Categories and Concepts

Computational models (part 1)

Brenden Lake

PSYCH-GA 2207

Why build a computational model?

"Verbally expressed statements are sometimes flawed by internal inconsistencies, logical contradictions, theoretical weaknesses and gaps. A running computational model, on the other hand, can be considered as a sufficiency proof of the internal coherence and completeness of the ideas it is based upon." (Fum, Del Missier, & Stocco, 2007)

Some famous psychological theories…

- Attention is like a spotlight
- A child learning about the world is like a scientist theorizing about science
- Language influences thought
- Working memory is having 7 +/- 2 slots to store information
- Categorization happens by comparing novel instances to past exemplars
- Categories influence perception

Each of these theories benefits from formalization with a computational model to…

- **Make predictions explicit**
- Implications often **defy expectations**
- **Aid communication** between scientists
- Support **cumulative progress**

My view echos Richard Feynman's

"What I cannot create, I do not understand."

Agenda for the next 2 classes

Two influential computational models

- ALCOVE
  - “An exemplar-based connectionist model of category learning”, Krushcke
- Rational Model
  - “An exemplar-based connectionist model of category learning”, Anderson

And some comments on category learning in modern AI

Psychological Review
1992, Vol. 99, No. 1, 22-44

0033-295X/92/$3.00

ALCOVE: An Exemplar-Based Connectionist Model of Category Learning

John K. Kruschke
Indiana University Bloomington

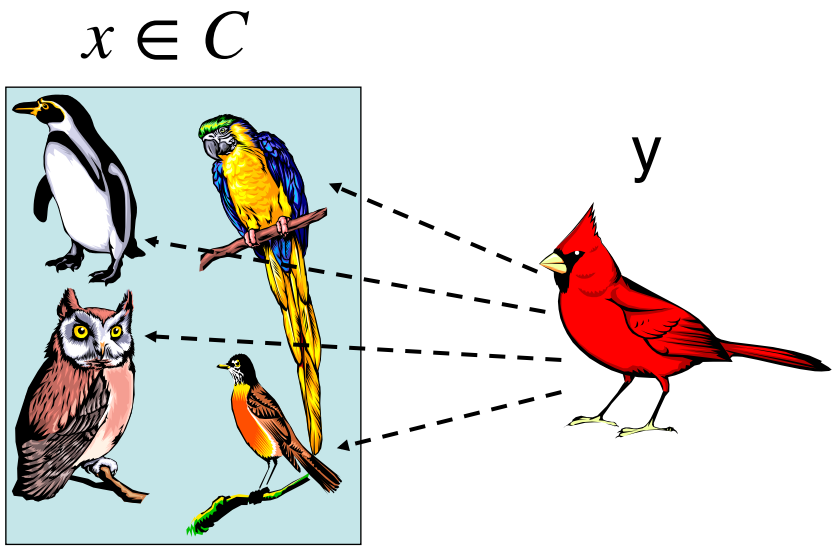
ALCOVE (attention learning covering map) is a connectionist model of category learning that incorporates an exemplar-based representation (Medin & Schaffer, 1978; Nosofsky, 1986) with error-driven learning (Gluck & Bower, 1988; Rumelhart, Hinton, & Williams, 1986). ALCOVE selectively attends to relevant stimulus dimensions, is sensitive to correlated dimensions, can account for a form of base-rate neglect, does not suffer catastrophic forgetting, and can exhibit 3-stage (U-shaped) learning of high-frequency exceptions to rules, whereas such effects are not easily accounted for by models using other combinations of representation and learning method.

This article describes a connectionist model of category learning called ALCOVE (attention learning covering map). Any model of category learning must address the two issues of what representation underlies category knowledge and how that representation is used in learning. ALCOVE combines the exemplar-based representational assumptions of Nosofsky's (1986) generalized context model (GCM) with the error-driven learning assumptions of Gluck and Bower's (1988a, 1988b) network models. ALCOVE extends the GCM by adding a learning mechanism and extends the network models of Gluck and Bower by allowing continuous dimensions and including explicit dimensional attention learning. ALCOVE can be construed as a combination of exemplar models (e.g., Medin & Schaffer, 1978; Nosofsky, 1986) with network models (Gluck & Bower, 1988a, 1988b), as suggested by Estes (1988; Estes, Campbell, Hatsopoulos, & Hurwitz, 1989; Hurwitz, 1990). Dimensional attention learning allows ALCOVE to capture human performance where other network models fail (Gluck & Bower, 1988a), and error-driven learning in ALCOVE gener-

