complete with saucers, spoons, salt shakers, etc. A boiled-down version of French's "Do this!" task was defined as follows. Consider Set #1 of Figure 2. Imagine an experimenter pointing to the triangle in the Source scene (the black hand) and saying, "Do this!" The subject's task is then to point to the "same" thing in the Target scene. Although simply stated, French has shown that the Tabletop domain can be full of subtlety (French, 1992). 1 Notice that the need for internal structured representations (such as those in Figure 1) is not specified by the task's description.

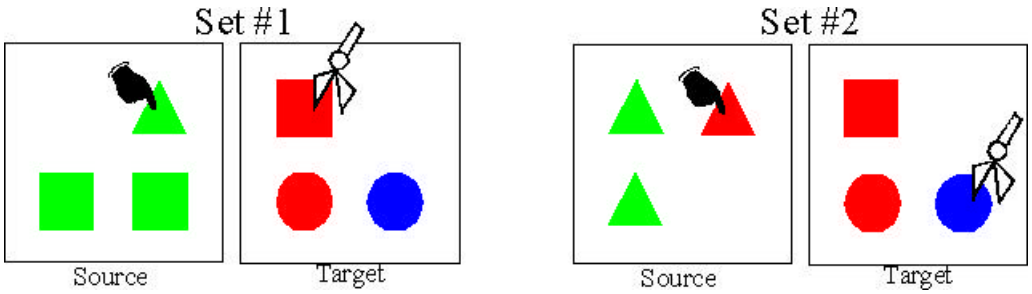

Figure 2. Simple geometric analogies designed in the style of Hofstadter and French's Tabletop model (Hofstadter and FARG, 1995).

Explicit analogy-making creates many assumptions about analogy and how one might model it. For instance, this view typically assumes that there exist pre-structured representations on which the mapping process operates. It is not at all clear that this assumption is cognitively plausible (Chalmers,

French, and Hofstadter, 1995). This view also suggests that not only can the mapping process be separated from the structure-forming process, but that it is also distinct from more general perceptual processes, such as recognition. This, too, seems unlikely. Although these issues have been partially addressed by Hofstadter and colleagues (1995), one question remains: How could a system *learn* to make analogies? In an attempt to answer this question, I have chosen to model the stripped-down "Do this!" task with a connectionist network.

Representation, Architecture & Training Procedure

I could have attempted to model the "Do this!" task in a connectionist network with representations similar to those in Figure 1. However, Hofstadter and colleagues have argued quite convincingly that pre-structured representations for analogy-making leave out an important portion of the problem: perception (Chalmers, French, and Hofstadter, 1995). A raw camera image would not work as a representation as it contains too much unstructured information. To provide the network with a slightly more abstract representation of a scene, a new representation was created. This section briefly describes the representation, network architecture and training method. For more details see (Blank, 1995) and (Blank, 1996).

Representation

A new representation was created based on Smolensky's Tensor Product method (Smolensky, 1990). A Tensor Product representation is a distributed set of activations formed by taking the outer product of a *role* vector and a *filler* vector (Smolensky, 1990). The resulting matrix is a distributed set of activations representing the *binding* of the role to the filler. In the new represention, the hard-edged categories of "role" were replaced with a vector representing "spatial location." The new representation, termed Iconic Representation (IR), was formed by taking the Tensor Product between a vector representing "what" an object is (e.g., "blue" and "triangle"), and a vector representing "where" it is found in a scene. The IRs of many objects can then be additively combined to represent scenes of many objects.

IRs do not explicitly represent *relations* between objects, but do explicitly represent *attributes*. For instance an IR of a **blue square above a red circle** would explicitly represent **red**, **blue**, **square**, and **circle** with **blue** bound to **square** and **red** bound to **circle**; it would not, however, represent **above** or **below** explicitly. For this reason IRs are incapable of representing hierarchical structure, or any kind of structure for that matter. This may be seen as a large weakness in the power of Iconic Representations. However, Hofstadter and colleagues have made convincing arguments against the use of pre-structured representations as inputs to analogical models (Chalmers, French, and Hofstadter, 1995). Structure is therefore left for the network to learn.

