A Computational Model for Learning Structured Concepts From Physical Scenes

Erik Weitnauer; David Landy; Robert L. Goldstone
({eweitnau,dlandy,rgoldsto}@indiana.edu)
Department of Psychological and Brain Sciences, 1101 E 10th St
Bloomington, IN 47405 USA

Helge Ritter (helge@techfak.uni-bielefeld.de)
CITEC, Bielefeld University, Inspiration 1,
33619 Bielefeld, Germany

Abstract

Category learning is an essential cognitive mechanism for making sense of the world. Many of the existing computational models of category learning focus on categories that can be represented as feature vectors, and yet a substantial part of the categories we encounter have category members with an inner structure and inner relationships. We present a novel computational model that perceives and learns structured concepts from physical scenes. The perception and learning processes happen simultaneously and interact with each other. We apply the model to a set of physical categorization tasks while promoting specific types of comparisons by manipulating the order in which the scenes are presented. We find that these manipulations affect the algorithm in the same ways as human participants that worked on the same task. Both benefit from juxtaposing scenes of different categories – especially ones that are similar to each other. When juxtaposing scenes from the same category they do better if the scenes are dissimilar to each other.

Keywords: computational modeling; category learning; order effects; similarity

Introduction

Inductive learning of categories from a given set of examples is an essential ability for making sense of the world. Existing theories and models of concept learning have largely focused on categories where the category members can be described using feature vectors (Love, Medin, & Gureckis, 2004; Kruschke, 1992; Anderson, 1991; Nosofsky, 1986). Yet the ability to learn concepts that take the inner structure and inner relationships of members into account is essential for understanding human cognition. With the increased availability of large structured datasets like medical data, social network data or images and videos, there has been a growing interest in the machine learning community in statistical learning techniques on relational data (Getoor & Taskar, 2007). Within cognitive science, the application of Bayesian inference over grammatically structured hypotheses spaces provides the potential for modeling learning of relational concepts (Goodman, Tenenbaum, Feldman, & Griffiths, 2008).

The analogy-making community has made many contributions to algorithmically relating structured representations (Falkenhainer, Forbus, & Gentner, 1989; Hummel & Holyoak, 1996). Hofstadter (1996) and his group have put the focus of their fluid analogy models on the interesting interaction of perception and structure mapping, which is something that isn't central to any of the other models mentioned here.

We introduce a novel computational model that is inspired from the model of Goodman et al. (2008) and the fluid analogy algorithms of Hofstadter's group, including Phaeaco by Foundalis (2006). The computational model learns structured concepts in the domain of Physical Bongard Problems (PBPs), which are rule-based categorization tasks with a set of physical scenes that belong to two mutually-exclusive categories. PBPs are an interesting and challenging domain, since the physical scenes from which the categories have to be induced typically have a rich inner structure including relationships between the parts of each scene. Additionally, the feature-space of potential categories is large and initially unknown to the learner.

Our model uses basic physical feature-detectors to perceive the features of the objects in each scene. Based on these perceptions, the model constructs structured rule-based interpretations of the scenes, gradually focusing on the most promising ones. The perception process and the process of hypothesizing about the correct categorization rule interact with and constrain each other.

In previous work (Weitnauer, Carvalho, Goldstone, & Ritter, 2014), we already used PBPs to look into the benefits different types of comparisons between members of the same or different categories have on learning performance. During training, we presented the PBP scenes in pairs and manipulated whether the scenes within each pair were from the same or from different categories, as well as whether they were similar or dissimilar to each other.

There is a strong body of evidence in psychology literature that these different kinds of comparisons provide different amounts of information. Typically, more variance in the irrelevant features possessed by examples within one category will make the task of telling them apart from the relatively stable, defining features of the category easier, leading to better learning (Medin & Ross, 1989; Rost & McMurray, 2009). When comparing instances from different categories, typically the opposite is true. Comparing relatively similar instances from two categories has the advantage of decreasing the likelihood of spurious differences being chosen as the basis for discriminating the categories and it additionally increases between-category contrast and discriminability (Carvalho & Goldstone, 2013; Birnbaum, Kornell, Bjork, & Bjork, 2012; Kang & Pashler, 2012).