dient descent on error, it is unlike standard back propagation in its architecture, its behavior, and its goals. Unlike the standard back-propagation network, which was motivated by generalizing neuronlike perceptrons, the architecture of ALCOVE was motivated by a molar-level psychological theory, Nosofsky's (1986) GCM. The psychologically constrained architecture results in behavior that captures the detailed course of human category learning in many situations where standard back propagation fares less well. Unlike many applications of standard back propagation, the goal of ALCOVE is not to discover new (hidden-layer) representations after lengthy training but rather to model the course of learning itself by determining which dimensions of the given representation are most relevant to the task and how strongly to associate exemplars with categories.

The purposes of this article are to introduce the ALCOVE model, demonstrate its application across a variety of category learning tasks, and compare it with other models to highlight its mechanisms. The organization of the article is as follows: First, the ALCOVE model is described in detail; then, its ability to

Like the Context Model, ALCOVE is an exemplar model



Review of the "Context Model"

Similarity between two items *x* and *y* (Medin & Schaffer, 1978)

Category A

	D_1	D_2	D_3	D_4
x	1	1	1	1
	1	1	1	0
	0	0	0	1

sim(y, x)

0.09

Category B

D_1	D_2	D_3	D_4
0	0	0	0
0	0	1	1
1	1	0	0

Transfer item

y	0	1	0	1

$$\textbf{sim}(y, x) = \prod_{D_i} m^{1[x_i \neq y_i]} = m \cdot 1 \cdot m \cdot 1 = 0.09$$

"mismatch" free parameter such that $m = 0.3$

Review: Context model

"classification is based on similarity to all exemplars in a class"
(Medin & Schaffer, 1978)

Item similarity

$$\mathbf{sim}(y, x) = \prod_{D_i} m^{1[x_i \neq y_i]}$$

"mismatch" free parameter *m*

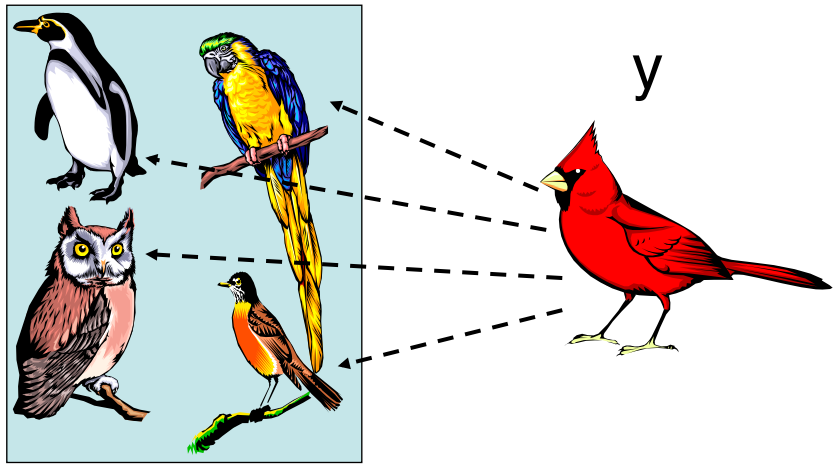
Category similarity

$$\mathbf{sim}(y, C) = \sum_{x \in C} \mathbf{sim}(y, x)$$

Probability of classification

$$P(y \in C) = \frac{\mathbf{sim}(y, C)}{\sum_{C'} \mathbf{sim}(y, C')}$$

$x \in C$



Birds
You've Seen

Bird?

ALCOVE's similarity between two items *x* and *y*

Context model's item similarity

$$\mathbf{sim}(y, x) = \prod_{D_i} m_i^{1[x_i \neq y_i]}$$

(assuming we have a different mismatch parameter m_i for each dimension)

Recall this identity:

$$\prod_{i=1}^{N} z_i = \prod_{i=1}^{N} e^{\log z_i} = e^{\sum_{i=1}^{N} \log z_i}$$

$$\alpha_i = \log m_i$$

(equivalent for binary dims, with this identity)

