California Engineer

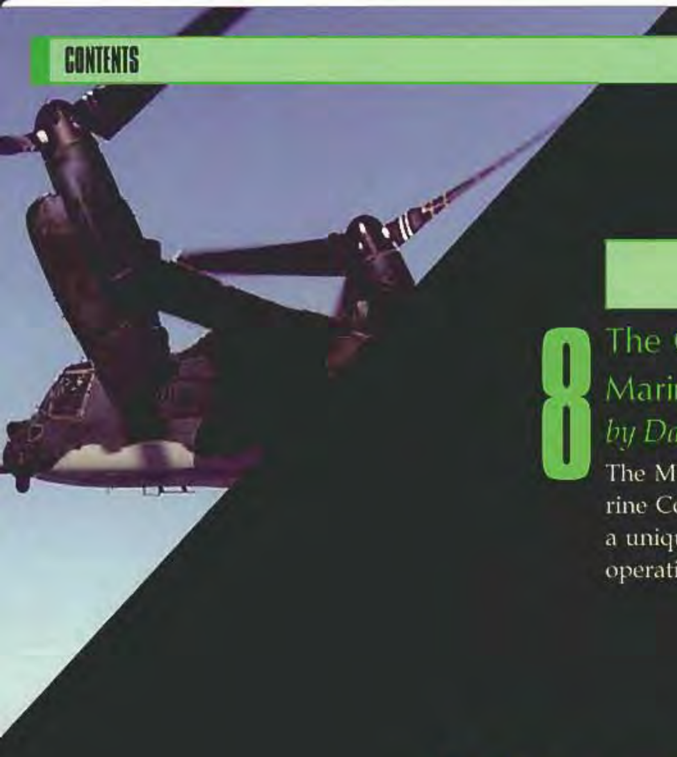
Student Journal of the UC Engineering Colleges

also in this issue

Volume 81 Issue 2 Winter 2002–2003

Semantic Priming   Flutes   Ultra-Wideband Technology

WINTER 2002-03 / VOLUME 81 / NUMBER 2



FEATURE ARTICLE

Also in this issue:

California Engineer
Student Journal of the UC Engineering

The California Engineer (ISSN 0008-1027) is published four times a year in March, May, September and December. California Engineer Publishing Company, Inc. Circulation 7,000. Subscriptions $8.00 per annum. Member of *Engineering* College Magazines Associated. The information provided herein is the responsibility of the individual authors. Correspondence to the Editor of California Engineer can be made via e-mail: editor@calengr.berkeley.edu, by phone (510) 642-8679 or through mail. Periodicals postage paid at San Dimas, California and at additional mailing offices. For national advertising rates call Pentagon Publishing Inc., Patrick Stansbury (CEO) 1-800-426-4231 (ext. 1251). For local advertising rates call California Engineer. **POSTMASTER: Please send address changes to: *California Engineer*, University of California, 221 Bechtel Engineering Center, Berkeley, CA 94720.**

"**se·man·tics** n.
Meaning or the study of meaning derived from morphemes, words, and sentences.

"**prim·ing** n.
The process by which an earlier encounter with a stimulus increases the likelihood of that stimulus or a related stimulus being remembered at a later time."

—Huffman, Karen.
Psychology in Action, 6th Ed.

learning categories using Semantic Priming in a Bayesian Framework

by Ray Luo

Human beings are capable of learning nouns with just a few examples and almost no explicit feedback. But a simple explicit memory model cannot account for either the rapidity or the accuracy of learning. Thus, reliance on an implicit memory model that accounts for the effects of past experiences is necessary. One approach is to represent nouns as basic categories with uncertain feature values. As more data are incorporated, the categories become more and more like what the nouns represent in the real world. Since categories are built up from experience, it is expected that physical interaction will improve category acquisition. Current research suggests that implicit category comparisons (via semantic priming) model this physical interaction.

Much of the modeling ideas presented here are inspired by Schooler et. al. [5]. As in Schooler et.al., rational analysis [1] is applied to the modeling of results from psychological experiments. The REM model of Schooler et. al. was an attempt to account for episodic retrieval via a model that assumes that priming acts "alter the word's lexical-semantic memory representation" [5]. Justification for direct alteration of the representation

Copy Edit by: Peng Zhou. Graphics by: Derek Chan. Layout by: Carven Chan

is given in Schooler et. al. Most objections to Bayesian models point to the lack of plausibility for the human mind to perform probabilistic computations during experimental tasks. As such, this experiment will make simplifying assumptions on the REM model by using simpler Bayesian methods that are assumed to be within reach of the neural computation.

The experiment focused on animal categories so as to simplify direct comparison with psychological experiments. Ideas suggested here are relevant for text categorization as well. In particular, any domain that can be decomposed into hierarchical categories can be modeled. The algorithm uses a simple representation of features and categories to facilitate comparison. In a real-world application, Bayesian network representations should be constructed for both basic and parent categories. A suggestion on how this can be done is presented later, although the implementation is only at an experimental phase.

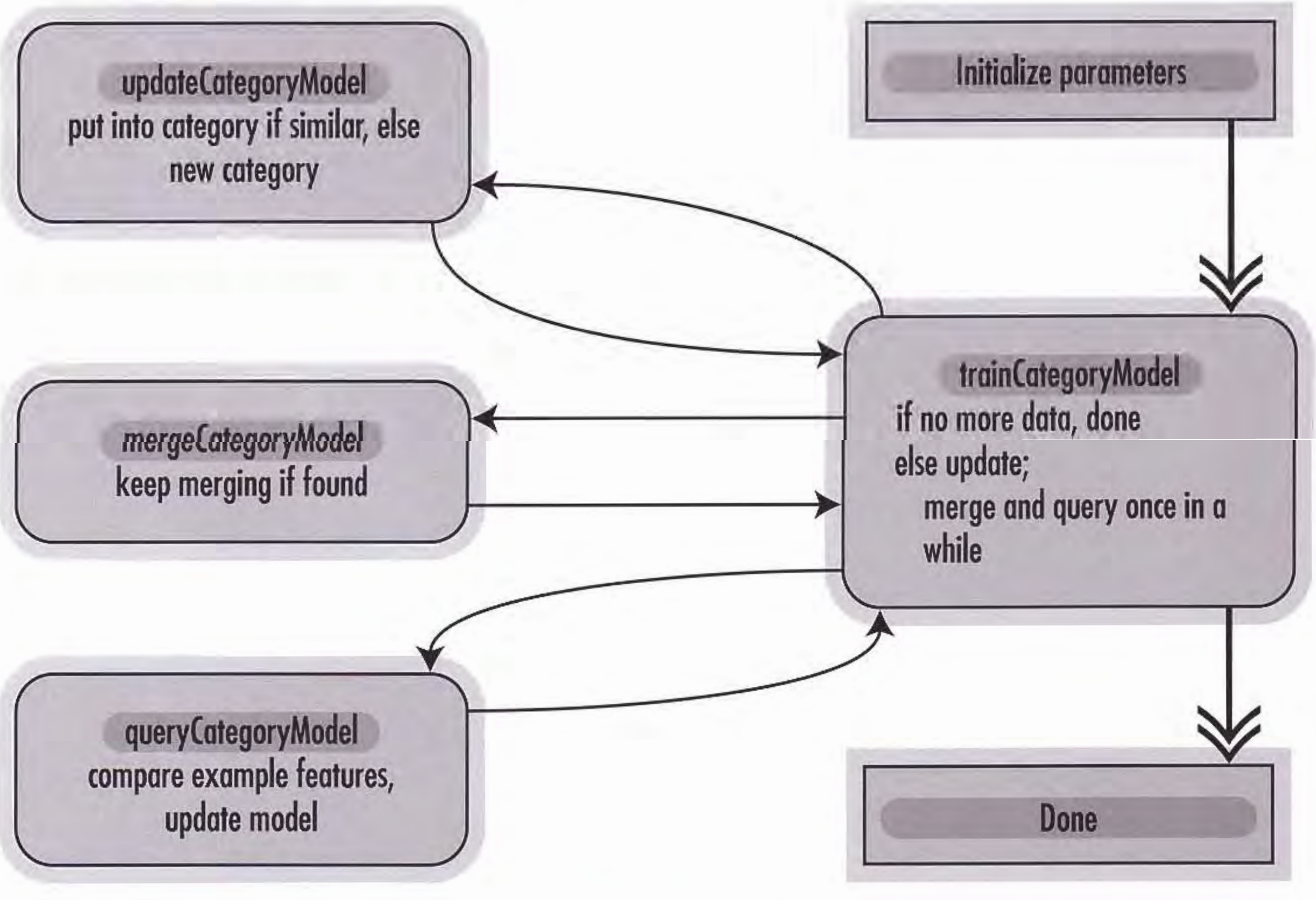
Assumptions

First, it is assumed that no perceptual errors are made during learning and testing. There is no probability distribution associated with input animal features for each example. The lack of errors in perception does not, however, imply that the input data is noiseless. It is assumed that the distributions of attributes of an animal are independent and jointly Gaussian. Data are generated using normal distributions with pre-determined means and variances that reflect the attribute domain for each animal. Noisy examples with random features are also fed into the model periodically. The assumption suggests that the model captures the only probabilistic aspect of category learning. Thus, the model itself (not the perceptual system) is responsible for handling noisy data.

Second, only two levels of categories are relevant to the task. The basic categories consist of animal types. In particular, data for ant, snail, frog, cat, wolf, cow, and whale were given as input. The model forms abstract models of the data, so no actual label is given to any category. The idea is that a category is described by average feature values and a measure of the spread of feature values. It is the model's task to minimize the complexity of sets of categories while maximizing the probability of correctly identifying the category of any example given the feature values. This can be done using inference on a belief network. The parent categories consist of graded conceptual categories involving the animal types. For example, the large animal category should include whale and cow as salient examples, with wolf possibly having graded membership. Graded membership is not represented explicitly, however. The model will keep track of one best prototype, which can be updated with a given probability during comparison tasks. Conceptual categories are constructed using the different features associated with animals. In particular, large, small, cute, hideous, ferocious, and tame animal were used. Ideally, other objects could be modeled as well, allowing the conceptual categories to be simply large, small, cute, hideous, etc. This makes the model more general, less domain-dependent, but harder to assess.

Third, inferences and comparisons done on particular members of basic categories are necessary for learning and structuring both basic and parent categories. Here the focus is on feature value comparisons. A question that can be asked is, for example, "which animal is larger, cat or frog?" Luo [4] reviews these tasks. In modeling early development, the semantic priming that results from querying the subject is associated with changes in the underlying categories. The claim is that these changes result from comparison of basic category feature values with prototype feature values. In the example above, size feature values for cat and frog are compared with the prototype large animal feature value. Changes are made to the abstract categorical representations of cat and frog size values and spreads. Prototype values are updated as needed. One natural question that arises is: if feature value estimates are already kept by our basic category representation, could not one just compare the estimates and return the appropriate category (in this case, cat for the larger animal)? There are three problems with this approach. First, the approach predicts that all comparisons should take the same amount of time on average. This has been shown to be false [2]. For example, it is much faster to decide between whale and ant than cat and frog. Second, the approach predicts that no errors could occur. For example, frogs would always be judged smaller than cats. This does not hold experimentally [4]. What is necessary is a model that is generally true, but can give wrong answers in some contexts. Third, there are uncertainties associated with feature estimates and this is why a measure of spread is needed. As more examples of cats are shown to the system, the model becomes better at estimating the relative size of cats, but it is never absolutely sure that cats are smaller than wolves. The approach is to compare cat and wolf sizes with the prototypical size. Relative distances between category and prototype size feature values can then model differences in judgment reaction times. This approach can be thought of as a variance reduction technique. Knowledge about prototypical cases, which also have associated uncertainty values, is leveraged.

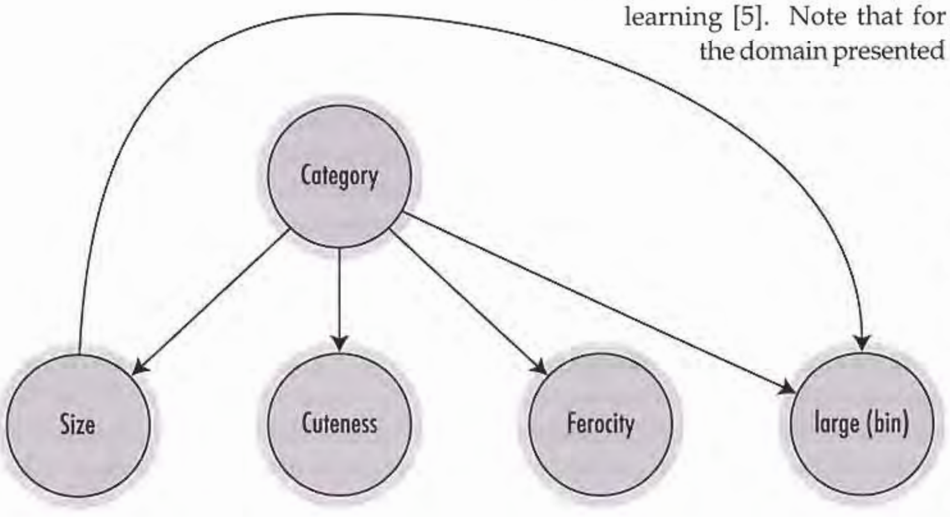
Fourth, there exist abstract category representations in the mind independent of but influenced by linguistic labels. This model keeps abstract categories with no particular linguistic interpretation. This model of the mind relies on generalization over similar word senses. Hence, frog and toad may refer to relatively similar animals that have similar values with respect to the set of features. A child may not be able to distinguish (at least at first) between these similar animal types. Thus a structure is needed that represents the features of a frog-toad without labeling it. It is possible then to compare these abstract categories amongst themselves and perform inferences purely on the basis of feature structures. Words with different senses (cat, for example) are described by different categories while different words denoting animals with similar features (e.g. frogs and toads) are described by the same category. In the future the linguistic label associated with an abstract category can, hopefully, be shown as a probability distribution over words or phrases acquired from experience. Instead of saying, for example, that the word cat has two senses (kitty and tiger), it'll be possible to say that abstract category A has a probability of 0.95 of being labeled "cat" and abstract category B has a probability of 0.5 of being labeled "cat." With experience, the abstract categories become finer because the variances of the feature values become smaller. In the limit of infinite experience, the two word senses of "cat" will be described by separate abstract categories with labeling probability 1.0, because a separate abstract category will describe "tiger" with probability 1.0. It is also hoped that someone will prove the following claim: given a pre-defined,



finite, sufficiently "nice" set of training examples, the category partitions formed by the algorithm will converge in probability to a set of non-overlapping category distributions (in the sense of zero variance feature values) as the number of sweeps goes to infinity. This statement would then imply that each category would eventually be labeled by its own word. For now, it is assumed that most likely labels are found magically by some other mechanism. It is not necessary to worry about possible ambiguity here because the comparison simulations are done with abstract categories and the prototypes generalize over the entire domain independently of particular word labels. Note, however, that a prototype is a single instance from the basic categories. When humans are confronted with a question regarding size they don't automatically think of a whale. The assumption here is that humans perform subconscious comparisons with a vague superordinate prototype whose variance is averaged over a few salient examples.

Lastly, it is assumed that this direct representation of human semantic processing can be efficiently implemented neurally. One approach is to translate the mean-variance representation into a simple Bayesian network. This will be described after the model and the algorithm.

Left: High-level overview of the algorithm. Right: A Bayesian network implementation.