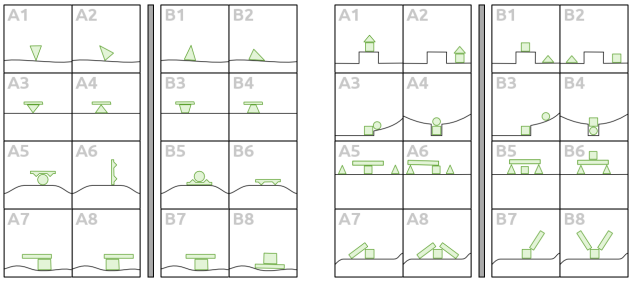
In this paper, we first describe the problem domain and de-

sign of our computational model and then report results from applying the model to the same tasks we gave human participants in our earlier studies.

Physical Bongard Problems

We use Physical Bongard Problems (PBPs, see Weitnauer & Ritter, 2012) as our problem domain, a variation of the classical Bongard problems described by Hofstadter (1979). Each PBP consists of two sets of 2D physical scenes representing mutually-exclusive concepts that must be identified. The scenes of the first concept are on the left, the scenes of the second concept on the right side. Figure 1 shows two example problems. What makes PBPs particularly interesting as a domain for concept learning is the inner structure in the scenes and their open-ended feature space. People do not know in advance which features a solution might be based on (or indeed what the features are), and while some of the problems rely on features that are readily available such as shape or stability, others rely on relationships between the objects or require the construction of features as a difficult part of the solution (e.g., the direction a particular object in the scene is moving in).

We grouped the 16 scenes of each PBP into 4 similarity groups, such that scenes within a group are on average more similar to each other than to scenes of other groups. Figure 3a shows PBP 24 with the similarity groups arranged in rows. Both the participants from the previous study and the model were only allowed to see two scenes at a time while solving a PBP, as shown in Figure 4.



(a) PBP 08 - stability (b) PBP 20 - square support

Figure 1: The task in Physical Bongard Problems is to identify the two concepts A and B. The concepts labels are not shown during a study.

Computational Model

Each scene in a PBP can, by itself, be interpreted in many different ways. The upper left scene in Figure 1b could be described as "two objects in the middle of the scene", as "a triangle on top of a square", as "a square supporting another object", etc. Which of these interpretations constitutes a solution to the PBP depends on the context set by all the scenes in the problem. In our model, we will refer to these interpretations as *hypotheses*. A hypothesis that matches all scenes from one side of a PBP and matches none of the scenes from the other side is a *solution*.

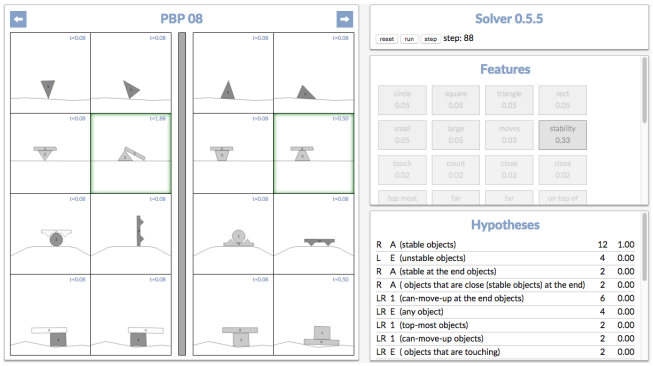


Figure 2: Our computational model including its source code is publicly available at `http://graspablemath.com/pbp-model`. It can be run interactively inside the Chrome browser.

Model Design Decisions

1. Conjunctive hypothesis space. We restrict hypotheses to conjunctions of object attributes, group attributes and object relationships. This means that extending a hypothesis always makes it more specific.

2. Perception-driven. The processes of perceiving features on objects and constructing and refining of hypotheses happen at the same time and influence each other. Initially, the algorithm does not know anything about the objects in the scenes except their positions and geometrical outlines. It perceives features step by step and builds scene descriptions based on those perceptions.

3. Probability-based decisions. The algorithm uses the information from previous hypothesis-scene matches to estimate how probable it is for hypotheses, features and objects in the current scene to be part of a solution. Based on those estimates, it makes a stochastic decision on what to perceive or check next.