ALCOVE's item similarity (**also works for continuous features**)

$$\mathbf{sim}(y, x) = e^{\sum_{D_i} \log m_i \; 1[x_i \neq y_i]} = e^{\sum_{D_i} \alpha_i |x_i - y_i|}$$

(parameters *c*, *q*, and *r* not shown)

ALCOVE's similarity between two items *x* and *y*

Category A $\quad$ **sim**(y, x) $\quad$ **Category B**

	D_1	D_2	D_3	D_4
	1	1	1	1
	1	1	1	0
	0	0	0	1

0.09

D_1	D_2	D_3	D_4
0	0	0	0
0	0	1	1
1	1	0	0

x

Transfer item

y

0	1	0	1

$$\textbf{sim}(y, x) = \prod_{D_i} m^{1[x_i \neq y_i]} = m \cdot 1 \cdot m \cdot 1 = 0.09$$

"mismatch" free parameter such that $m = 0.3$

$$\textbf{sim}(y, x) = e^{\sum_{D_i} \log m_i \, |x_i - y_i|} = e^{-1.20 + 0 - 1.20 + 0} = 0.09$$

ALCOVE

Item similarity
(discrete or continuous)

$$\mathbf{sim}(y, x) = e^{\sum_{D_i} \alpha_i |x_i - y_i|}$$

key parameter: attention weights α_i

Category similarity

$$\mathbf{sim}(y, C) = \sum_{x \in X} w_{cx}\ \mathbf{sim}(y, x)$$

X is the set of all exemplars across all categories

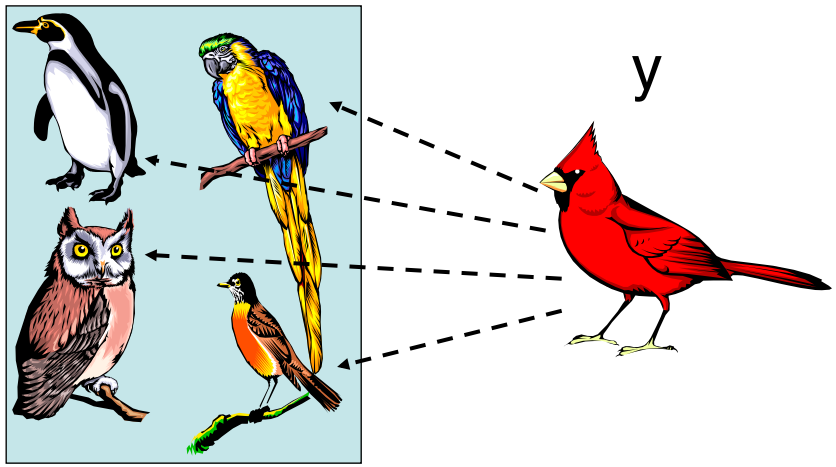
key parameter: association weights between exemplar *x* and category *c* w_{cx}

Probability of classification

$$P(y \in C) = \frac{e^{\phi \mathbf{sim}(y,C)}}{\sum_{C'} e^{\phi \mathbf{sim}(y,C')}}$$

ϕ response "temperature"

$x \in X$



ALCOVE

Context model

Item similarity

$$\textbf{sim}(y, x) = e^{\sum_{D_i} \alpha_i |x_i - y_i|}$$

key parameter: attention weights α_i

$$\textbf{sim}(y, x) = \prod_{D_i} m^{1[x_i \neq y_i]}$$

"mismatch" free parameter *m*

Category similarity

$$\textbf{sim}(y, C) = \sum_{x \in X} w_{cx}\ \textbf{sim}(y, x)$$

key parameter: association weights w_{cx} between exemplar *x* and category *c*

$$\textbf{sim}(y, C) = \sum_{x \in C} \textbf{sim}(y, x)$$

(only considers exemplars *x* assigned to C)

Probability of classification

$$P(y \in C) = \frac{e^{\phi \textbf{sim}(y,C)}}{\sum_{C'} e^{\phi \textbf{sim}(y,C')}}$$

ϕ response "temperature"