Model

The basic claim of the model is that the meaning of a noun in some restricted domain is its abstract representation in relation to all other nouns in the domain[1]. Moreover, this relationship is captured by comparisons with sets of superordinate prototypes that capture general characteristics of objects referred to by basic terms.

The model acquires basic category representations by merging abstract models of category partitions and by performing inferences and comparisons using these abstract partitions. It is assumed that implicit reinforcements for correct and incorrect responses have been provided internally. This model controlled experimental paradigms in which the subject is given a pair of objects whose feature values fall on some subjective scale. Errors are frequently caused by priming and going too fast. Moreover, learning probability distributions for category labels is assumed to be incremental, so it is not necessary to tell the subject not to say something, because it will not be said as predicted.

Thus, the model has abstract categories that model the internal representations of nouns and learning mechanisms that takes knowledge from the real world, and reconstructs, as best as they can, the entities that the nouns represent, using comparison-based implicit learning [5]. Note that for the domain presented here, this consists only of two hierarchical levels. Extensions beyond animal types can be constructed by treating each superordinate category as also a basic category.

Algorithm

Both model merging and prototype formation depend on order of data presentation. Since any possible merge with lower cost will be performed, there is a chance that the optimal merge will not be performed. Similarly, since early experience is weighted more heavily in prototype formation, it is possible to get unrepresentative prototypes fairly late into the learning process.

Prototype acquisition is modeled as "Markov." That is, the mean feature value for a prototype is drawn from either the current value or the maximum (or minimum) value over subordinate categories. As the model accumulates experience, it tends to stay with the current value; initially, it tends to choose the optimal value. The idea is that a single basic category instance serves as the prototype, but that the variance associated with the prototype is estimated from the current variance and the difference between the optimal value and the current value. Hence, the model forgets the past because the current prototype value variance captures all that is needed to know about past prototype values. The reason for this design decision is the seemingly transient nature of a prototype. When people think of a cat, they are basing their estimates on prior knowledge about distributions of feature values of a cat. When people think of a large animal, however, they first think of the idea of large before thinking of a prototype. People do not know everything about this prototype animal but they do know that it is large. After using this information, the prototype is put away. The claim here is that not all the feature value distributions of any prototype of any superordinate category are known. Given this assumption, one sensible model would be to base the prototype on either the last prototype or the best possible prototype. The probability of choosing between the two depends on experience. As more and more examples are observed, it becomes less likely to switch to the best possible prototype, because the current prototype must already be pretty good.

Here is a walkthrough of the algorithm in detail. First, random examples are created consisting of size, cuteness, and ferocity values for our animal types (ant, snail, frog, cat, wolf, cow, and whale), with noisy examples given random features values. Next, the data is given to the model incrementally. The model first scans through the categories to see if the new example can be put into an existing category. If not, a new category is constructed. After a few data incorporation steps, the merging algorithm is invoked. The cost of every pair of possible merges is computed and the best merge to perform is chosen. The first merge is performed with a higher probability (lower cost) than the probability (cost) of the original model. The basic idea is to try and maximize

P(Model | Data) which is equivalent to *P(Data | Model)* · *P(Model)* / *P(Data)*.

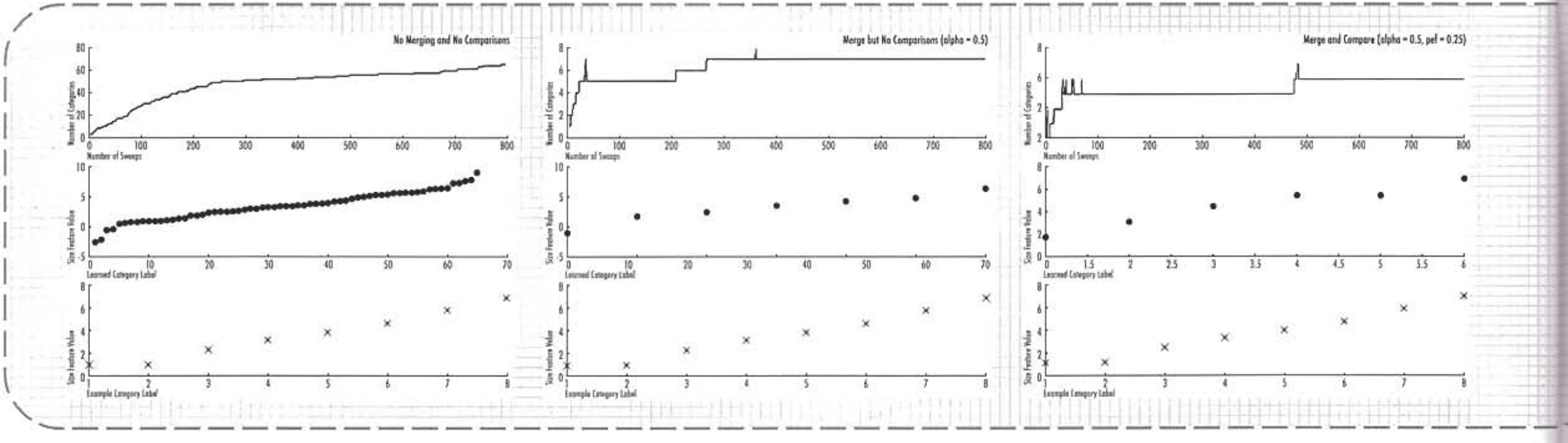
Here *P(Data)* can be ignored since it is the same for any merge. This equation is then translated to:

cost(Model) being equal to

$\alpha \cdot size\ (Model) + (1 - \alpha) \cdot var(Model)$, where *size(Model)* is the total number of categories (i.e. the size of the partition), and *var(Model)* is the total variance of all category feature values. The idea is that for a set of data, a smaller model is a better, more succinct description of an underlying representation. Therefore, it is desirable to minimize the size of the partition by merging categories together. But merging requires the re-computation of the variance of feature values for the resulting category, which can increase because the two categories to be merged are distinct. This should be kept low if good predictions are to be made about which category a novel set of feature values belongs to. Thus there is a trade-off between compactness and representational power. The algorithm will keep finding and merging categories until a merge with a lower cost cannot be found.

Between data incorporation steps, the query mechanism is periodically invoked, which models implicit comparisons that facilitate category learning. The prototypes are first updated probabilistically by taking on the parameters modeling the experience of a child. A past experience factor (pef) between 0 and 1 is defined that is initially small, but becomes larger as the number of sweeps increases. With probability 1 minus pef, the superordinate prototypes are replaced (large animal, small animal, cute animal, hideous animal, ferocious animal, tame animal) with the example with the optimal feature value in the category partition. Since pef increases with time, the model is less likely to change prototypes as it obtains more and more experience. Next, the best matching categories that correspond to the querying examples are found. These categories are compared with the prototype categories for each feature. The categories are explicitly changed by a small amount that is related to pef. The larger category is moved toward the value of the larger prototype, the smaller category toward the value of the smaller prototype, and similarly for cuteness and ferocity. The idea is that given a pair of examples, the model is asked to choose, for example, which animal is larger. The target example is coded as being large and the rejected example as being small. This primes future judgments on both of the examples [4]. Specifically, the internal representation of the target and rejected examples are changed by some small factor determined by experience and by the distance between feature values of examples and prototypes. Note that the prototype representation can also be changed if the target example feature value is "more optimal" than the prototype feature value. This can happen because the prototypes are only updated probabilistically so the prototypes may not be the best.

It is possible to translate the model to a Bayesian network. As suggested by Koller & Sahami [3], the categories can be represented as discrete nodes that serve as parents of Gaussian feature nodes. Thus the statistics for each feature are specified and query the category node given the feature evidence. The problem is to implement an unsuper-

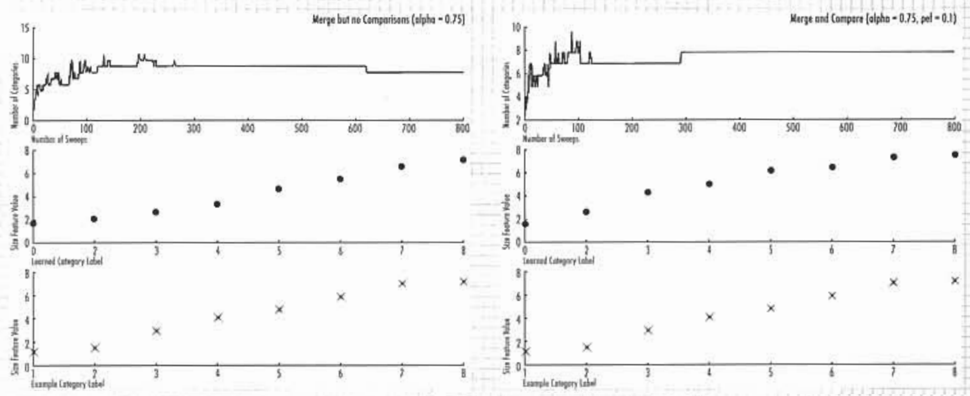


vised learning algorithm that updates the values that the category node can take on as more evidence arrives. Here, independence of each feature value is assumed. The superordinate category node "large" is a binary representation of a "decision" based on current feature estimates. Note that prototype values are kept implicitly within the distribution for the superordinate node. It may be better to use decision nodes for the superordinate category nodes, but whether inference would work is still in question.

Results

In general, category partitions learned using feature comparisons tend to generalize a bit more, so that a smaller number of categories results. Note that one obvious problem with the model merging algorithm is that it tends to give inaccurate values for animals whose size is small.

It is possible to change the parameters of both the model merging and the feature comparison algorithms. Over many iterations of the experiment, it is discovered that feature comparison tends to do wonders when pef is set to be small (R 0.1). In that case, feature comparison tends to generalize more than model merging alone.



Discussion

This Bayesian model-merging algorithm applied to category-acquisition is not without its faults. For example, it is assumed that in the presence of small sample sizes, humans still believe that the distribution of feature values in the real world is approximately normal for each distinct animal. More reasonable approaches involving the Dirichlet density can be found in Anderson [1]. It is also assumed that each feature comparison task occurs regularly and that all features of a category are compared. In the real world, this is rarely the case. A person might be led into comparing, for example, sizes more than cuteness, and cats and dogs more often than polar bears and lady bugs. A proper model for comparison tasks in early language acquisition would involve modeling the world in which the child lives, which is beyond present scope and capabilities. Note also that the independence of each feature given the category is assumed, which cannot be a true model of the world. Finally, the model assumes that model merging and feature comparison happen independently. Note, however, that depending on the current comparison task, it might be more or less suitable to merge disjoint categories. For example, consider the comparison of cardinals and blue jays. The model does not have much of an idea of what they are and might consider them to be close to each other in size. Are they also in one category?

One other proposal would be to treat the problem as minimization of Boolean complexity given features and examples. The best model is that which minimizes the minimal formula associated with the disjunctive normal form of the evidence written in conjunction with features and disjunction among examples. This has not been worked out yet.

The Bayesian network representation still needs to be explored thoroughly. A way in which learning of category values will work for a Bayes net still needs to be worked out. Superordinate category representation can also be improved. One idea is to use separate nodes for each category value (ie. animal name). Then all the model has to do is prune the network as more evidence arrives. The problem is that each node would be the parent of the feature values. Hence each augmentation of a feature adds O(k) links, where k is the number of values a category can take on. The model should also allow feature values to link with each other, thereby removing the feature value independence assumption.

This model has presented a Bayesian approach to learning category representation. It can be seen that learning is facilitated by implicit interactions with the real world. Here, the focus is on feature comparison and priming. The principle, however, is more general. In modeling domain knowledge, the various contexts in which a name can occur and the various interpretations the name can take on with respect to superordinate category prototypes need to be considered.

References

1. Anderson, J. R. 1991. The adaptive nature of human categorization. Psychological Review, 98(3):409-429.
2. Banks, W. P.; Fujii, M.; and Kayra-Stuart, F. 1976. Semantic congruity effects in comparative judgments of magnitud es of digits. Journal of Experimental Psychology: Human Perception and Performance, 2:435-447.
3. Koller, D. and Sahami, M. 1997. Hierarchically classifying documents using very few words. In Proc. ICML-97, 170-178.
4. Luo, R. 2000. Response interference in categorical judgment tasks. Online http:// inst.eecs.berkeley.edu/~rluo/categ_learn_doc/perc&cog_np&sc-revised-text.doc.
5. Schooler, L. J.; Shiffrin, R. M.; and Raaijmakers, J. G. W. 2001. A Bayesian model for implicit effects in perceptual identification. Psychological Review, 108(1):257-272.

Matlab code for the algorithm can be found at http//inst.eecs.berkeley.edu/~rluo/categ_learn_doc/categ_learn.zip.

Run the file learn_animal_categories in Matlab to begin experimenting. You should have the BNT toolkit installed http://www.cs.berkeley.edu/~murphyk/Bayes/request.html

Learning Categories Using Semantic Priming in a Bayesian Framework

Ray Luo

Electrical Engineering and Computer Sciences
University of California
Berkeley, California 94720
rluo@cory.eecs.berkeley.edu

Abstract

A probabilistic algorithm for unsupervised learning of categories using internal comparisons is presented. The Bayesian merging algorithm uses parent category prototypes to facilitate basic category comparisons. Our model takes on the new task of learning internal representation with no explicit feedback. We assume that humans make implicit comparisons between novel categories and merge categories they consider similar, in a probabilistic sense. We find that the algorithm out performs simple data incorporation, and that the algorithm supplemented by feature comparisons did even better. The consideration of context brought about by feature comparison allows the model to generalize over noisy examples. However, a more compact representation is desired.

Introduction

Humans can learn nouns easily given a few examples and almost no explicit feedback. A simple explicit memory model can account for neither the rapidity nor the accuracy of learning. Thus we must rely on a model of implicit memory that accounts for the effects of past experience. One approach is to represent nouns as basic categories with uncertain feature values. As more data are incorporated, the categories look more and more like what the noun represents in the real world. Since categories are built up from experience, we expect physical interaction to improve category acquisition. We suggest that implicit category comparisons (via semantic priming) model this physical interaction.

Much of the modeling ideas presented are inspired by Schooler et. al. (2001). As in Schooler et. al., we apply rational analysis (Anderson, 1990) to the modeling of results from psychological experiments. The REM model of Schooler et. al. was an attempt to account for episodic retrieval via a model that assumes that priming acts to "alter the word's lexical-semantic memory representation" (2001). Justification for direct alteration of the representation are given in Schooler et. al. Our priming effects will be based on comparison tasks. Most objections to Bayesian models point to the lack of plausibility for the human mind to perform probabilistic computations during experimental tasks. Our model will make simplifying assumptions on the REM model by using simpler Bayesian methods that we assume are at least can be computed neurally. Moreover, we will apply Bayesian reasoning to learning a best current model. We will simplify the algorithm by making some assumptions on human perceptual capabilities and mental representations.

We focused on animal categories so as to simplify direct comparison with psychological experiments. Ideas suggested here are relevant for text categorization as well. In particular, any domain that can be decomposed into hierarchical categories can be modeled. Our algorithm uses a simple representation of features and categories to facilitate comparison. In a real-world application, Bayesian network representations should be constructed for both basic and parent categories. We'll suggest how this can be done later, although the implementation is only at an experimental phase.

Assumptions

First, we assume that no perceptual errors are made during learning and testing. That is, there is no probability distribution associated with input animal features for each example. The lack of errors in perception does not, however, imply that the input data is noiseless. We assume that the distributions of attributes of an animal are independent and jointly Gaussian. Data are generated using normal distributions with pre-determined means and variances that reflect the attribute domain for each animal. Noisy examples with random features are also fed into the model periodically. The assumption suggests that the only probabilistic aspect of category-learning is captured by our model. Thus, our model (and not the perceptual system) is responsible for handling noisy data.