4. Local actions. The algorithm can only perceive features and check hypotheses on the currently visible scene pair. Therefore, estimating the probability of a hypothesis needs to take into account that different hypotheses will have been checked on a different number of scenes. Although the algorithm keeps track of all hypotheses that are created, typically just a few hypotheses are actively explored. Only when a promising hypothesis turns out to be wrong, attention is shifted to others.

Model Behavior

We'll give a brief description one run of the model here. See Figure 2 for a screen shot of the model in action.

When the model starts working on a PBP, the first step is to load all the scenes which are provided as SVG images into memory. The objects and the ground in each scene are represented as polygons that describe their outline, which act as the basis for a physics engine that is used both to simulate how the scenes will unfold and to perceive physical object features like stability.

Initially however, the model knows nothing about the objects beside their existence and starts gathering information

about the objects in the first visible scene pair. It selects features, like *large* or *stable*, and objects to perceive the features on. After a new perception was made, a corresponding scene description, a *selector*, is created (e.g., "large objects"). This selector is then applied to both scenes in the currently visible scene pair, potentially resulting in a number of objects in both scenes that match. The match results and the selector are both captured in a *hypothesis*, which represents a potential solution or potential part of a solution.

After some perception steps, the model switches to the next scene pair. It can now continue to perceive features on the new objects or check existing hypotheses on the new scenes to gather additional evidence about their likelihood. The third available type of action is to combine existing hypotheses to build more complex ones. For example, "large objects" and "small objects on top of any object" can be combined into "small objects on top of large objects".

The model stops as soon as a hypothesis was checked on all scenes and is a solution, in fact matches all scenes from one side and none of the scenes from the other side. The search is aborted after a fixed number of actions if no solution was found until then.

During a run of the model, it determines the type of the next action by randomly drawing from a fixed multinomial distribution. The elements the chosen action is acting on are determined stochastically based on the information from all hypothesis–scene matches done so far. More promising hypotheses will be checked first; objects and features that play a role in promising hypotheses will be picked with a higher probability for perceiving further features.

Implementation Details

Scenes hold physical representations of their objects. A physics engine is used to both predict how the scene unfolds over time and to perceive physical features on objects. Stability, for example, is perceived by observing how much an object moves after poking it.

Objects keeps track of all perceptions that were made on them. *Groups* are sets of objects and contain all matching objects of one or several selectors like "square" or "any object".

Selectors represent a specific, structured interpretation of a scene by describing what to look for. When applied to a scene they select a subset of the scene's objects. If the subset contains at least one element, the selector and the scene 'match'. Selectors are conjunctions of percepts, like "small $\wedge$ hits (big $\wedge$ rectangular)" or "square $\wedge$ count = 2". *Hypotheses* keep track of which scenes matched or mismatched a specific selector.

Features and Percepts. The model currently has 33 inbuilt feature-detectors, including detectors for static object properties like size and shape, physical properties like stability and movement, spatial relationships like 'left-of' or 'close' and group attributes like object count. Each feature-detector can perceive its feature on any object or object group and the resulting percept stores the perceived value of the feature as a membership degree between 0 and 1 (e.g., A is almost left-of B). Currently, the algorithm uses a fixed threshold of 0.5 to decide whether a feature is considered active or not (e.g, A is left-of B or not).

Actions. At each step, the model chooses one of three actions by sampling from a multinomial distribution. It perceives a feature on an object with $p = 0.6$, it checks an existing hypothesis with $p = 0.3$ and it combines two hypotheses with $p = 0.1$.

In the *perceive feature action* the algorithm does one of two things with equal probability. It either first selects an object or group from one of the current scenes and then selects which new feature it should perceive on the object or group. Or, it first selects a feature and then selects a new target to perceive the feature on. In case a relationship is perceived, a target object for the relationship is chosen, too. If the action results in a new percept, it schedules a *create hypothesis action*, which turns the percept into a corresponding hypothesis that is then checked in the next step.

The *check hypothesis action* selects one of the hypotheses that was not checked on the current scenes yet and checks it. Finally, the *combine hypotheses action* selects an object and merges two hypotheses that include that object into a new hypothesis that is then checked in the next step.