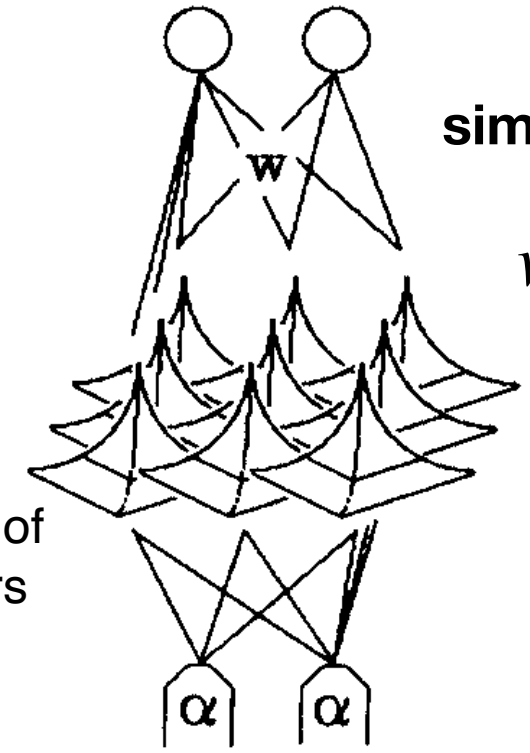
$$P(y \in C) = \frac{\textbf{sim}(y, C)}{\sum_{C'} \textbf{sim}(y, C')}$$

ALCOVE

Response rule

$$P(y \in A) = \frac{e^{\phi \mathbf{sim}(y,A)}}{e^{\phi \mathbf{sim}(y,A)} + e^{\phi \mathbf{sim}(y,B)}}$$

category A category B



$$\mathbf{sim}(y, C) = \sum_{x \in X} w_{cx}\ \mathbf{sim}(y, x)$$

w_{cx} association weights between exemplar *x* and category *c*

$$\mathbf{sim}(y, x) = e^{\sum_{D_i} \alpha_i |x_i - y_i|}$$

x denotes one of the exemplars

α_i attention weights

y current stimulus

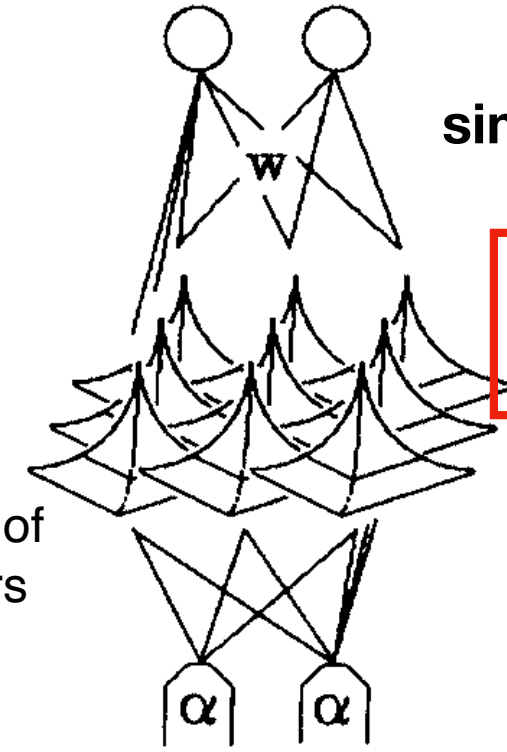
How does ALCOVE learn?

Learning is incremental fitting of the attention weights and association weights

Response rule

$$P(y \in A) = \frac{e^{\phi \mathbf{sim}(y,A)}}{e^{\phi \mathbf{sim}(y,A)} + e^{\phi \mathbf{sim}(y,B)}}$$

category A category B



$$\mathbf{sim}(y, C) = \sum_{x \in X} w_{cx}\, \mathbf{sim}(y, x)$$

w_{cx} association weights between exemplar *x* and category *c*

$$\mathbf{sim}(y, x) = e^{\sum_{D_i} \alpha_i |x_i - y_i|}$$

x denotes one of the exemplars

α_i attention weights

y current stimulus

Learning by optimizing an objective function

Let's use log-likelihood as our **objective function**:

$$F(w, \alpha) = \sum_{x \in A} \log P_{w,\alpha}(x \in A) + \sum_{x \in B} \log(P_{w,\alpha}(x \in B))$$

("maximum likelihood" is also the objective function for linear regression, logistic regression, and many other stats/machine learning algorithms…)