Second, only two levels of categories are relevant to our task. Our basic categories consist of animal types. In particular, data for ant, snail, frog, cat, wolf, cow, and whale were given as input. Note, however, that the model forms abstract models of the data, so no actual label is

Submitted to NIPS Neural Information Process Systems given to any category. The idea is that a category is described by average feature values and a measure of the spread of feature values. It is the model's task to minimize the complexity of sets of categories while maximizing the probability of correctly identifying the category of any example given the feature values. This can be done using inference on a belief network. Our superordinate parent categories consist of graded conceptual categories involving the animal types. For example, the large animal category should include whale and cow as salient examples, with wolf, and possibly cat having graded membership. Graded membership is not represented explicitly, however. The model will keep track of one best prototype, which can be updated with a given probability during comparison tasks. We construct conceptual categories using the different features associated with animals. In particular, large animal, small animal, cute animal, hideous animal, ferocious animal, and tame animal were used. Ideally, we should model other objects as well and allow our conceptual categories to be simply large, small, cute, hideous, etc. This makes the model more general, less domain-dependent, but harder to assess. Thus we stick to animals at present.

Third, inferences and comparisons done on particular members of basic categories are necessary for learning and structuring both basic and superordinate parent categories. Here we focus on feature value comparisons. We can ask, for example, "which animal is larger, cat or frog?" Luo (2000) reviews these tasks. In modeling early development, we can associate the semantic priming that results from querying the subject with changes in the underlying categories. Our claim is that these changes result from comparison of basic category feature values with prototype feature values. In the example above, size feature values for cat and frog are compared with the prototype large animal feature value. Changes are made to the abstract categorical representations of cat and frog size values and spreads. Prototype values are updated as needed. One natural question that arises is: if feature value estimates are already kept by our basic category representation, couldn't we just compare the estimates and return the appropriate category (in this case, cat for the larger animal)? There are three problems with this approach. First, the approach predicts that all comparisons should take the same amount of time on average. This has been shown to be false (Banks et. al., 1976). For example, it is much faster to decide between whale and ant than cat and frog. Second, the approach predicts that no errors could occur. For example, frogs would always be judged smaller than cats. This does not hold experimentally (Luo, 2000). We need, instead, a model that is generally true, but can give wrong answers in some contexts. Third, there are uncertainties associated with feature estimates (This is why we need a measure of spread). For example, some cats (tigers) are very large while other cats (kitty) are very small. As more examples of cats are shown to the system, the model becomes better and better at estimating the relative size of cats, but it is never absolutely sure that cats are smaller than wolves. Our approach is to compare cat and wolf sizes with the prototypical size. We can model differences in judgment reaction times by relative distances between category and prototype size feature values. You can think of our approach as a variance reduction technique. We are leveraging our knowledge about prototypical cases, which also have associated uncertainty values.

Fourth, there exist abstract category representations in the mind independent of, but influenced by, linguistic labels. Our model keeps abstract categories with no particular linguistic interpretation. The reason for this is that synonyms abound in the real world. While there is evidence that no "exact" synonyms exist in the technical sense, our model of the mind relies on generalization over similar word senses. Hence, frog and toad may refer to relatively similar animals that have similar values with respect to our set of features. A child may not be able to distinguish (at least at first) between these similar animal types. Thus we need a structure that represents the features of a frog-toad without labeling it. We can then compare these abstract categories amongst themselves and perform inferences purely on the basis of feature structures. Words with different senses (cat, for example) are described by different categories while different words denoting animals with similar features (frog and toad, for example) are described by the same category. We hope to show in the future that the linguistic label associated with an abstract category is a probability distribution over words or phrases acquired from experience. Instead of saying, for example, that the word cat has two senses (kitty and tiger), we say that abstract category A has a probability of 0.95 of being labeled "cat" and abstract category B has a probability of 0.5 of being labeled "cat." With experience, the abstract categories become finer because the variances of the feature values become smaller. In the limit of infinite experience, the two word senses of "cat" will be described by separate abstract categories with labeling probability 1.0, because a separate abstract category will describe "tiger" with probability 1.0. We hope that someone will prove the following claim (it shouldn't be hard): Given a pre-defined, finite, sufficiently "nice" set of training examples, the category partitions formed by the algorithm will converge in probability to a set of non-overlapping category distributions (in the sense of zero variance feature values) as the number of sweeps goes to infinity. This statement would then imply that each category would eventually be labeled by its own word. In this paper, we assume that most likely labels are found magically by some other mechanism. We don't really have to worry about possible ambiguity here because our comparison simulations are done with abstract categories and our prototypes generalize over the entire domain independently of particular word labels. Note, however, that a prototype is a single instance from our basic categories. When humans are confronted with a question regarding size, however, they don't automatically think of, say, a whale. Our assumption here is that they performs subconscious comparisons with a vague superordinate prototype whose variance is averaged over a few salient examples.

Lastly, we assume that our direct representation of human semantic processing can be efficiently implemented neurally. One approach is to translate our mean-variance representation into a simple Bayesian network. We'll describe this after giving the model and the algorithm.

Model

The basic claim of the model is that the meaning of a noun in some restricted domain is its abstract representation in relation to all other nouns in the domain. (This, of course, is nothing new; see, for example, (Anderson, 1990).) Moreover, this relationship is captured by comparisons with sets of superordinate prototypes that capture general characteristics of objects referred to by basic terms.

The model acquires basic category representations by merging abstract models of category partitions and by performing inferences and comparisons using these abstract partitions. Implicit reinforcements for correct and incorrect responses are assumed to be provided internally. (This models controlled experimental paradigms in which the subject is given a pair of objects whose feature values fall on some subjective scale. Errors are frequently caused by priming and by going too fast.) Moreover, learning probability distributions for category labels is assumed to be incremental, so that we don't need to tell the subject not to say something, because she won't say it (as the model predicts).

Thus we have abstract categories that model the internal representations of nouns and learning mechanisms that takes knowledge from the real world, and reconstructs, as best as they can, the entities that the nouns represent, using comparison-based implicit learning (Schooler et. al., 2001). Note that for our domain, this consists only of two hierarchical levels. Extensions beyond animal types can be constructed by treating each superordinate category as also a basic category.

Algorithm

Both model merging and prototype formation depend on order of data presentation. Since any possible merge with lower cost will be performed, there is a chance that the optimal merge will not be performed. Similarly, since early experience is weighted more heavily in prototype formation, we may get unrepresentative prototypes fairly late into the learning process.

We chose to model prototype acquisition as "Markov." That is, the mean feature value for a prototype is drawn from either the current value or the maximum (or minimum) value over subordinate categories. As the model accumulates experience, it tends to stay with the current value; initially, it tends to choose the optimal value. The idea is that a single basic category instance serves as the prototype, but that the variance associated with the prototype is estimated from the current variance and the difference between the optimal value and the current value. Hence, the model forgets the past in the sense that the current prototype value variance captures all we need to know about past prototype values. The reason for this design decision is the seemingly transient nature of a prototype. When we think of a cat, we are basing our estimates on prior knowledge about distributions of feature values of a cat. When we think of a large animal, however, we first think of the idea of large. Then we may think of some specific animal prototypical of large animals. We don't know everything about this prototype animal but we do know that it is large. After using this information, the prototype is put away. Our claim is that we don't know *all* the feature value distributions of any prototype of any superordinate category. Given this assumption, one sensible model would be to base our prototype on either the last prototype or the best possible prototype. The probability of choosing between the two depends on experience. As more and more examples are observed, we are less likely to switch to the best possible prototype, because the current prototype must already be pretty good.

Now, we will walk through the algorithm in detail. An overview is presented in Fig. 6. First, we create random examples consisting of size, cuteness, and ferocity values for our animal types (ant, snail, frog, cat, wolf, cow, and whale), with noisy examples given random features values. Next, we give the data to the model incrementally. The model first scans through the categories to see if the new example can be put into an existing category. If not, a new category is constructed. After a few data incorporation steps, the merging algorithm is invoked. We compute the cost of every pair of possible merges and choosing the best merge to perform. For our purposes, we perform the first merge with a higher probability (lower *cost*) than the probability (*cost*) of the original model. The basic idea is that we are trying to maximize

$$P(Model \mid Data) = P(Data \mid Model) \cdot P(Model) / P(Data).$$

P(Data) can be ignored since it is the same for any merge. This equation is translated to:

$$cost(Model) = \alpha \cdot size(Model) + (1 - \alpha) \cdot var(Model),$$

where *size(Model)* is the total number of categories (i.e. the size of the partition), and *var(Model)* is the total variance of all category feature values. The idea is that a set of data, a smaller model is a better, more succinct description of an underlying representation. Therefore, we want to minimize the size of the partition by merging categories together. But merging requires us to recomputed the variance of feature values for the resulting category, which can increase because the two categories to be merged are distinct. We should keep the variability low if we are to make good predictions about which category a novel set of feature values belongs to. Thus we have a trade-off between compactness and representational power. The algorithm will keep finding and merging categories until a merge with a lower cost cannot be found.

Between data incorporation steps, we periodically invoke the query mechanism, which models implicit comparisons that facilitates category-learning. We first update the prototypes probabilistically by parameterizing the experience of a child. We define a past experience factor (*pef*) between 0 and 1 that is initially small, but becomes larger as the number of sweeps increases. With probability 1 minus *pef*, we replace our superordinate prototypes (large animal, small animal, cute animal, hideous animal, ferocious animal, tame animal) with the example with the optimal feature value in our category partition. Since *pef* increases with time, we are less likely to change our prototypes as we obtain more and more experience. Next, we find the best matching categories that correspond to the querying examples. We compare these categories with the prototype categories for each feature. We explicitly change our categories by a small amount that is related to *pef*. The larger category is moved toward the value of the larger prototype, the smaller category is moved toward the value of the smaller prototype, and similarly for cuteness and ferocity. The idea is that given a pair of examples, we are asked to choose, for example, which animal is larger. We code the target example as being large and the rejected example as being small. This primes our future judgments on both of the examples (Luo, 2000). Specifically, our internal representation of the target and rejected examples are changed by some small factor determined by experience and by the distance between feature values of examples and prototypes. Note that our prototype representation can also be changed if the target example feature value is “more optimal” than the prototype feature value. This can happen because we only update the prototypes probabilistically (so the prototypes may not be the best prototypes).

Next, we discuss the translation of the model to a Bayesian network. As suggested by Koller & Sahami (1997), we can represent categories as discrete nodes that serve as parents of Gaussian feature nodes. Thus we specify the statistics for each feature and query the category node given the feature evidence. The problem to be worked out is to implement an unsupervised learning algorithm that updates the values that the category node can take on as more evidence arrives. Fig. 7 shows an early attempt at integration in which a superordinate parent category is represented just as another evidence node. Here we have assumed independence of each feature value. The superordinate category node “large” is a binary representation of a “decision” based on current feature estimates. Note that prototype values are kept implicitly within the distribution for the superordinate node. It may be better to use decision nodes for the superordinate category nodes, but we’re not sure how inference would work.

Results

Fig. 1 shows the size of the category partition, the categories given as examples, and the categories learned using no model merging and no feature comparison. For simplicity, only the size feature values of the categories are shown. Similar results can be plotted for the cuteness and ferocity dimensions. Note that a lot of categories were formed despite data incorporation, because no pruning is allowed. Fig. 2 shows the categories learned using model merging alone and Fig. 3 shows the categories learned using both model merging and feature comparisons. In general, category partitions learned using feature comparisons tend to generalize a bit more, so that we typically get a smaller number of categories. Note that one obvious problem with the model merging algorithm is that it tends to give inaccurate values for animals whose size is small.

We can change the parameters for both the model merging and the feature comparison algorithms. Fig. 4 and Fig. 5 shows the results for example runs for which the algorithms took into account the size of the model more and made greater changes to feature values during comparison. Note that both learned categories are very good. Note also that with feature comparison, we model the values for the 3rd and 7th categories better than with model merging alone. Over many iterations of the experiment, we find that feature comparison tends to do wonders when *pef* is set to be small (P 0.1). In that case, feature comparison tends to generalize more than model merging alone.

Discussion

Our Bayesian model merging algorithm applied to category-acquisition is not without its faults. For example, we are assuming that in the presence of small sample sizes, humans still believe that the distribution of feature values in the real world is approximately normal for each distinct animal. More reasonable approached involving the Dirichlet density can be found in Anderson (1990). We are also assuming that each feature comparison task occurs regularly and that all features of a category are compared. In the real world, this is rarely the case. We might be led into comparing, for example, sizes more than cuteness, and cats and dogs more often than polar bears and lady bugs. A proper model for comparison tasks in early language acquisition would involve modeling the world in which the child lives, which is beyond our present scope and capabilities. Note also that we are assuming the independence of each feature given the category, which cannot be a true model of the world. Finally, the model assumes that model merging and feature comparison happens independently. Note, however, that depending on our current comparison task, we might be more or less willing to merge disjoint categories. For example, we may be asked to compare cardinals and blue jays. We don't have much of an idea of what they are and might consider them to be close to each other in size. Might they also not be in one category?

One other proposal would be to treat the problem as minimization of Boolean complexity given features and examples. The best model is that which minimizes the minimal formula associated with the disjunctive normal form of the evidence written with conjunction among features and disjunction among examples. We haven't worked this out, however.

The Bayesian network representation needs to be explored thoroughly. We need to work out the way learning of category values will work for a Bayes net. Superordinate category representation is another area for improvement. One idea is to use separate nodes for each category value (ie. animal name). Then all we have to do is prune the network as more evidence arrives. The problem is that each node would be the parent of the feature values. Hence each augmentation of a feature adds O(k) links, where k is the number of values a category can take on. We should also allow feature values to link with each other, thereby removing the feature value independence assumption.

We presented a Bayesian approach to learning category representation. We saw that learning is facilitated by implicit interactions with the real world. Here, we considered feature comparison and priming. The principle, however, is more general. In modeling domain knowledge, we need to consider the various contexts in which a name can occur and the various interpretations the name can take on with respect to superordinate category prototypes.

References

Anderson, J. R. 1991. The adaptive nature of human categorization. *Psychological Review*, 98(3):409-429.

Banks, W. P.; Fujii, M.; and Kayra-Stuart, F. 1976. Semantic congruity effects in comparative judgments of magnitudes of digits. *Journal of Experimental Psychology: Human Perception and Performance*, 2:435-447.

Koller, D. and Sahami, M. 1997. Hierarchically classifying documents using very few words. In *Proc. ICML-97*, 170-178.

Luo, R. 2000. Response interference in categorical judgment tasks. Online http://inst.eecs.berkeley.edu/~rluo/categ_learn_doc/perc&cog_np&sc-revised-text.doc.

Schooler, L. J.; Shiffrin, R. M.; and Raaijmakers, J. G. W. 2001. A Bayesian model for implicit effects in perceptual identification. *Psychological Review*, 108(1):257-272.

Appendix

Matlab code for the algorithm can be found at http//inst.eecs.berkeley.edu/~rluo/categ_learn_doc/categ_learn.zip.

Run the file learn_animal_categories in Matlab to begin experimenting. You should have the BNT toolkit installed http://www.cs.berkeley.edu/~murphyk/Bayes/request.html.