The timing of when to switch to the next scene pair is based on how promising the current hypotheses are. If one is likely to be the solution, the algorithm moves to the next scenes earlier so that hypothesis can be checked against the remaining scenes. If none of the hypotheses is particularly promising, it keeps perceiving the current scenes longer.

Probability Estimation

Whenever the model is selecting a hypotheses to check or combine, or an object and feature to perceive, the choice is made stochastically based on previous results of matching hypotheses against scenes. While seeing, e.g., a lot of circles is not very telling in itself, if a "circle objects" hypothesis captures that those circles are only found in scenes on one side, it should give some credibility to circles playing a role in a solution to the problem.

We represent all hypothesis–scene matching results in a match matrix M. The columns correspond to hypotheses, the rows to the scenes of the PBP and each element $m_{i,j}$ is set to 1 if hypothesis h_j matched scene s_i, to 0 if it didn't match and is blank if it was not tested on the scene, yet.

Hypotheses We estimate the probability of an hypotheses being the solution or part of a solution using the following heuristic. In case h_i can be a solution given all match results so far (all tested scenes that matched were from one side and all that didn't match from the other), we set

$$P(h_i|M) = 0.5^{blank} P_0(h_i),$$

where *blank* is the number of scenes on that h_i was not tested on so far and $P_0(h_i)$ is a measure of complexity for h_i and acts as a prior. In practice, this heuristic ensures that the more

scenes an hypothesis is successfully checked on, the higher the estimated probability of it being a solution gets.

In case h_i can't be a solution but still might be part of a combined, conjunctive solution (it only mismatches scenes of one side), we set

$$P(h_i|M) = 0.5^{blank+S/2+incomp} P_0(h_i),$$

where $S/2$ is a fixed penalty set to half the total scene count and *incomp* is the number of scene matches / mismatches that are incompatible with h_i being a solution by itself. In the special case of a hypothesis matching *all* scenes on both sides and containing base-level-features only (shape and size in our case), we set $incomp = 0$. In practice, this makes hypotheses that are close to being a solution more probable than ones that are farther off, while accounting for the special situation in which the "same" object is showing up in each scene. Finally, in case a hypothesis can't be part of a solution (it mismatches scenes from both sides), its probability is set to 0.

Objects The probability that any particular object plays a role in a solution is estimated based on the probabilities of all current hyptheses that select that object.

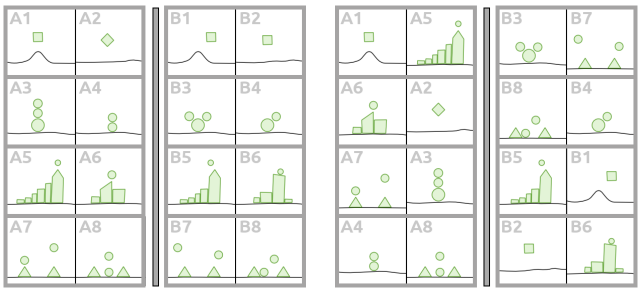
$$P(o|M) = P_0(o)Z \sum_{h \in H_o} \frac{1}{N_o(h)} P(h|M),$$

where o is an object, H_o is the subset of hypotheses that are know to select o, $P(h|M)$ is the estimated probability of hypothesis h and $N_o(h)$ is the number of objects that h selects in the scene o belongs to. $P_0(o)$ is the prior probability of the object and Z is a normalization factor that ensures the activities of all objects in a scene add up to 1. The relative priors for objects depend on the attributes that were perceived on each object so far and give more probability to objects that are initially moving or top-most in a scene. This heuristic estimates the probability of an object by equally distributing the probability of all hypotheses to the objects they are known to select. The algorithm will by design always consider an "any object" hypothesis, which ensures that each object in the scene is selected by at least one hypothesis.

Features The estimation of the probability that a feature is used in the solution is identical to the object formula above, with one difference. We add a small fixed term to the sum such that features that are not used in any of the current hypotheses still have a chance of being selected for perception. This accounts for the case that the solution is not among the current hypotheses. The relative priors for features are 3.0 for shape and size attributes, 2.0 for movement and stability and 1.0 for all others – reflecting that humans are more readily perceiving and encoding some features than others, as well as the expectation that PBP solutions will more often use such features.