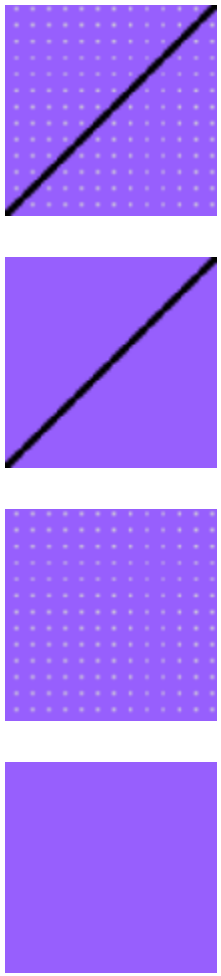
Then the optimal parameters are given by:

$$\textbf{argmax}_{w,\alpha} \; F(w, \alpha)$$

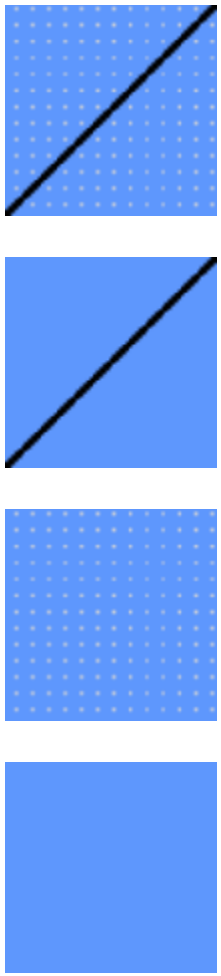
(note that this is a different objective function than used in original ALCOVE paper)