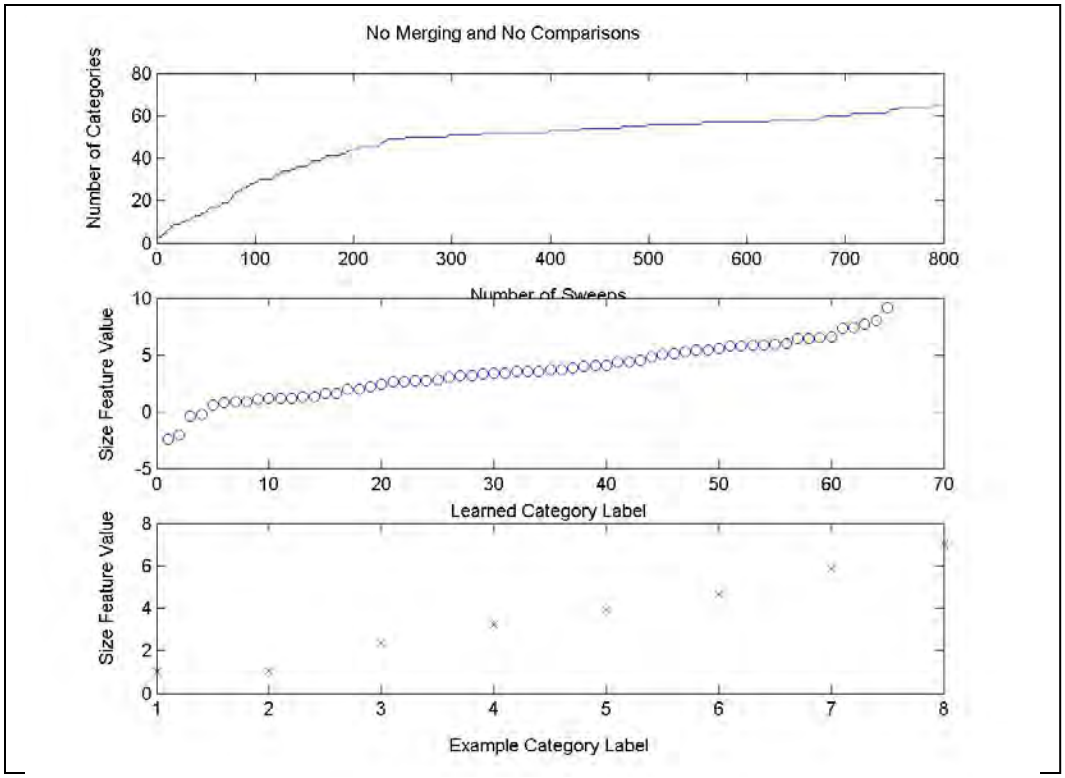


Fig 1. Size features of categories learned with no merges and no compares.

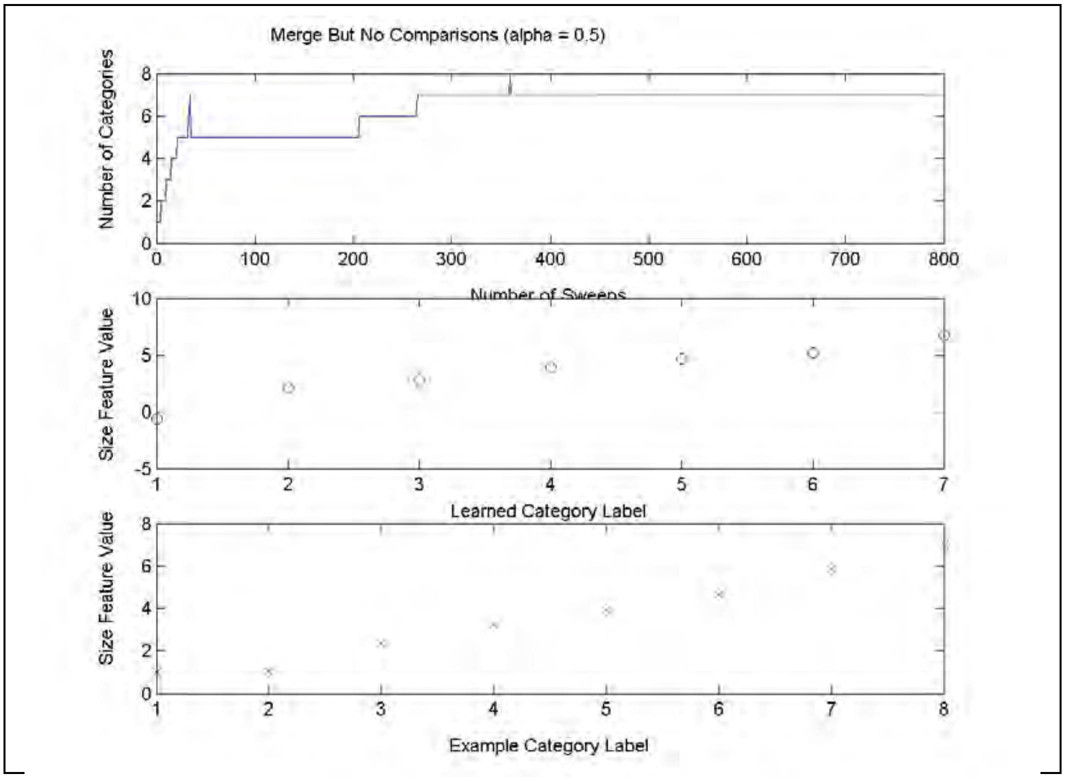


Fig 2. Size features of categories learned with merges but no compares (expt 1).

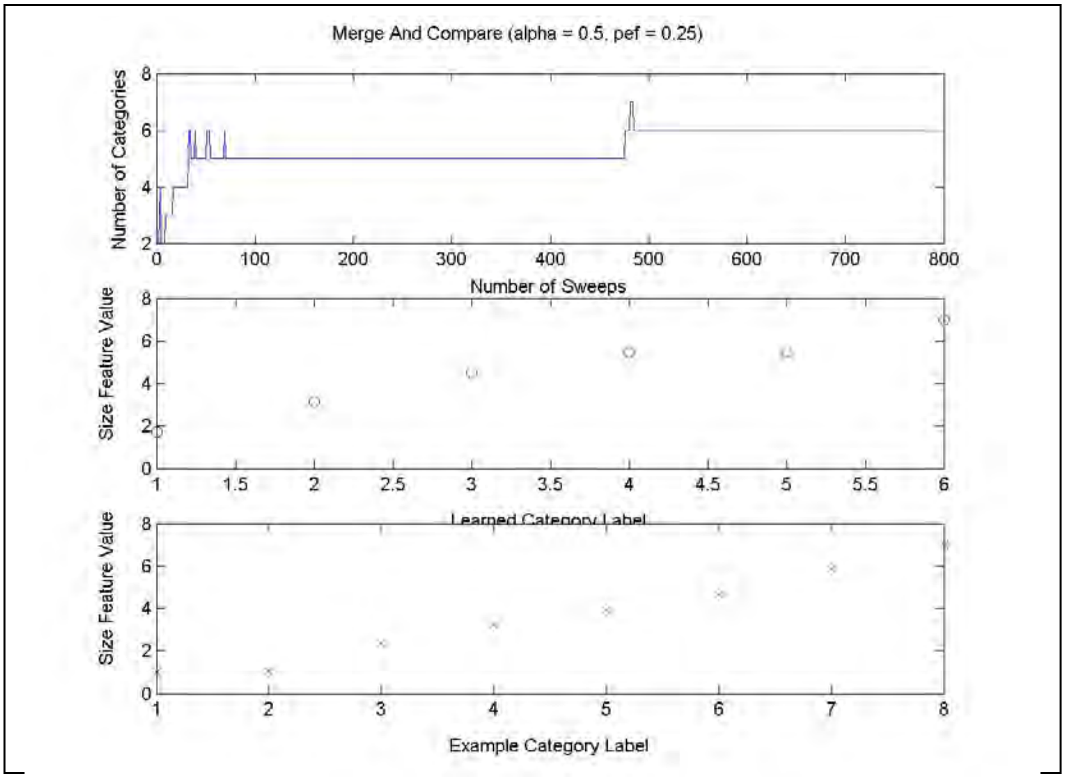


Fig 3. Size features of categories learned with merges and compares (expt 1).

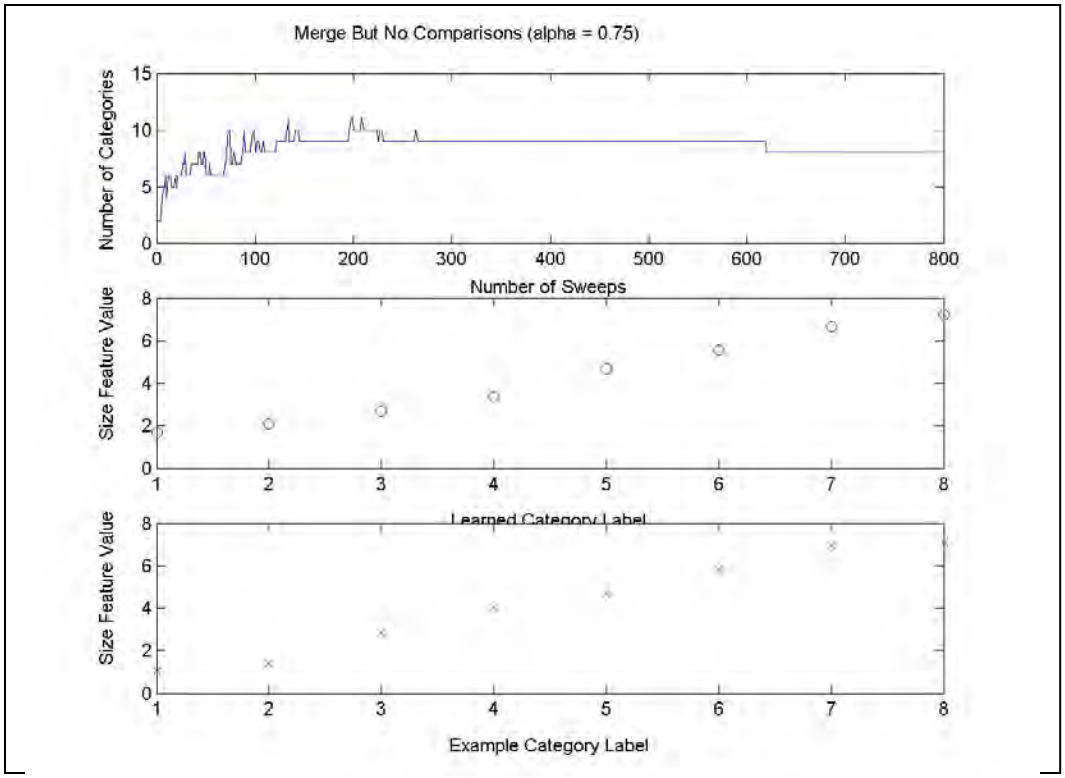


Fig 4. Size features of categories learned with merges but no compares (expt 2).

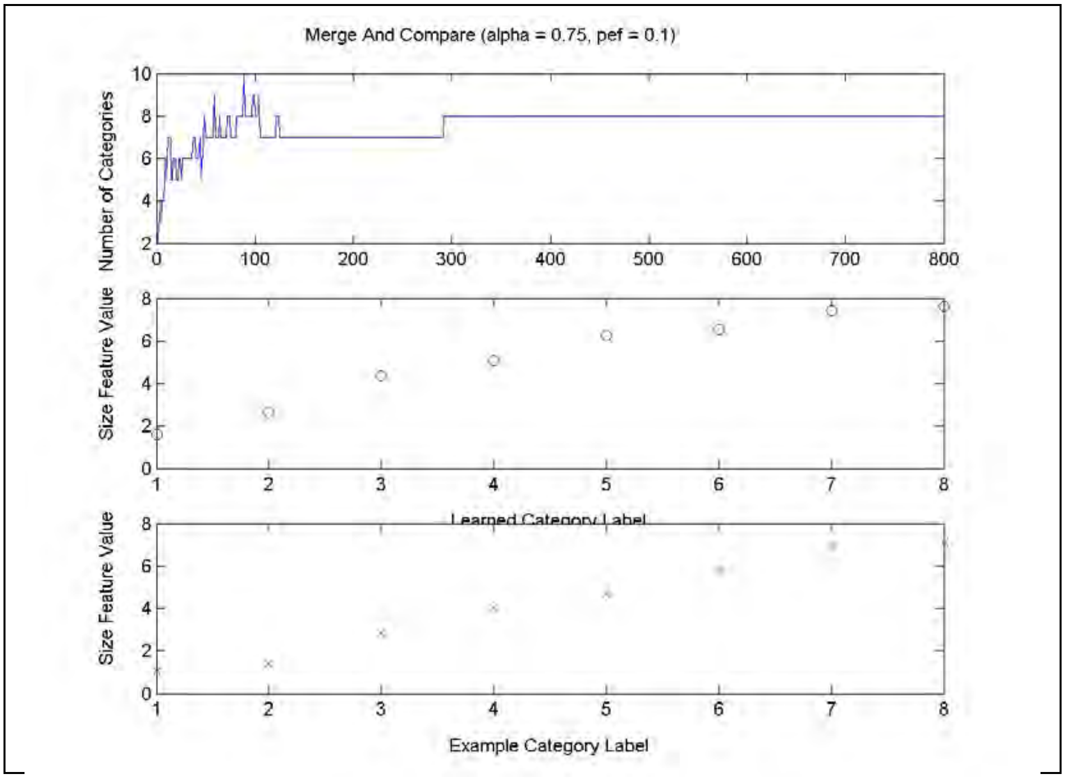


Fig 5. Size features of categories learned with merges and compares (expt 2).

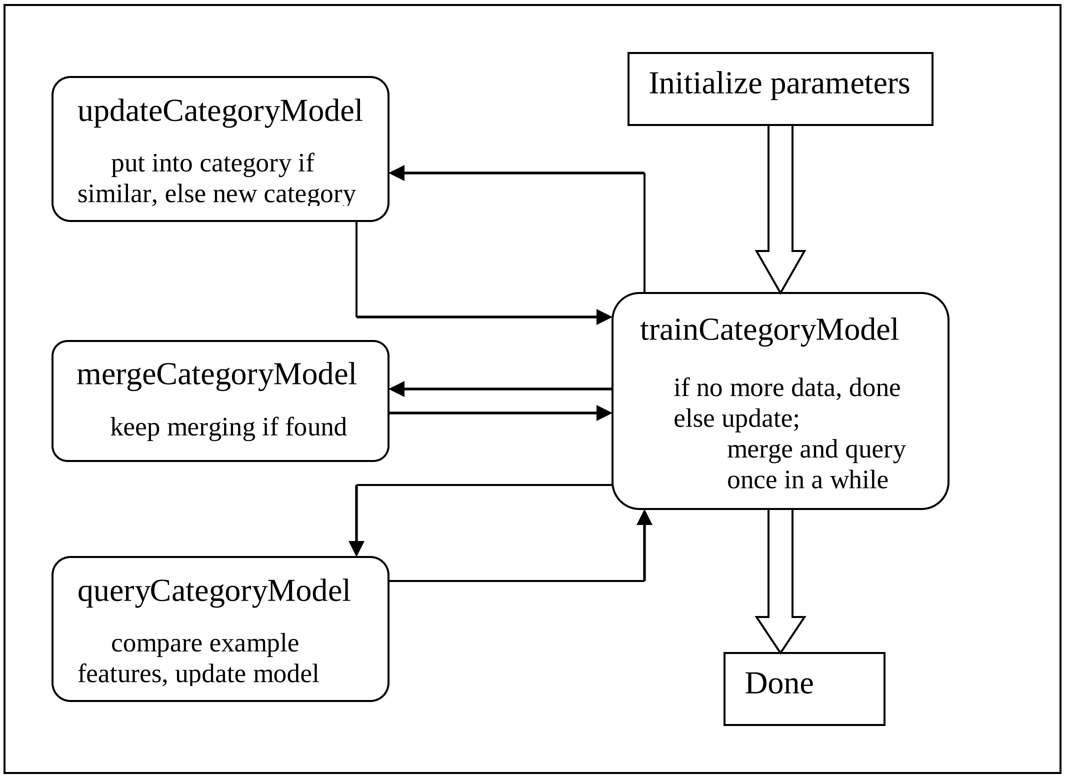

updateCategoryModel
put into category if similar, else new category
Initialize parameters
trainCategoryModel
if no more data, done else update;
merge and query once in a while
mergeCategoryModel
keep merging if found
queryCategoryModel
compare example features, update model
Done

Fig 6. High-level algorithm overview.