Architecture & Training Procedure

A 3-layer 343x49x110 recurrent connectionist network was created (See Figure 3). Standard backpropagation was used to adjust the weights (Rumelhart, Hinton, and Williams, 1986). The network was trained to make analogies in the following manner. First, an IR of a Source scene was created and placed on the input layer (see Figure 3, left panel). The "what" of an object was represented by a 6 unit localist encoding indicating the object's color (3 units) and shape (3 units). The "where" was represented by a 2x2 area of activation inside a 7x7 pixel scene. A representation of the object being pointed to was created by activating nodes (2x2) in the Pointer bank (7x7) corresponding to the object's location in the scene. The network was then trained to produce the representations of the *figure* and *ground* objects of

the Source scene on the output layer (Figure 3, middle panel). Here, the *figure* of a scene is defined to be the object being pointed to and the *ground* is everything else in the scene. Next an IR of the Target scene was placed on the same units that held the Source IR from the previous time step. Now, instead of placing activations representing an object being pointed to, the hidden unit activations (called the *context*) were copied from the previous time step into the Pointer bank (as with a simple recurrent network (Elman, 1990)) (Figure 3, right panel). The network was then trained to complete the analogy, that is, to produce on the output layer the analogous object as Figure. The remaining objects were regarded as Ground.

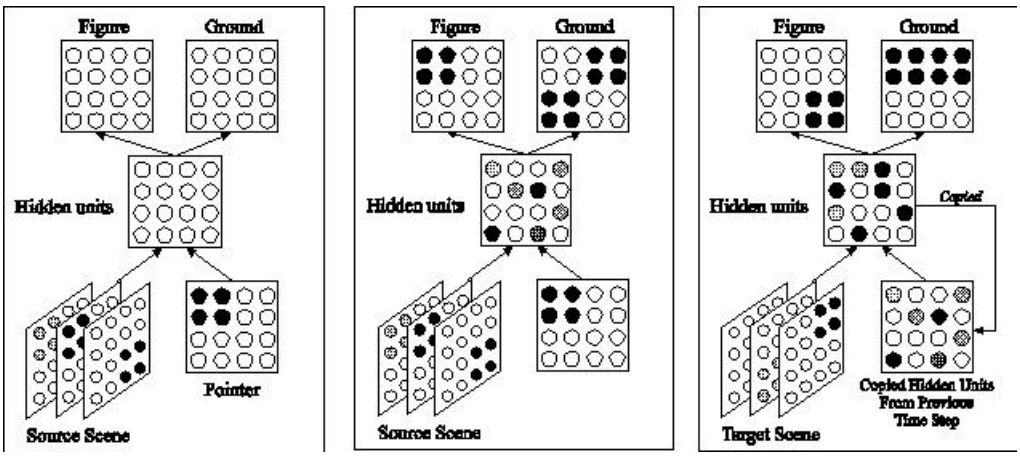

Figure 3. The network architecture and training procedure.

This architecture and training procedure had the following desirable features. Instead of having a bank of units for the Source IR and a separate bank for the Target IR they share the same connections, thereby making generalization easier. Likewise, the Pointer and the context activations also shared units. The Pointer and context activations filled analogous roles in the training of the network: looking at the "Source scene" and the "object being pointed to" is analogous to looking at a "Target scene" and "remembering the Source scene plus what was pointed to in it."

The iconic representations provide a flexible method to encode a wide variety of scenes without explicitly representing relations. The training procedure allows the network to learn to identify the Figure and the Ground in a scene when an object is being pointed to, and when a scene is being compared to a previously viewed scene. The network allows for a natural sharing of weights between items with similar function (between source and target scenes, and between pointing and context).

Experiments

This section describes three types of experiments made using the architecture, representation, and training procedure described above. During the training of each of the following experiments only three objects were placed in each scene, and their positions were limited to the four corners, although this need not be the case.

Experiment #1: Abstract category analogies

The first experiment was made to see if the network could learn to identify objects based on abstracted categories such as "the object that is different from the other two on the dimension of color." This was

tested by creating a Source scene that had a single object differ from the other two objects by either shape *or* color, and a Target scene that had one object differ by shape from the other two objects (in the Target scene) *and* another object differ from the other two by color (see Figure 2.) The object that was different in the Source (by either shape or color) was pointed to, and the network's goal was to select the object in the Target that differed "in the same way." For instance, a **green circle** differed from two **blue circles** in the same way that a **red triangle** differed from a **green triangle** and a **green square**. That is, they both differ on the dimension of color. Note that "same" and "different" were not explicitly represented but had to be learned by the network. Position played no role in this task.

Given these constraints there were a possible of 216 variations of the Source scene for each of the two types (e.g., "differ by shape" and "differ by color"). The Target scene was generated as follows. Three objects exactly alike were created. One object was picked and its shape changed. A different object was picked and its color changed. Given these constraints there are a total of 1,080 possible Target scenes and a total of 466,560 unique combinations of pairs of Source and Target scenes.