ALCOVE applied to SHJ Type I category

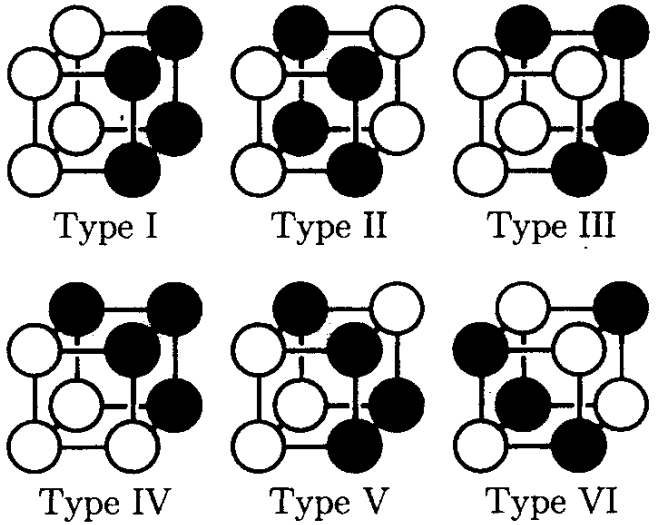
category A

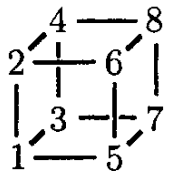


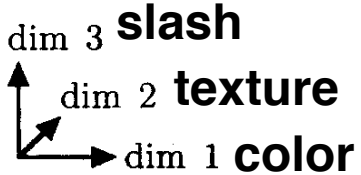
category B



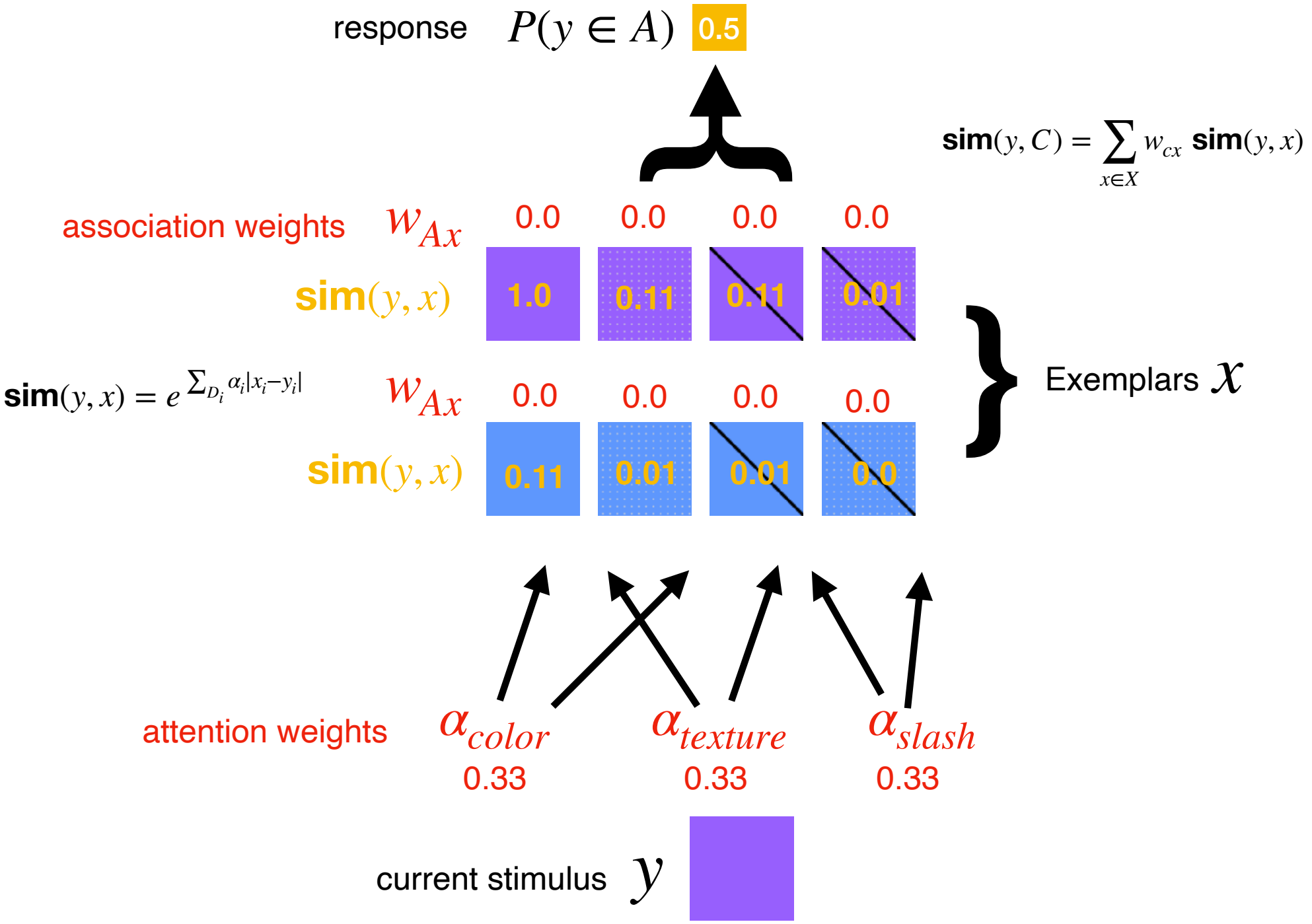
Shepard, Hovland, & Jenkins (1961)