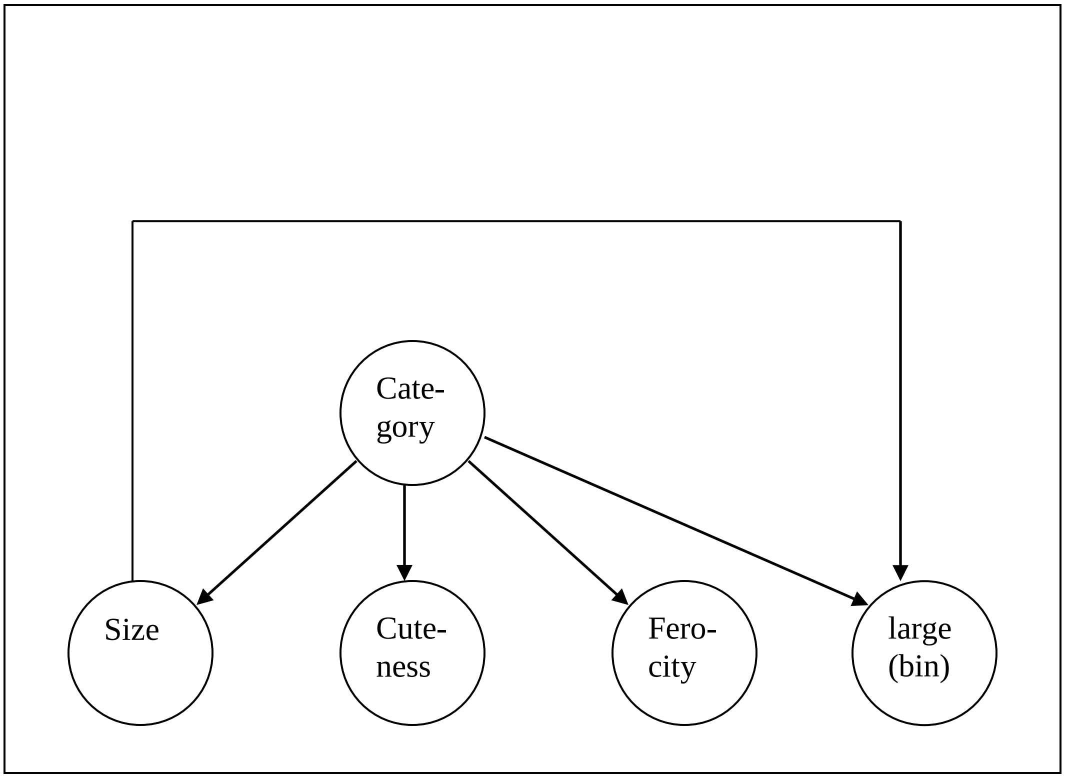


Fig 7. A Bayesian network implementation.

Response Interference in Categorical Judgment Tasks

Ray Luo

with William Banks (Pomona College)

University of California, Berkeley[1]

[1] This paper was part of the work pursued while a student of Harvey Mudd College. Revised and reedited for presentation.

Abstract

Response delay and facilitation in categorical judgment tasks were studied under the framework of two potentially competing theories. The negative priming model and the semantic coding model were presented as possible explanations of response interference. To test the models, subjects were asked to either choose the larger or choose the smaller of a pair of animals. Reaction time (RT) differences were found when subjects were required to choose an animal in the target trial that was previously ignored in the prime trial. For example, when subjects were asked to perform the same categorical judgment (e.g. choose larger) in both the prime and the target trial, mean (target trial RT minus prime trial RT) was found to be 29.9 msec. When subjects were asked to respond to the smaller of two small animals that was previously ignored, mean RT difference was found to be –152.1. Results indicate that neither negative priming nor semantic coding fully explains the data. A related effect known as semantic congruity, however, was found in the data and provided support for the coding model.

Response Interference in Categorical Judgment Tasks

We make binary categorical decisions almost continuously. Where should I move to? LA or San Francisco? Which day was warmer? Yesterday or today? Who is faster? Dan or Dave? Which is more important? Homework or video game? Natural questions arise in any attempt to understand human decision making processes. The two most debated research questions are: What type of information is represented by an encoding? And how does the processed information affect categorical response? This paper examines both of these traditional questions from a different perspective. Instead of focusing on "What?" and "How?" we will concern ourselves with the question "When?" namely, when do processing and response occur.

To describe categorical judgments more rigorously, two models of binary decision-making are presented and compared under a standard experimental procedure designed to measure reaction time. The first model was first used to explain a phenomenon observed in rapid encoding-response experiments. Tipper (1985) presented his subjects with pictures of two distinct objects, one in each color, and asked them to pay attention to only the object in a specified color. In the prime trial, for example, subjects might be presented with a green trumpet and a red anchor and might be asked to identify the green object. In the target trial, the previously ignored anchor would now appear in green along with some other arbitrary red object. It was found that in the target trial, reaction times (RTs) were significantly slower. Tipper explained this phenomenon by postulating a "negative priming" effect in which previously rejected stimuli suppressed subsequent response involving those stimuli. This model claims that information (such as object identity) is processed by mechanisms that inhibit irrelevant details.

In contrast with negative priming's explanation of rapid object-incoherent response tasks is a different model that was first used to explain semantic information encoding during categorical judgment tasks. Banks et al. were the first to propose a semantic coding model to explain the results of symbolic comparative experiments (e.g. Banks, 1977). According this hypothesis, three types of mental structures are responsible for comparative judgments: database, processing codes, and processing mechanisms. Banks suggested that when a pair of stimuli is first presented, the processing mechanisms “generate processing codes from the data base and manipulate them until they match the previously stored and coded instructions” (p. 131). For example, suppose that the subject is presented with a table and a car and asked to choose the more expensive object. After this prime trial, the car is coded with E+ for expensiveness and table is coded with I+ for inexpensive, since an object can only be expensive or cheap, and not both. In the target trial, the same table is presented along with a box of crayons. When asked to choose the more expensive object, the response will be table, but the reaction time (RT) will be longer since the subject must convert the I+ encoding to an E+ encoding. Note that this is also what negative priming predicts: table is rejected in the prime trial, thus inhibiting the response in the target trial.

Now suppose instead that the subject is presented with the table and a diamond in the target trial. If she is asked to choose the *less* expensive object, the semantic coding model predicts a faster RT (since table is already coded as I+). Negative priming, however, suggests that the response ought to be slow because, again, the previously rejected stimulus (table) suppresses subsequent response involving the stimulus. Here, then, is the trial that distinguishes the two models.

Additional Considerations

Although each conjecture stated above gives a different prediction of the observed RT difference between target and prime trials, we cannot conclude that they contradict one another.

Negative priming was traditionally used to model rapid, previously inhibited responses in the probe trial while semantic coding was used to explain memory mismatch between response and question posed. Studies have shown (Banks, Fujii & Kayra-Stuart, 1976) that in a set of trials involving the comparison of magnitudes of a pair of digits, subjects were faster when asked to pick the larger of two large digits (e.g. 8 and 9) and slower when asked to pick the smaller of two large digits. Similar effects were observed for smaller digits. It is possible that long term effects of repeatedly coding large digits as large and small digits as small contribute to this phenomenon. To distinguish the delay associated with judgment-instruction mismatch from semantic coding effects due to prime trial stimulation, we will refer to the phenomenon described above as semantic congruity, as Banks et al. have done.

Another complication involves the size differences between a pair of stimuli. Moyer and Landuer (1967) found that RTs associated with digit comparison tasks increased as the numerical difference between the two digits decreased. For example, subjects gave quick responses when asked to choose the larger of 3 and 8, and slow responses when asked to choose the larger of 6 and 7. This phenomenon, which we refer to as the distance effect, needs to be accounted for even under situations when semantic congruity is observed.

Instruction mismatch between prime and target trials can also influence the observed RT. We expect that subjects would need additional processing time to encode a new instruction during target trial (e.g. Which is smaller?) relative to the old instruction given in the prime trial (Which is larger?). Care must be taken to ensure that instruction mismatch is accounted for by appropriate controls.

We also need to control for the RT decrease or increase due to expectancy. When subjects are presented during the target trial with a stimulus that has already appeared in the prime trial, they may respond differently in different situations. They could, for example, find the coincidence bizarre. In this case, RTs would be longer than normal. They could also find that the processing is easier perform (since they already coded for the stimulus in a previous trial). In this case, RTs would be shorter than normal. Expectancy is an important contributor to observed RTs; another form of control is needed to account for it.

Recent Developments

Confrontation between negative priming and semantic coding was never considered in the literature because the former was used to model rapid inhibitory decision-making processes and the latter was used to predict RTs for a single trial due to coding mismatch in previous trials. Semantic mismatch was generally regarded as the result of discrepancies between the representation of the stimulus in the present trial and its representation in short or long term *memory*. Negative priming, on the other hand, modeled small RT differences in fast responses that rely on the speed of mental *reflex*. In other words, subjects were forced to analyze even the rejected stimulus in a semantic coding task, but were encouraged to pay no attention to the rejected stimulus in a negative priming task. Recall that in Tipper's original experiments, subjects were not required to pay attention to the green object in order to *recognize* the red object.

Recently, the interpretation of negative priming given above has been dispelled by MacDonald, Joordens, and Seergobin (1999). In a series of experiments designed to test negative priming under conditions where attention to distractors was necessary, MacDonald et al. found that negative priming effects were enhanced. Subjects were asked to discriminate (not just recognize) the larger of two animals in both the prime and the probe trial. Distractors in the prime trial became targets in the probe trial. It was found that probe trial RTs were significantly longer than those in traditional negative priming experiments (e.g. Tipper, 1985, using Stroop color words). MacDonald et al. believed that their findings challenged “the basic assumption that the negative priming effect arises because the critical item was ignored or not attended to on the prime trial” (1999). We may ask, however, “Are these RT differences due to negative priming, or some other effect?” To qualify their interpretation, the present study examines a task in which both negative priming and semantic coding make predictions about outcome RTs. The task even allows us to distinguish between the two models because they give different predictions for a specific type of trial.

Method

Subjects

Six students at the Claremont Colleges (Pomona College and Harvey Mudd College, Claremont, California) participated in the experiment. Two of the students generated data for qualitative analysis only and did not sit through the entire experiment. The other subjects each performed the experiment three times, the first run being a practice exercise. All participants were sufficiently fluent in English to understand the stimuli displays. No subject had any difficulty seeing the stimulus array.

Apparatus

A Macintosh computer from the Pomona College psychology lab was used to display the stimuli and record the RTs of the response. The Superlab program was used to perform the experiment.

Procedure

Before the trials, subjects were given a list of all animal names used in the experiment:

mouse, dog, horse, elephant, and whale. They were asked to verify the size order of the animals by ranking them in increasing integers of 1 to 5 (1 being the smallest). This was done to ensure familiarity of the subjects with the stimulus array. During each run, general instructions were followed by a total of 66 stimulus displays. Each successive display was shown right after the response to the previous display was received. Instructions for each pair of stimulus (i.e. choose smaller or choose larger) were shown at the center of the top of the screen; two animal names were shown at the left and right sides of the screen. The subjects were asked to press the "z" and "m" buttons to select the leftmost and rightmost animal, respectively.

Four types of trials were presented during each run. Fig. 1 shows a typical pair of prime and probe displays in the first type of trials (reliability trials). Note that negative priming (NP) and semantic coding (SC) give the same prediction in reliability trials. NP predicts a RT delay in the probe trial due to previous inhibition and SC predicts a RT delay in the probe trial due to coding mismatch. The second type of trials (discrimination trials) presented displays similar to the ones shown in Fig. 2. In this case, NP still predicts a RT delay during the probe trial. SC, however, predicts a shorter RT in the probe trial. As Fig. 2 shows, the distractor in the prime trial (horse) is coded S+. When asked in the probe trial for the smaller of whale and horse, the previously S+ encoding of horse ought to facilitate the response. Note, however, that in nature, horse is not generally speaking a small animal. Thus the semantic congruity effect predicts a slower RT just for the probe trial since the subject is asked to choose the smaller of two large animals. This illustrates a peculiarity in the discrimination trials. As Fig. 3 shows, displays involving a match in the instruction-judgment relationship (i.e. semantically congruent trials) were also used. In these pairs of prime and target trials, both semantic coding and semantic congruity predict a faster RT during probe response. From Fig. 3, dog was first coded as S+ in the prime trial. During the probe trial, the previously rejected smaller animal (dog) was asked for. In this case, dog *is* a small animal, so semantic congruity predicts a fast RT. Semantic congruity was not explicitly considered in the reliability trials. In that case, however, semantically congruent pairs were not systematically distributed. In calculating mean RT differences between prime and probe trials (i.e. mean(probe trial RT − prime trial RT)), we expect the random placement of semantically congruent and incongruent pairs into probe and prime trials to cancel out the net semantic congruity effect.

The remaining two types of trials were part of a network of controls designed to isolate the difference in prediction given by NP, SC, and semantic congruity. Typical displays of control trials are shown in Fig. 4. Notice that prime and target stimuli are all different; no linkage between prime and target trials are predicted. To account for the expectancy effect, a new type of control known as associated control was introduced (see Fig. 5). In the associated control trials, the previously accepted animal was again asked for in the probe trial. If expectancy is indeed a factor, we expect RT differences between the responses given in prime and probe trials. We then subtract this difference from reliability and discrimination trial RTs to get the net effect of NP or SC.

To control for the distance effect, only animals similar in size were used in the reliability and discrimination trials. For example, elephant may be compared with horse or dog, but not with mouse. Similarly, whale would be compared with horse or elephant, but not with dog. The distance effect was not explicitly accounted for in the control and associated control trials although the placement of long distance vs. short distance pairs was randomized. If the distance effect were important in these trials, they would cancel each other when mean RT differences are calculated, since it is equally likely for a mouse-whale pairing to occur in the prime or the probe trial. Note also that each animal (mouse, dog, horse, elephant, and whale) is sufficiently distinguishable from each other in terms of size. There is also no clustering of animals into a particular size range. This ensures that the distance effect is predictable and easy to account for.

To control for instruction mismatch between prime and target trials, we allowed some control and associated control trials to display prime and probe arrays that differed in instructions (i.e. Which is larger? vs. Which is smaller?). The RT delay associated with these trials should be subtracted from RT differences found in discrimination trials, because all discrimination trial stimuli involved changes in instruction between prime and target displays.