Experiment

In the original study with human subjects on PBPs, the participants were presented a sequence of scene pairs, so that exactly two scenes were visible at any time. In half of the conditions, the scenes within the pairs were chosen from the same category, promoting within-category comparisons (*blocked schedule*). In the other half, the scenes in the pairs were from different categories, promoting across-category comparisons (*interleaved schedule*), see Figure 4. We additionally manipulated the similarity of scenes within and between pairs, as shown in Figure 1b. We exactly replicated these conditions for the model.



(a) similar within pairs (b) dissimilar within pairs

Figure 3: Two scene arrangements of PBP 24 for the blocked schedule. The arrangements vary in the similarity of the scenes within the shown scene pairs, here marked by gray rectangles.

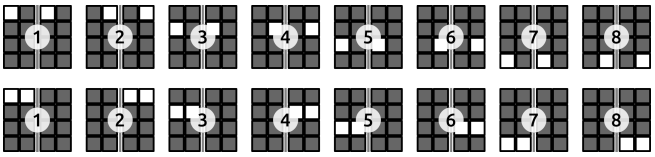


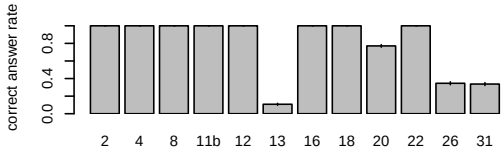
Figure 4: Presentation schedules. Above are the position and sequence in which the scenes are shown during blocked (top) and interleaved (bottom) presentation. Both the algorithm and the participants could proceed through the eight states as often as they wanted using their own pace. White squares represent visible, gray squares represent hidden scenes.

The participants worked on 22 PBPs, each of which requires knowledge of a set of basic features like relative spatial positions or physical properties like stability to solve them. Based on the basic feature-detectors we equipped our model with, it can in principle solve 12 of the original 22 PBPs. We ran the model 100 times for each condition for each of the 12 problems. For each run, we recorded whether a solution was found in less than 2500 actions and if so, how many actions were performed.

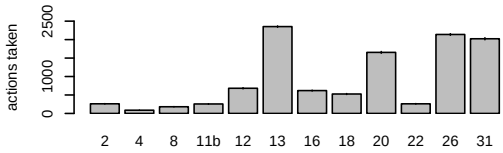
Results

Figure 5 shows the model's success rate and how many actions it used in average. The model's performance is at ceiling for most of the problems and we will therefore focus on the number of actions in our subsequent analysis.

In an analysis of the data generated by the runs of the model, there are two valid ways of mapping the eight conditions onto three 2-level factors. First, we can look at the presentation schedule (blocked vs interleaved), the between-category similarity (high vs low), and within-category similarity (high vs low) of scenes in the same or adjacent pairs,



(a) model's rate of solving



(b) average number of actions taken by model

Figure 5: (a) The model's rate of finding a solution to each of the 12 problems with a fixed cut-off at 2500. (b) The average number actions taken by the model until a solution was found or the search was stopped. Error bars represent standard errors. The x-axes show the problem numbers.

which is the type of analysis we used in our previous study. The respective $2 \times 2 \times 2$ repeated measures ANOVA with the number of actions taken by the model as the dependent variable gives the following results. There is a significant effect of presentation schedule, $F(1,99) = 255$, $p < .001$, of within-category similarity, $F(1,99) = 6.92$, $p = .01$ and of between-category similarity, $F(1,99) = 11.1$, $p = .001$. Additionally, both of the similarity conditions interacted with the presentation schedule with $F(1,99) = 13.1$, $p < .001$ and $F(1,99) = 4.6$, $p = .034$, for within-category similarity and between-category similarity, respectively. There were no other significant effects ($p > .05$).