Network BEFORE training

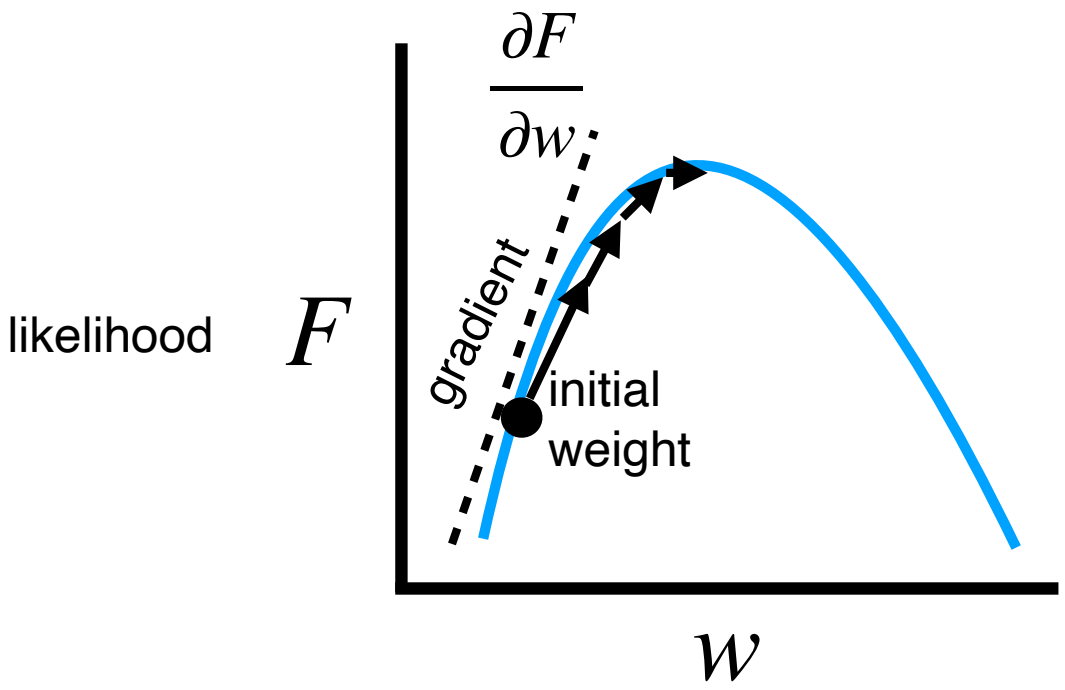


Maximizing likelihood via gradient descent

(the workhorse of modern machine learning, and often cognitive modeling too)

objective function to maximize

$$F(w, \alpha) = \sum_{x \in A} \log P_{w,\alpha}(x \in A) + \sum_{x \in B} \log(P_{w,\alpha}(x \in B))$$



Computing the gradient tells us which direction to go for steepest ascent:

$$w \leftarrow w + \gamma \frac{\partial F}{\partial w}$$

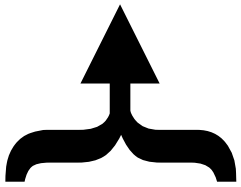
γ : learning rate (small constant)

ALCOVE after one gradient step

response $P(y \in A)$ 0.503

Response is a little better, since y is in A!