Serial positioning of the stimuli was controlled by random placement of the animal names in the left and right sides of the display. As a consequence, correct keystroke responses for each display was also randomized (i.e. the probability of having the same correct keystroke response for both prime and probe trials is 0.5). Residual encoding was controlled by displaying a normative pair of stimuli before each trial. These normative displays contained animal names that matched none of the three animals names found in each reliability, discrimination, and associated control trials. The idea is that normative displays “normalizes” the playing field before each trial, so that RT differences can reliably be attributed to the prime and target trial responses. For the control trials, a normative display consisted of the animal ignored in the probe trial and another animal that appears in neither the prime nor the probe trials. Since there were only five animal names available, we felt that having an ignored name in the normative display would make virtually no difference in the mean RTs.

Another variable that we attempted to control was the time delay between successive displays. In half of the runs, subjects were given the standard treatment of randomized blocks of six reliability trials, six discrimination trials, six control trials, and four associated control trials.

Each trial consisted of three displays (normative, prime, probe) in sequence. The order of appearance of the trials was randomized. Subjects were not given any breaks between successive stimuli pairs. In another half of the runs, each trial was preceded by a 2.5 second blank display followed by an instruction which asked the subject to press both “m” and “z” keys simultaneously to continue. The effect of these two extra displays was to slow down the experiment and eliminate any possible residual encoding left over from the previous trial. By comparing the results found in the fast runs against the slow runs, we can examine the effect of time delay between successive trials.

Results

In this experiment, we are interested in the RT difference (ΔRT) given by probe trial RT minus prime trial RT. It this number is positive, a RT delay in the target trial is observed; if this number is negative, task facilitation in the target trial is observed. Table 1 shows the mean ΔRT described above for each type of trial, averaged from about 20 to 40 samples each. Extreme outliers are excluded. Notice that ΔRT for the discrimination trials is greater than the ΔRT for the reliability trials. This, of course, does not take into account instruction mismatch effects, expectancy, or semantic congruity. Note also that the discrimination trials involving semantically congruent target responses (discrimination trials 3 and 5) display a large facilitation in the probe trial response. ΔRTs for instruction mismatch trials (control trials 2 and 5, associated control trial 1) are also calculated. Uncertainties for the data are given by standard deviation of the mean of the sample. Fig. 6 shows a box plot of the set of data. Note the greater variability of ΔRTs for semantically congruent discrimination probe trials. This is due to the smaller sample size for these trials, which are a subset of the discrimination trials.

To find the actual effects on ΔRT due to each type of stimuli, we first subtract the mean control trial ΔRT from the mean ΔRTs for reliability, discrimination, associated control, and semantically congruent trials (instruction mismatch trials are essentially a subset of the control trials). The reason mean ΔRT for control trials is slightly negative is that subjects felt more comfortable with each successive display during any given trial (the effect is most conspicuous for the slow runs). Next, we subtract the mean associated control ΔRT from the result calculated as above. The mean ΔRT for associated control trials is slightly positive (since we had to subtract control trial mean ΔRT from it to get the net effect) indicating a RT delay due to expectancy. Finally, we subtract the mean instruction mismatch ΔRT from the resulting mean ΔRTs of discrimination and semantically congruent trials found above (we do not subtract from the reliability trials because for these trials, all instructions were congruent for prime and probe displays). Doing all the calculations we just mentioned, we obtain the net ΔRT due to specific types of trial stimuli. These results are given in Table 2 and illustrated graphically in Fig. 7. In Table 2, the error given for each mean ΔRT is found by propagating from the errors in Table1.

From Fig. 7, we observe that negative priming cannot explain the semantic congruity effect. For example, given two small animals, subjects will choose the smaller of the two quickly despite what occurs in a previous trial. Negative priming does not explain ΔRT for discrimination trials particularly well since the mean ΔRT is nearly zero, indicating neither inhibition nor facilitation. Semantic coding does not explain the discrimination trial ΔRT very well either since the expected facilitation does not occur. In any case, both theories support the findings in the reliability trials since mean RTs are increased in the probe trial as expected. The most statistically significant piece of information in Table 2 is the large target facilitation found in discrimination trials where targets and instructions are semantically congruent (e.g. Fig. 3).

This indicates that even in a fast experimental task, previous semantic encoding is extremely important. Every time we see the words elephant or whale, for example, we automatically retrieve out of our memory the concept of large size. It appears that semantic congruity is a part of every judgment the subject makes. The influence of development and previous coding seems strong even for a rapid judgment task.

Fast vs. Slow Runs

Further analysis of the experimental data reveals some surprising results. When we separate the mean ΔRTs for slow and fast runs of the experiment, we find that, contrary to expectations, reliability trial mean ΔRT decreases and discrimination trial mean ΔRT increases. Semantically congruent discrimination trial mean ΔRT stays about the same. Negative priming can now better explain the discrimination data and semantic coding falls apart for the same data. Both results, however, are extremely variable. Many of the effects observed may simply be due to random chance. We therefore concluded that to alter ΔRTs significantly, greater change of speed is necessary for this particular task.

Discussion

According to MacDonald et al. negative priming effects are increased for tasks where attention to distractors is required. In this experiment, we have challenged the assumption that the RT increases observed for these tasks are actually due to negative priming. Negative priming predicts that previously ignored stimuli are inhibited in the probe trial. We found, however, that in tasks such as those given by Fig. 3, response to ignored stimulus is actually significantly enhanced in the probe trial. If negative priming is to explain the wider scope associated with semantic interference tasks, it must append to its simple theoretical framework an account of phenomena such as the semantic congruity effect.

According to Banks et al. semantic coding is responsible for the cross-trial interference found in a categorical judgment task. While this experiment does not contradict the semantic coding hypothesis for successive trials, it does offer some insight into possible improvements of the theory. It is noted that long-term coding effects are more important than immediate encoding of information. In fact, between-trials encoding seems to neither facilitate nor delay probe trial RTs. This experiment supports the claim, however, that information in a categorical judgment task is indeed coded by categorical variables such as L+ and S+.

Implications

Looking again at the data from Table 2, we must conclude that semantic coding or learning takes place slowly. In a rapid decision-making task, information is retrieved from memory, and not necessarily encoded from previous trials. Hence the effect of semantic congruity is significant but the effect of semantic coding is not. We can understand the data by postulating an automatic mechanism for semantic processing. Every time we see the word whale or elephant, L+ encoding is retrieved from memory, and not directly encoded. The next time we see whale or elephant, L+ is retrieved again. This experiment supports the idea that the mind does not keep a handful of information at its fingertips, even in a rapid binary decision task. If it did, we would see large semantic coding effects or perhaps even negative priming. Instead, the mind asks for the encoding again and again, comparing it each time to relevant information and discarding the result as time passes. This new model of code processing is given in Fig. 8.

Under this model of memory processing, we would explain a typical trial as follows. First the subject is told about the category of discrimination, in this case, size. This shifts the memory system into a binary coding mode involving two categories: L+ and S+. A whale and a horse are presented, with instructions asking for the larger animal. As soon as the word whale is read, the binary coding machine (mind) automatically processes whale as L+. Similarly horse is processed as L+. Now the machine must distinguish between the larger of two L+s. This is where the bulk of the processing time is distributed. After generating a response, most of the encoded information is thrown out since a new pair of stimuli now appears. The process continues until the end of the experiment. How, then, does the mind learn (and remember certain codes) ? It learns by focusing attention to the task. It must process and encode a specific piece of information continuously. For example, if it must learn that a “quoma” is “large,” then it must associate quoma with the L+ binary encoding. If the association between quoma and large is weak, RT for the next target trial quoma will be long. For familiar objects, however, the machine already has encoded information available; it need not encode additional information.

Qualification

We have been concentrating on the large drop in target RT associated with semantically congruent discrimination trials. One important qualification must be put in place. When we examine the 5th discrimination trials, we note that not only is the probe display semantically congruent, but that the prime display is semantically congruent as well (see Fig. 3). (This does not occur in the other semantically congruent discrimination trial.) Yet we observe an extremely negative ΔRT for these trials? If semantic congruity is the sole factor in determining RTs, then trial 5 ΔRTs should be zero. This suggests that there is a complex interaction between semantic coding and semantic congruity. It appears that long term and short term processing can influence each other in complicated ways. If neither semantic coding nor semantic congruity can fully explain the data, then we need a new theory that models the interactions between long term and short term semantic coding. The solution to this problem may lie in the analogous interactions between automatic and controlled processing.

References

Banks, W.P. Encoding and processing of symbolic information in comparative judgments. In G. H. Bower (Ed). (1977). *The Psychology of Learning and Motivation*, Vol 2. New York: Academic Press.

Banks, W. P., Fujii, M., & Kayra-Stuart, F. (1976). Semantic congruity effects in comparative judgments of magnitudes of digits. *Journal of Experimental Psychology: Human Perception and Performance*, 2, 435-447.

MacDonald, P. A., Joordens, S., & Seergobin, K. N. (1999). Negative priming effects that are larger than a breadbox: Attention to distractor does not eliminate negative priming, it enhances it. *Memory & Cognition*, 27(2), 197-207.

Moyer, R. S., & Landauer, T. K. (1967). The time required for judgments of numerical inequality. *Nature*, 215, 1519-1520.

Tipper, S. D. (1985). The negative priming effect: Inhibitory priming by ignored objects. *Quarterly Journal of Experimental Psychology*, 37A, 571-590.

Tipper, S. D., & Cranston, M. (1985). Selective attention and priming: Inhibitory and facilitatory effects of ignored primes. *Quarterly Journal of Experimental Psychology*, 37A, 591-611.

Tables and Figures

Type of Trial	Mean RT Difference	Error in Mean RT Difference
Reliability	23.3	41
Discrimination	58	47
Control	-37.8	37
Associated Control	-6.6	39
Semantically Congruent	-95.2	90
Instruction Mismatch	63.5	46

Table 1: Mean of probe trial RT minus prime trial RT for each type of trial. Note that semantically congruent trials are a subset of discrimination trials and that instruction mismatch trials are a subset of control and associated control trials.

Type of Trial	Net Mean RT Difference	Error in Net Mean RT Difference
Reliability	29.9	77
Discrimination	1.1	92
Semantically Congruent	-152.1	120

Table 2: Net mean RT differences calculated by taking into account appropriate controls, expectancy and instruction mismatch. This gives the net effect of varying each probe trial variable. Note that only semantically congruent trial data is significant with respect to the errors.

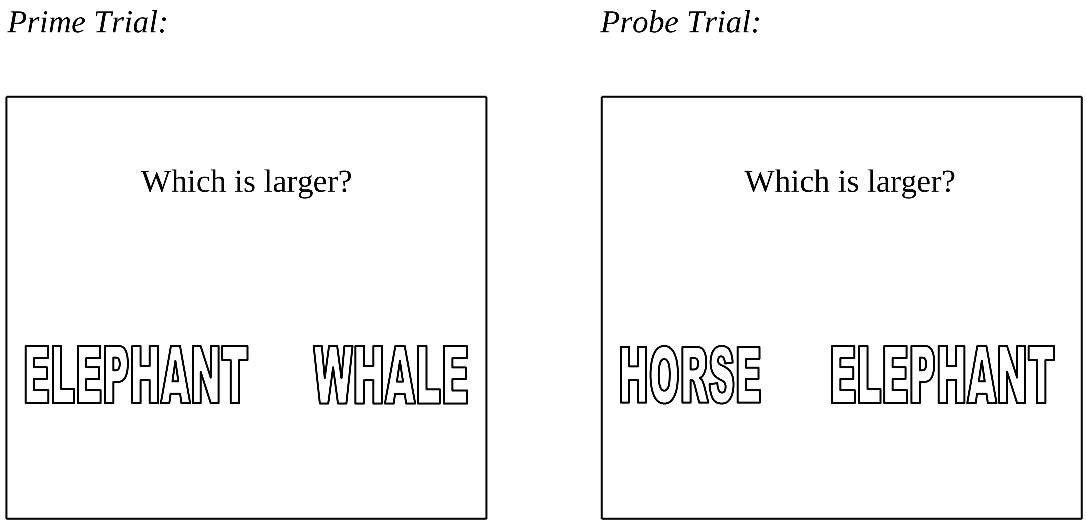


Figure 1: Typical displays in prime and probe trials in a reliability trial. Predictions of each model:
Negative priming: elephant ignored in prime, response to elephant slow in probe.
Semantic coding: elephant coded as S+ in prime, response to elephant slow in probe since we ask for the larger animal.

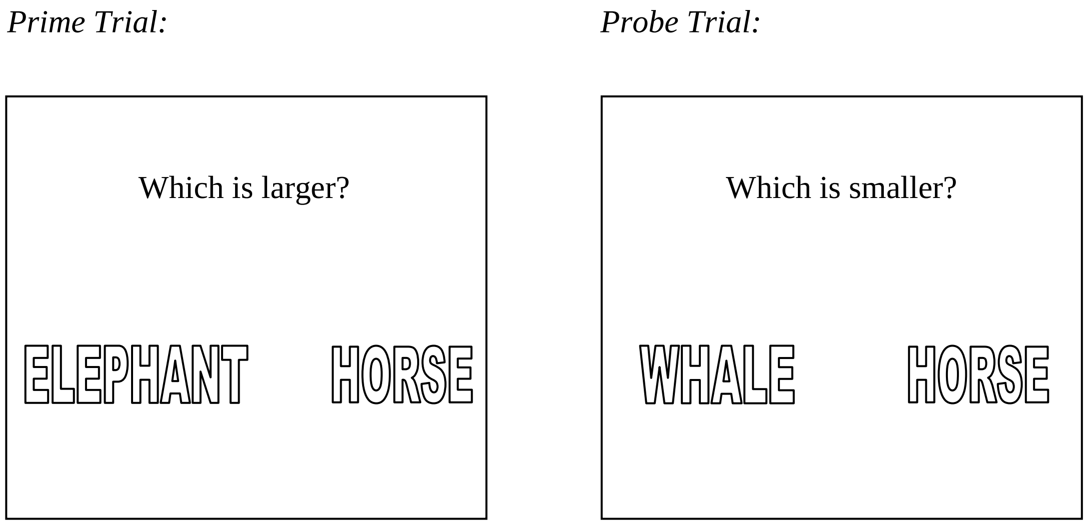


Figure 2: Typical displays in prime and probe trials in a discrimination trial. Predictions are:
Negative priming: horse ignored in prime, response to horse slow in probe.
Semantic coding: horse coded as S+ in prime, response to horse fast in probe since we ask for the smaller animal.

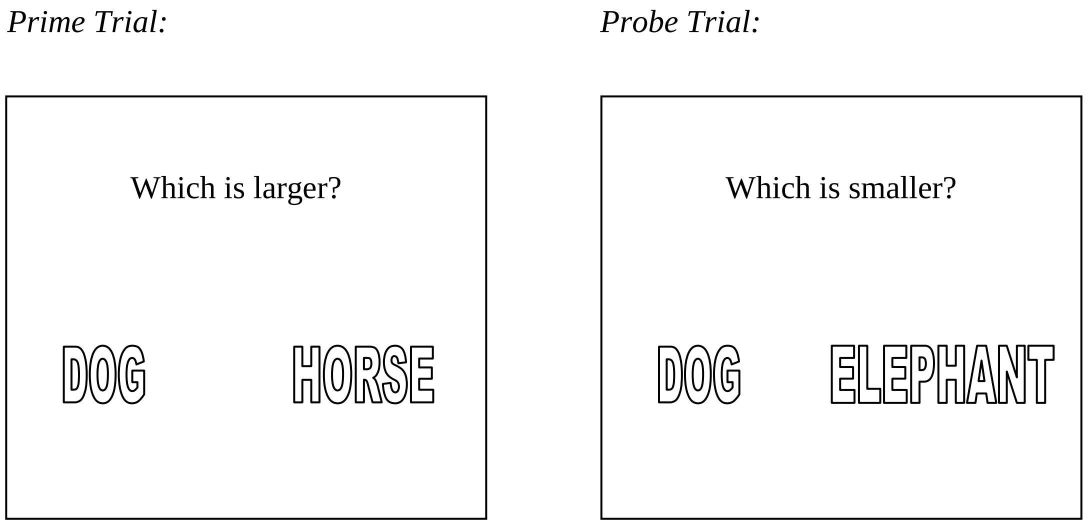


Figure 3: Typical displays in prime and probe trials in a *semantically congruent* discrimination trial.
Negative priming: dog ignored in prime, response to dog slow in probe.
Semantic coding: : dog coded as S+ in prime, response to dog fast in probe since we ask for the smaller animal. (Due to semantic congruity, response should be very fast.)