A database of 10,000 unique Source-Target pairs was generated randomly. The network was trained on 9,850 of these, leaving 150 to test for generalization after training. After being trained 100,000 times on instances of the 9,850 Source-Target training pairs, the network reached 100% performance. That is, the network had learned that if the pointed-to object in a Source scene differed from the other objects (in the Source scene) by shape then the object in the Target that also differed by shape was to be "pointed-to" (as with Set #1 of Figure 2.) To test the generalization ability, the 150 pairs of Source-Target scenes that were not used in training were tested and the network got each problem correct.

Experiment #2: Position-based analogies

The first experiment successfully showed that analogies could be made with the network based on abstracted categories based on shape and color (the "what" attributes.) Experiment #2 was designed to see if the network could ignore shape and color details, and make analogies based solely on position in a scene (the "where" portion of the representation). This was tested by training a network to select the analogous object in a spatial pattern.

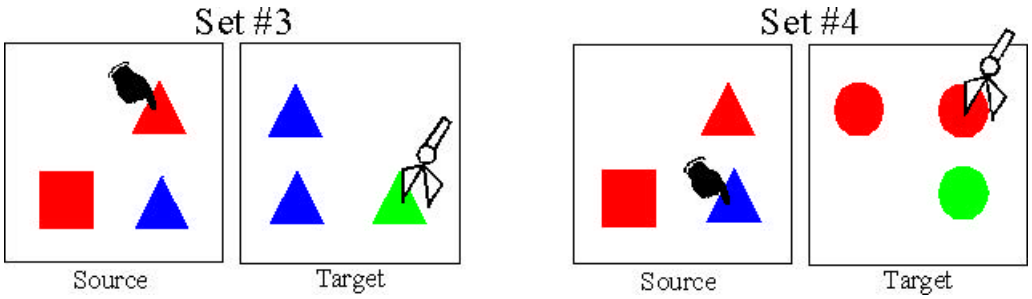

Figure 4. Samples of Experiment #2. The robot claw shows the correct choices as defined by the task. Set #3 has a Target scene which has been rotated 1 turn.

The Source scenes were generated exactly as were the Source scenes from Experiment #1. That is, one object was made to differed by either shape or color from the other two. The Target scene was created by taking a scene formed like the Source scene but rotating it clockwise 0, 1, 2, or 3 turns. For instance, the Target scene of Set #3 of Figure 4 was generated by rotating the Source scene 1 turn, and the Target scene of Set #4 was generated by rotating the Source scene 3 turns.

There were 432 unique Source scenes, however this time any one of the three objects could be pointed to

rather than just the one that differed. That gave 1,296 unique Source scenes, and 432 unique corresponding Target scenes for a total of 559,872 unique Source-Target pairs. A database of 31,000 random, unique Source-Target pairs was created, and 1,000 were saved for testing. Training went much more quickly than that of Experiment #1; the network reached 100% in 11,800 exposures. Obviously, this network did not need to be trained on nearly as many different analogies as were available. The network was then tested on the remaining 1,000 Source-Target pairs, and it got each one correct.

Experiment #3: Abstract category and Position-based analogies

This experiment was created to test analogies that required information from both relations and attributes. Target and Source scenes were created as before with the following changes. Target scenes were versions of the Source scene with a twist: they were either flipped horizontally, vertically, or rotated 0, 1, 2, or 3 times, or a result of flipping and rotating. Figure 5 shows a sample of a Target scene that was flipped about a vertical axes. This problem was more difficult than the previous two experiments: the network was unable to learn to do all of the types of twists no matter how the network was trained.

To test to see if this was just too hard a task for the network to learn, a series of "concept nodes" were added to the input. These nodes provided "hints" as to the type of twisting for each analogy (Abu-Mostafa, 1990). Four units indicated the amount of rotation (0, 1, 2, or 3 turns) and 4 units indicated the flipping (horizontal, vertical, or none.) Using these hints, 20,000 pairs of Source-Target sets were generated. The network was trained on 19,500, leaving 500 for testing. With the hints the network was then able to learn all of the tasks getting 100% correct after only 66,000 exposures. Of the remaining 500 testing pairs left, it got 484 correct (96.8%.)