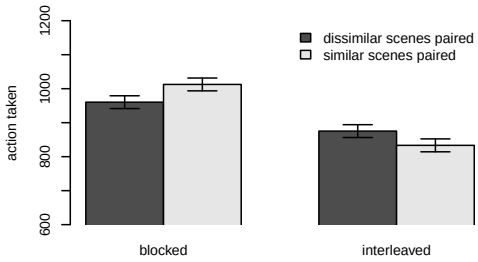
The second way of looking at the data uses the presentation schedule, as well as the similarity of scenes within each scene pair and between adjacent scene pairs as factors. A respective $2 \times 2 \times 2$ repeated measures ANOVA with the number of actions as the dependent variable reveals a significant main effect of the presentation schedule, $F(1,99) = 255$, $p < .001$ and a significant interaction of the presentation schedule and the similarity of scenes within the presented scene pairs, $F(1,99) = 39.3$, $p < .001$. There were no other significant effects ($p > .05$).

Discussion

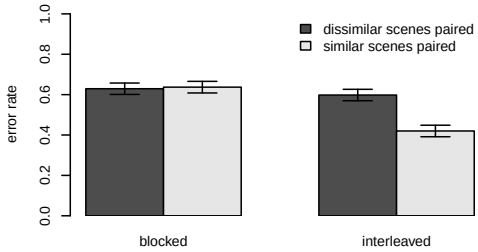
Naturally, the results of both analyses are compatible. The computational model finds solutions significantly faster for interleaved compared to blocked presentation, an effect we also found in our previous study with human participants.

In the interleaved condition, which promotes comparisons across categories by composing scene pairs of scenes from different categories, the model performed better when those scenes were similar to each other. This is in line with our expectations both from the reviewed literature and our previous experiments with human participants. Figure 6b shows the result of the first study from Weitnauer et al. (2014), that used the same setup and conditions we ran the model on. We reanalyzed the original data using only the 12 problems that were solved by the model.

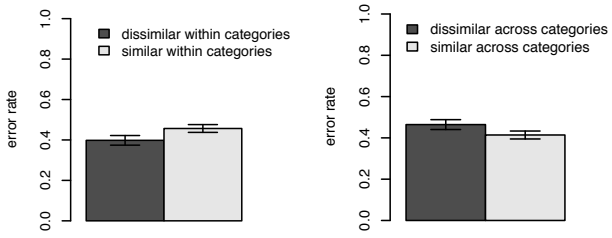
In the blocked condition, scene pairs were composed of scenes from the same category promoting within category comparisons and in this case, the model performed better when the scenes were dissimilar to each other. This result was expected from the reviewed literature, too. Figure 6c depicts the results of the second study from Weitnauer et al. (2014), were participants were presented all scenes simultaneously. The within- and between-category similarity was manipulated by placing similar or dissimilar scenes next to each other. The study showed a significant benefit of placing dissimilar scenes next to each other within each category. It is plausible to relate this result to the blocked condition in the sequential presentation, as we know from a preliminary eye-tracking study that participants do about three times more within-category saccades than between-category saccades when being shown all scenes at once.



(a) computational model



(b) humans, sequential presentation



(c) humans, simultaneous presentation

Figure 6: The computational model (top) finds solutions significantly faster for interleaved presentation, when comparing similar versus dissimilar scenes across categories, and when comparing dissimilar versus similar scenes within categories. The same advantage of interleaved presentation and comparing similar scenes across categories was significant in a study with human participants using the same setup (middle). In a study that showed all scenes simultaneously to human subjects (bottom), there was a significant advantage of comparing dissimilar scenes within categories. Error bars represent standard errors.

General Discussion

We presented a novel computational model for learning structured concepts on physical scenes. It solves a challenging problem since the Physical Bongard Problems we used as concept learning task have an open-ended feature space and concepts based on the inner structure of the physical scenes. The model perceives the scenes while also building and checking hypotheses, with both processes meaningfully interacting with each other and constraining each other.

We applied the model on the same problems and conditions we used for human participants in an earlier study and qualitatively replicated the results. The model's learning performance is effected in the same way when manipulating the kinds of comparisons between scenes that can be made directly. Specifically, comparing instances between categories as well as comparing similar versus dissimilar instance between categories and dissimilar versus similar instances within categories improves learning performance. This is in line the research in the category learning literature we reviewed in the introduction.