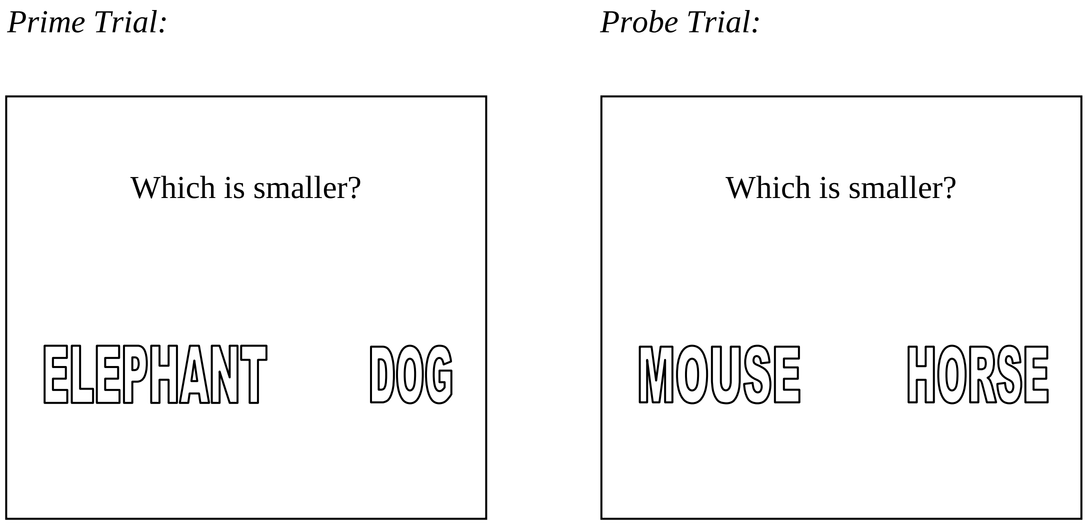


Figure 4: Typical displays in prime and probe trials in a control trial. Predictions for each model:
Negative priming: elephant ignored in prime, no effect on response to horse in probe.
Semantic coding: elephant coded as L+ in prime, no effect on response to horse in probe.
(Control is used to compute net effect of trial stimuli arrangements.)

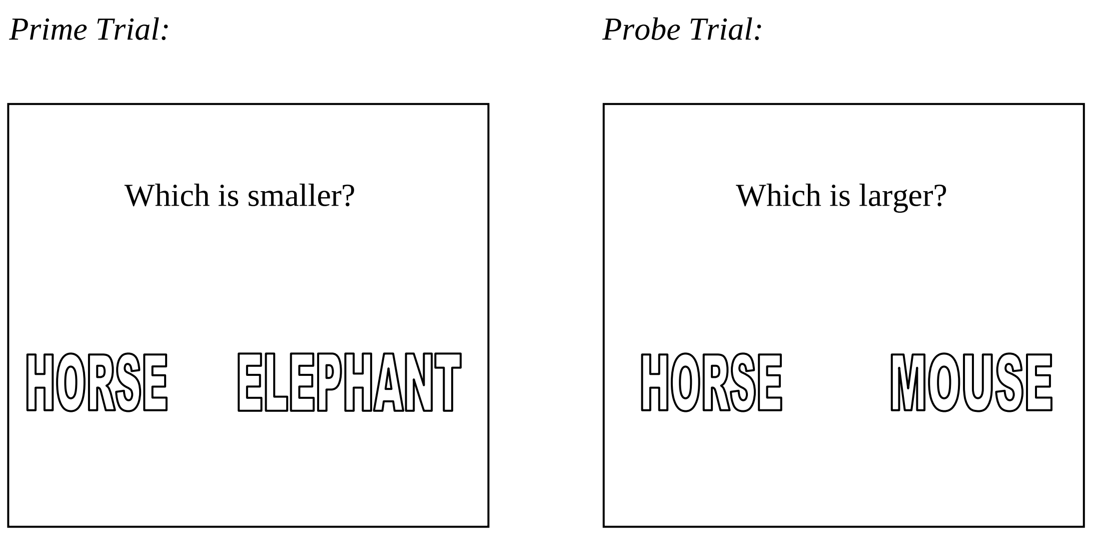


Figure 5: Typical displays in prime and probe trials in an *associated* control trial. Predictions are:
Negative priming: elephant ignored in prime, no effect on response to horse in probe.
Semantic coding: horse coded as S+ in prime, response to horse slow in probe since we ask for the larger animal.

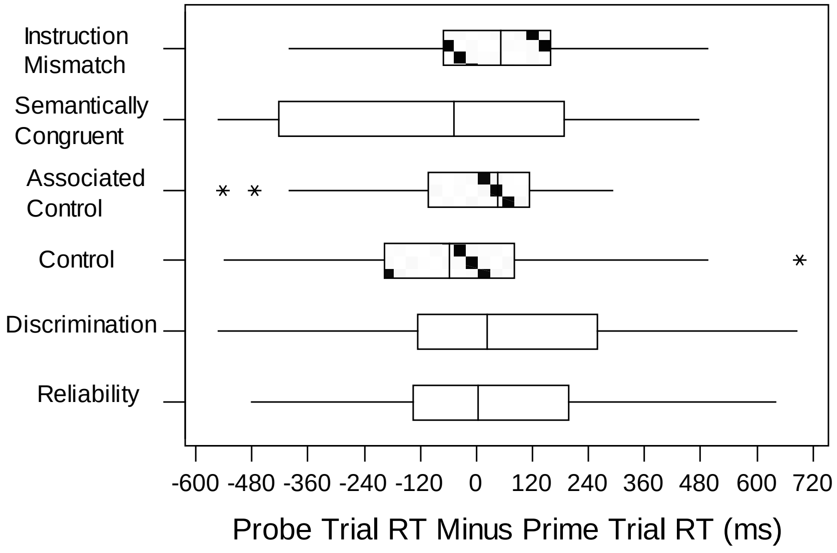


Figure 6: Box plots of the result in Table 1. Note the outliers represented as stars.

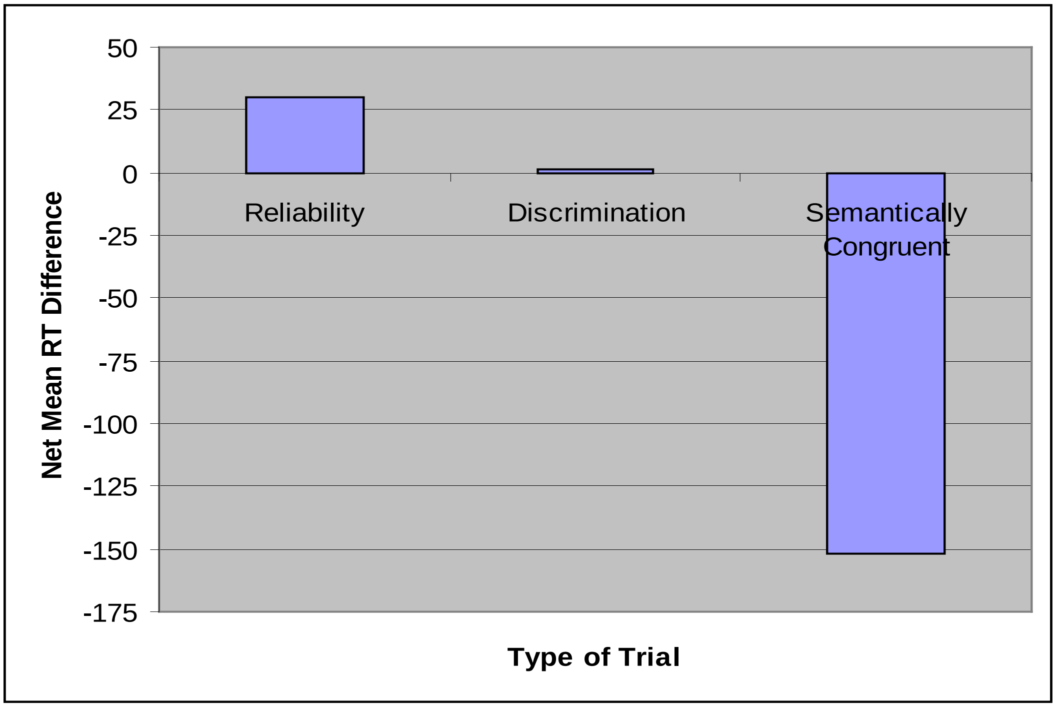


Figure 7: Graphical display of net mean RT differences found in Table 2. Error bars omitted.

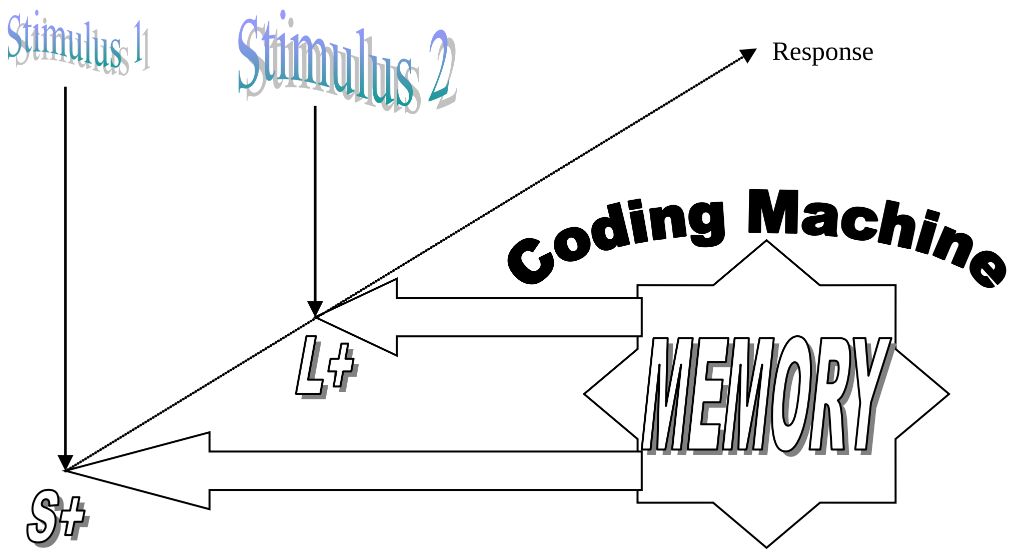


Figure 8: Revised model of semantic information processing using the memory retrieval machine. Here, the memory machine has been primed as a binary (large/small) processor. The processing is semantically congruent in this case.

Visuomotor Tracking

Ray Luo
rluo@cory.eecs.berkeley.edu
Department of Electrical Engineering and Computer Sciences
University of California
Berkeley, CA 94720

Abstract

Motor planning is modeled as a tracking problem in the case of rapid motor executions as in swatting a fly. Visual feedback and commanded motor movement are coupled in a linear dynamical systems graphical model of the task. Parameters learned by the EM Algorithm on an artificial data set reflect underlying system dynamics.

1 Introduction

How does motor planning and execution incorporate visual information when tracking a fast-moving object in space? One hypothesis involves the activity of an internal model consisting of a forward predicting and an inverse kinematic module [10]. Reviews by Jordan ([4] and [3]) give more details. A recent attempt to model selection of motor programs learns an HMM-based switch model for different types of objects [1]. A similar approach is advocated by Shademehr and Thoroughman [9].

Our approach here is simpler. We treat the motor planning problem as a tracking problem in which the nervous system estimates the positions of both the target and the hand position. The resulting graphical model is a coupled linear dynamical system that can be learned using the EM Algorithm. The approach is similar to what Wu and Huang calls co-inference tracking [11]. In their paper, they gave a sequential Monte Carlo algorithm that utilizes a *bottom up* EM step to track both color and shape distributions in a video sequence. We will begin wih a similar model adapted, in our case, to track a target in 2D with a 3D representation of hand position and velocity. We hope to extend the model by incorporating a switching motor control module within the arm location estimator.

2 Graphical model

Every node in the model is continuous, with equations that look like

$$\begin{aligned} H_s(t+1) &= BV_s(t+1) - H_s(t) + W_H(t), \\ V_s(t+1) &= AV_s(t) + W_V(t), \\ H_o(t) &= C_H H_s(t) + U_H(t), \\ V_o(t) &= C_V V_s(t) + U_V(t), \end{aligned}$$

where H_s is the control module associated with the hand, H_o is the estimated hand parameters (in particular, hand position), V_s and V_o are the estimated and observed target motion characteristics, A, B, and C are corresponding output matrices, and the Ws and Us are noise. H_o and V_o are observed. Note that the first equation accounts for sensory correction due to visual feedback.