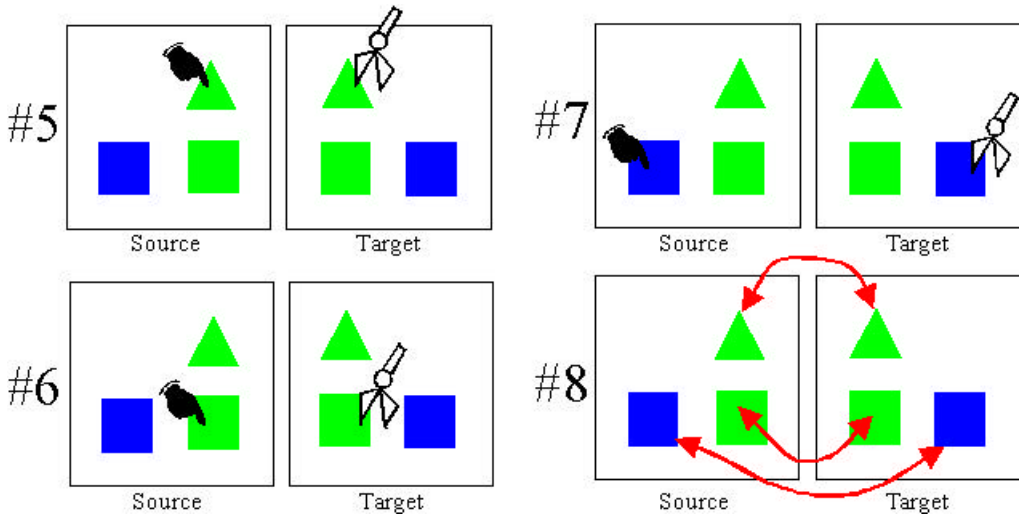

Figure 5. Samples from Experiment #3. Sets #5, 6, and 7 show all the parts of an implicit analogy. In contrast, Set #8 shows an explicit representation of the analogy.

Discussion

One might object to a model of analogy-making that first requires 100,000 trials of training; it is well-known that people can make analogies instantly and spontaneously. If one considers that these networks "knew" nothing initially but had to learn about figure, ground, location, color, shape, etc. then

that objection becomes unfounded. On the other hand, the networks, after training, "gush" their answers out with no temporal processing to speak of (Hofstadter and FARG, 1995). Although analogies often come to mind seemingly instantly a model that does not "actively explore" an analogy problem appears lacking. These objections (and other limitations and analysis) are discussed at length in (Blank, forthcoming).

By supplying Iconic Representations and training to make a basic Figure/Ground distinction, a simple network learned to perform these analogy-making tasks. Looking at the system's behavior we see that it must have learned to make distinctions based on self-created "categories." Although the analogies made by these initial networks were very simple, this methodology does expose a large, unexplored area available for future explorations in analogy-making.

References

Abu-Mostafa, Y. (1990) Learning from hints in neural networks. *Journal of Complexity* 6:192-198. [Citations in text: 1]

Blank, D. (forthcoming) Learning to see analogies: a connectionist exploration. PhD thesis, Computer Science, Indiana Univeristy, Bloomington, IN. [Citations in text: 1]

Blank, D. (1995) A distributed representation of multiple objects in a visual scene. In *Proceedings of the 1995 Midwest Artificial Intelligence and Cognitive Science Society Conference*. See also unpublished reports at http://www.cs.indiana.edu/hyplan/blank.html [Citations in text: 1]

Blank, D. (1996) Behavior-based analogy-making. To appear in *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society*. [Citations in text: 1]

Chalmers, D., French, R., and Hofstadter, D. (1995). High-level perception, representation, and analogy: a critique of artificial-intelligence methodology. In *Fluid concepts and creative analogies*, pages 169-194. Basic Books, New York, NY. [Citations in text: 1, 2, 3]

Elman, J. (1990). Finding structure in time. Cognitive Science, 14:179-212. [Citations in text: 1]

French, R. (1992). Tabletop: An emergent, stochastic model of analogy-making. Doctoral dissertation, EECS Department, University of Michigan, Ann Arbor. [Citations in text: 1]

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. Cognitive Science, 7(2), pages 155-170. [Citations in text: 1, 2, 3]

Hofstadter, D. and the Fluid Analogies Research Group (1995). Fluid concepts and creative analogies. Basic Books, New York, NY. [Citations in text: 1, 2, 3, 4]

Holyoak, K., and Thagard, P. (1989). Analogical mapping by constraint satisfaction. Cognitive Science, 13(3), pages 295-355. [Citations in text: 1]

Rumelhart D., Hinton, G. and Williams, R. (1986). Learning internal representations by error propagation. In Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Vol. I. (J. McClelland and D. Rumelhart, eds.) Cambridge, MA. Bradford Books/MIT Press. [Citations in text:

1]

Smolensky, P. (1990). Tensor product variable binding and the representation of symbolic structures in connectionist systems. Artificial Intelligence, 46, pages 159-216.
[Citations in text: 1, 2]

---

[1] Hofstadter and French's formulation of the problem was not nearly as limited as this; there was not a clear delineation between the "source" and "target" scenes, and there were many possible relationships (semantic and spatial) between objects. Also, Hofstadter and French's model is an explicit domain-mapping model. (Return to Text)