While Physical Bongard Problems are categorization tasks and not the kind of situations that analogy-making literature is typically concerned with, both are connected in that structured situations have to get related to each other. Although our computational model never considers the mappings between individual elements of one scene to the elements of another scene explicitly, they can easily be derived once a common interpretation of the scenes is found. Applying, e.g., the interpretation "the circle ends up left of the other object" to several scenes will identify the corresponding circle and 'other' elements in all situation. Cases in which no meaningful 1:1 mappings between objects can be extracted are cases where they do not exist in a straightforward way, like in "all objects are close to each other".

An essential part of the presented cognitive model are the heuristics for deciding which objects, features and hypotheses to look at next. Our design of the heuristics was guided in part by what is known about limitations in human cognition, and in part by insights from introspection during solving PBPs ourselves. The resulting model can solve many of the original problems and the way different presentation schemes influence its performance qualitatively resembles results from human participants. Still, there remains much to explore in how the current or similar heuristics lead to the learning behavior of the model and what makes those heuristics plausible or not. One way of gaining further insight is to derive what a rational decision would be, given the so-far perceived data and a set of overt assumptions about the structure of the solution space. We are currently working on such a Bayesian derivation and several of the algorithm's heuristics are following naturally from reasonable assumptions.

References

Anderson, J. R. (1991). The adaptive nature of human categorization. *Psychological Review*, *98*(3), 409.

Birnbaum, M., Kornell, N., Bjork, E., & Bjork, R. (2012). Why interleaving enhances inductive learning: The roles of discrimination and retrieval. *Memory & Cognition*, 1–11.

Carvalho, P. F., & Goldstone, R. L. (2013). Putting category learning in order: Category structure and temporal arrangement affect the benefit of interleaved over blocked study. *Memory & cognition*.

Falkenhainer, B., Forbus, K. D., & Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial intelligence*, *41*(1), 1–63.

Foundalis, H. E. (2006). *Phaeaco: A cognitive architecture inspired by bongard's problems*. Unpublished doctoral dissertation, Indiana University.

Getoor, L., & Taskar, B. (2007). *Introduction to statistical relational learning*. MIT press.

Goodman, N. D., Tenenbaum, J. B., Feldman, J., & Griffiths, T. L. (2008). A rational analysis of rule-based concept learning. *Cognitive Science*, *32*(1), 108–154.

Hofstadter, D. (1979). *Gödel, escher, bach: an eternal golden braid*. Harvester Press.

Hofstadter, D. (1996). *Fluid concepts and creative analogies: Computer models of the fundamental mechanisms of thought*. Basic Books.

Hummel, J. E., & Holyoak, K. J. (1996). Lisa: A computational model of analogical inference and schema induction. In *Proceedings of the eighteenth annual conference of the cognitive science society* (pp. 352–357).

Kang, S., & Pashler, H. (2012). Learning painting styles: Spacing is advantageous when it promotes discriminative contrast. *Applied Cognitive Psychology*, *26*(1), 97–103.

Kruschke, J. K. (1992). Alcove: an exemplar-based connectionist model of category learning. *Psychological review*, *99*(1), 22.

Love, B. C., Medin, D. L., & Gureckis, T. M. (2004). Sustain: a network model of category learning. *Psychological review*, *111*(2), 309.

Medin, D., & Ross, B. (1989). The specific character of abstract thought: Categorization, problem solving, and induction. *Advances in the psychology of human intelligence*, *5*, 189–223.

Nosofsky, R. M. (1986). Attention, similarity, and the identification–categorization relationship. *Journal of experimental psychology: General*, *115*(1), 39.

Rost, G. C., & McMurray, B. (2009). Speaker variability augments phonological processing in early word learning. *Developmental Science*, *12*(2), 339–349.

Weitnauer, E., Carvalho, P. F., Goldstone, R. L., & Ritter, H. (2014). Similarity-based ordering of instances for efficient concept learning. In *36th annual conference of the cognitive science society* (p. 1760-1765). Quebec City, Canada.

Weitnauer, E., & Ritter, H. (2012). Physical bongard problems. *Artificial Intelligence Applications and Innovations*, 157–163.