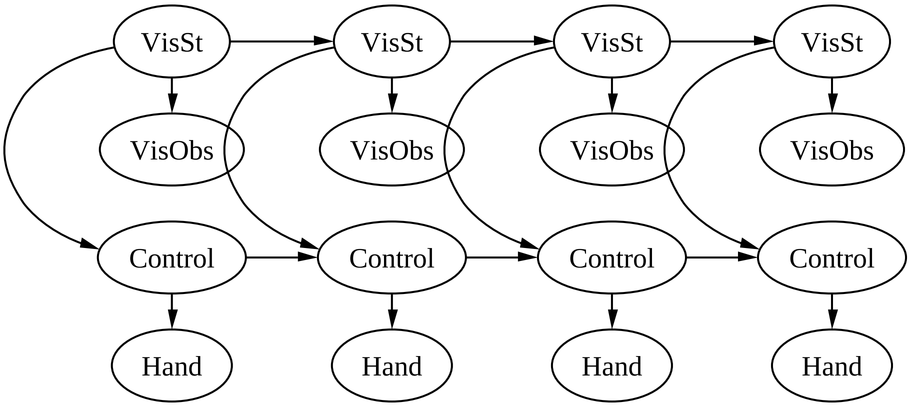
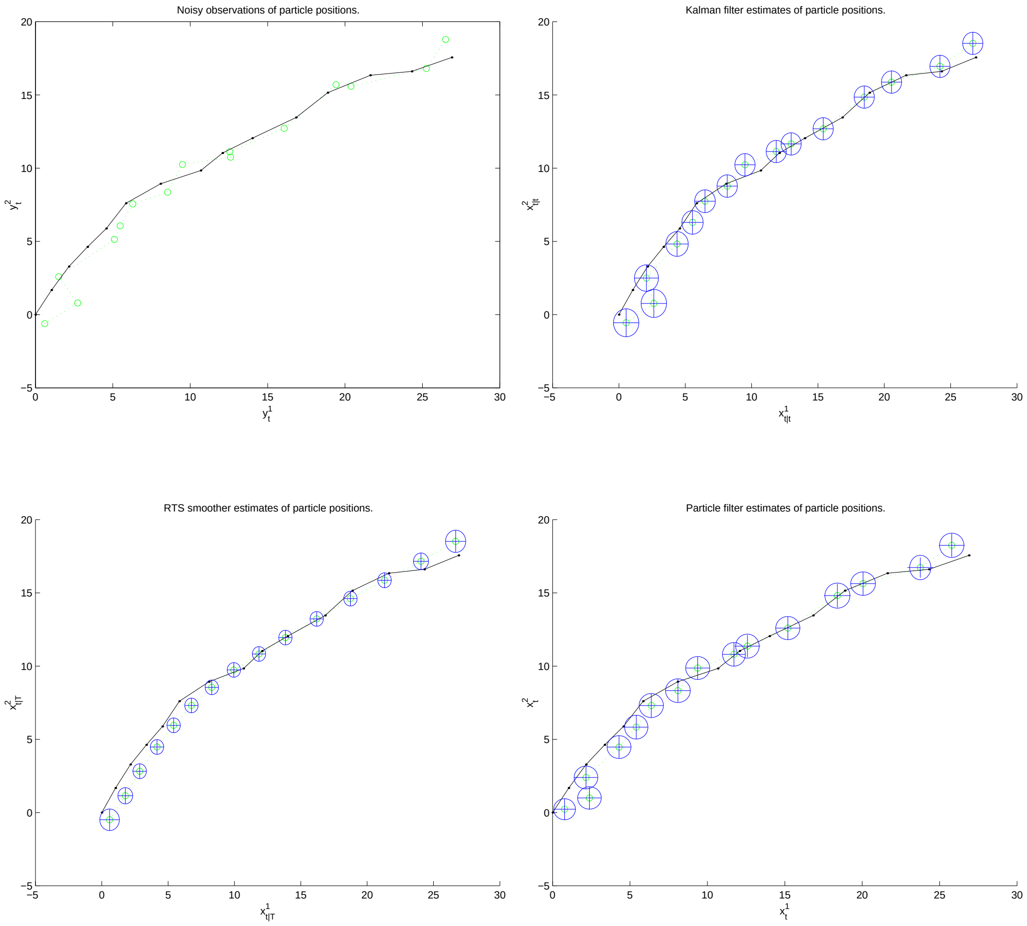


Figure 1: Graphical model of visuomotor tracking. Note that $VisSt$ and $Hand$ nodes are observed in our case.

From Figure 1, we see that visual information enters the system by correcting the control signal at the next time step (the alignment of the time slice is arbitrary). Given the controller, the observed motor characteristics are independent of all visual information.

3 Implementation and results

I first built a Kalman filter with RTS smoothing ([5]) for tracking positions and velocities, for the V_s and V_o cluster of the graphical model. I implemented a sequential MCMC ([8], [7]) algorithm known as the Particle filter [2], which can handle the case of non-Gaussian emission and transition probabilities. I tested the two algorithms on data sampled from a linear dynamical system with Gaussian emissions (see the file `track.m`; availability discussed in the appendix).



The upper left figure is the observed data, the upper right is the Kalman filtered trajectory with Gaussian confidence elipse, the lower left is the RTS smoothed trajectory, the lower right is the particle filtered trajectory with variance calculated across samples for each dimension (and zero covariance). Note that the Particle filter has nearly constant variance across time. The major disadvantage of the Particle filter is that it is not obvious how to learn the parameters of the sample propagation (transition probabilities) and likelihood weighting (emission probabilities). We could, for example, run the algorithm many times to learn the optimal parameters. Here, I set the parameters equal to the initial Kalman filtering parameters.

Next I implemented the model of Figure 1 using linear dynamical components. I generated some test data from reasonable dynamical considerations. For example, subjects usually move slowly initially, speeds up in the middle of the trajectory, and slow down again towards the end of the movement.

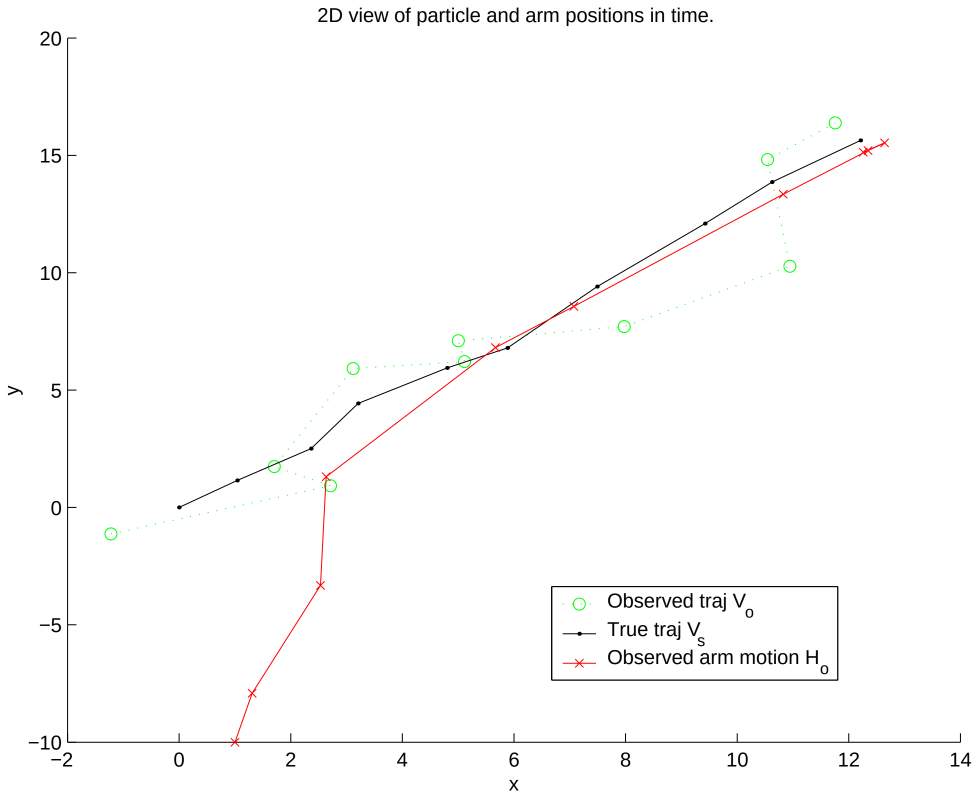


Figure 2: Artificial data used to train the graphical model, shown in 2D.

Figure 3 shows the V_o as green dots and H_o as red lines against the true target trajectory in black. Figure 3 shows the 3D representation. Our model assumes a 6D H_s representing the three spatial dimensions and their derivatives, a 4D V_s for visual tracking of 2D coordinates and their time derivatives, and 3D H_o and 2D V_o, observations of spatial locations. The parameters learned are given in the appendix. A log likelihood trace of EM is shown in Figure 3.

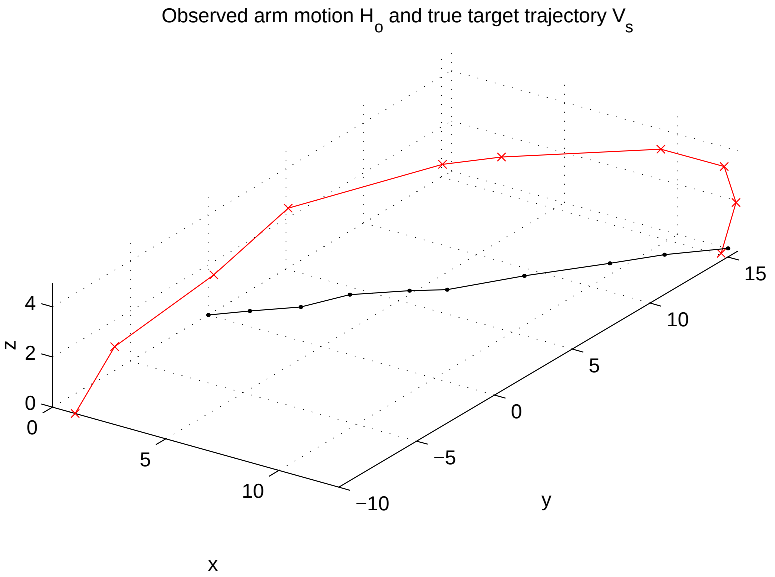


Figure 3: Trajectory of arm movement used to train the model.

4 Future work

It would be fruitful to conduct a linear systems analysis of the stability of the system. We have the visual signal as input, the motor signal as feedback, and the arm as the plant; output is the controller parameters.

The EM Algorithm takes a long time on this model. It would be useful to implement variational inference for this factorized model, as is done in [11]. More analysis of the results is necessary. In particular, we can look into the velocity profiles of the learned model. We investigate how spatial cueing effects come about by looking for consistencies among parameters learned

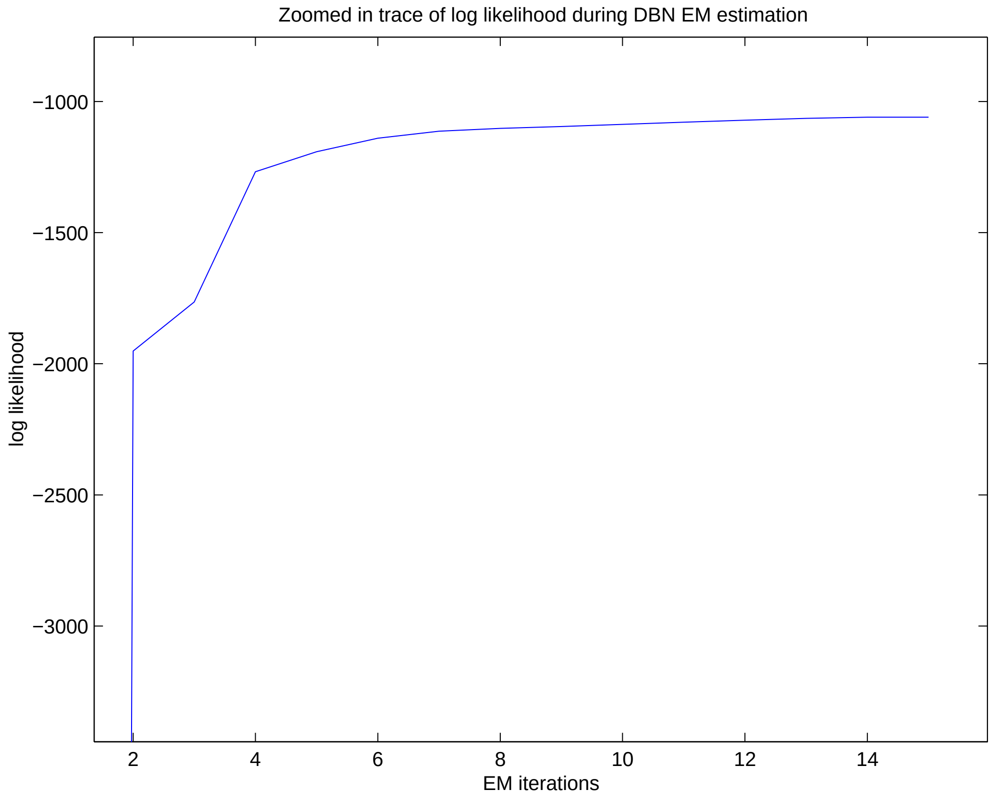


Figure 4: Trace of log likelihood during EM.

for different conditions in an experiment. We can examine the smoothness condition that is learned by the model.

The next step is to build a modular control system within the visuomotor tracking framework. This allows us to recognize and track different objects using a switch model. For example, swatting a fly is much more difficult than catching a baseball. We want to capture this difficulty in parameters of the model learned.

Appendix

Matlab code is found at `http://inst.eecs.berkeley.edu/~rluo/cs281`. See in particular the file `vismotor.m` to start things off. Some portions of the code needs the BNT toolkit written by Kevin Murphy.

The parameters learned by EM for the model in Figure 1 are:

```
node H_s0
m: 0.2125
   -0.0174
   2.0652
   0.2764
   -0.3802
   -0.8041
s: 0.0058 -0.0018 0.0000 0.0000 0.0000 -0.0000
   -0.0018 0.0025 0.0000 0.0000 -0.0000 -0.0000
   0.0000 0.0000 0.0370 -0.0445 0.0802 -0.0502
   0.0000 0.0000 -0.0445 0.1391 -0.2566 0.1239
   0.0000 -0.0000 0.0802 -0.2566 0.4787 -0.2293
   -0.0000 -0.0000 -0.0502 0.1239 -0.2293 0.1175
node V_s0
m: 2.0652
   0.2764
   -0.3802
   -0.8041
s: 0.0370 -0.0445 0.0802 -0.0502
   -0.0445 0.1391 -0.2566 0.1239
   0.0802 -0.2566 0.4787 -0.2293
   -0.0502 0.1239 -0.2293 0.1175
```

```
node H_o
m: 0.0000
   0.0000
   0.0000
s: 0.7829 0.2654 1.2374
   0.2654 1.8653 -0.0461
   1.2374 -0.0461 2.4727
node V_o
m: 0.0000
   0.0000
s: 0.3569 -0.3596
   -0.3596 1.0751
node H_s1
m: 0.0000
   0.0000
   0.0000
   0.0000
   0.0000
   0.0000
s: 0.3214 -0.0319 0.0612 0.1263 0.0506 0.1083
   -0.0319 0.0901 -0.0219 -0.1511 -0.0948 -0.0554
   0.0612 -0.0219 4.6232 8.6694 7.0972 6.5681
   0.1263 -0.1511 8.6694 17.0857 13.6294 12.7710
   0.0506 -0.0948 7.0972 13.6294 11.4500 10.2067
   0.1083 -0.0554 6.5681 12.7710 10.2067 9.7093
node V_s1
m: 0.0000
   0.0000
   0.0000
   0.0000
s: 0.0724 -0.0086 -0.0062 -0.0170
   -0.0086 0.0501 -0.0154 0.0176
   -0.0062 -0.0154 0.0824 -0.0183
   -0.0170 0.0176 -0.0183 0.0369
```

References

[1] M. Haruno, D. M. Wolpert, and M. Kawato. MOSAIC model for sensorimotor learning and control. *Neural Computation*, 13:2201–2220, 2001.

[2] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision*, pages 343–356, Cambridge, UK, 1996.

[3] M. I. Jordan. Computational aspects of motor control and motor learning. In H. Heuer and S. Keele, editors, *Handbook of Perception and Action: Motor Skills*. Academic Press, NY, 1996.

[4] M. I. Jordan and D. M. Wolpert. Computational motor control. In M. S. Gazzaniga, editor, *The Cognitive Neurosciences*. MIT Press, MA, 1995.

[5] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the American Society of Mechanical Engineers - Journal of Basic Engineering*, 82(D):35–45, 1960.

[6] A. Leon-Garcia. *Probability and Random Processes for Electrical Engineering*. Addison-Wesley, Reading, MA, 2nd edition, 1994.

[7] J. S. Liu and R. Chen. Sequential Monte Carlo methods for dynamical systems. *Journal of the American Statistical Association*, 93:1032–1044, 1998.

[8] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092, June 1953.

[9] K. A. Thoroughman and R. Shadmehr. Learning of action through adaptive combination of motor primitives. *Nature*, 407:742–747, October 2000.

[10] D. M. Wolpert, Z. Ghahramani, and M. I. Jordan. An internal model for sensorimotor integration. *Science*, 269:1880–1882, September 1995.

[11] Y. Wu and T. S. Huang. A co-inference approach to robust visual tracking. In *International Conference on Computer Vision*, Vancouver, Canada, 2001.