$$\textbf{sim}(y, C) = \sum_{x \in X} w_{cx}\ \textbf{sim}(y, x)$$



association weights w_{Ax} 0.015 0.002 0.002 0.0

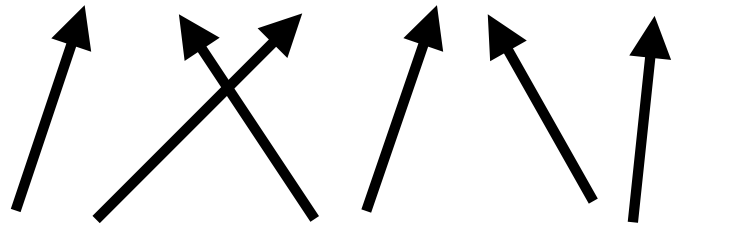
$\textbf{sim}(y, x)$ 1.0 0.11 0.11 0.01

$$\textbf{sim}(y, x) = e^{\sum_{D_i} \alpha_i |x_i - y_i|}$$

w_{Ax} 0.002 0.0 0.0 0.0

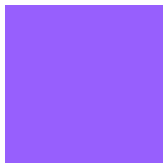
$\textbf{sim}(y, x)$ 0.11 0.01 0.01 0.0

} Exemplars x

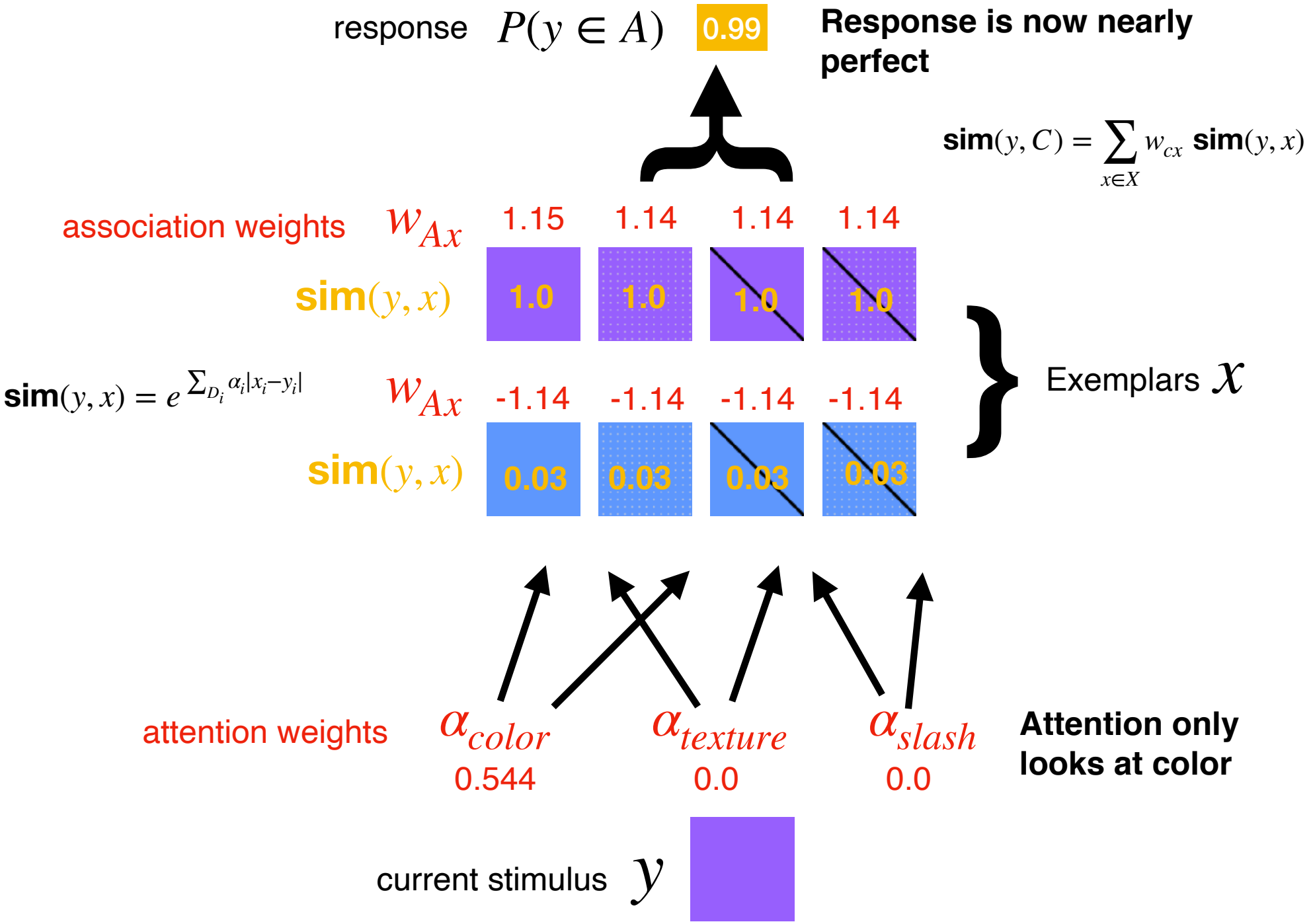


attention weights α_{color} 0.33 $\alpha_{texture}$ 0.33 α_{slash} 0.33

current stimulus y

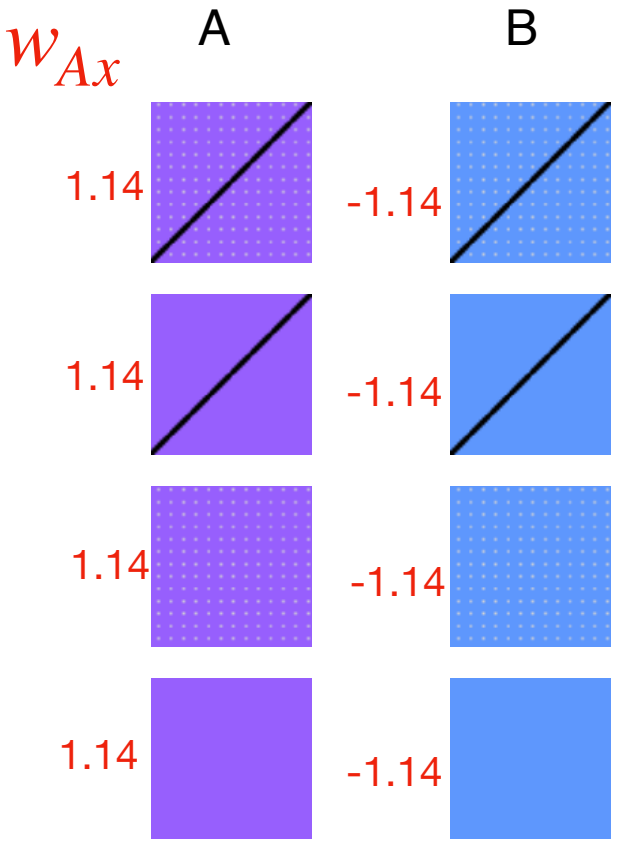


ALCOVE after many more gradient steps…



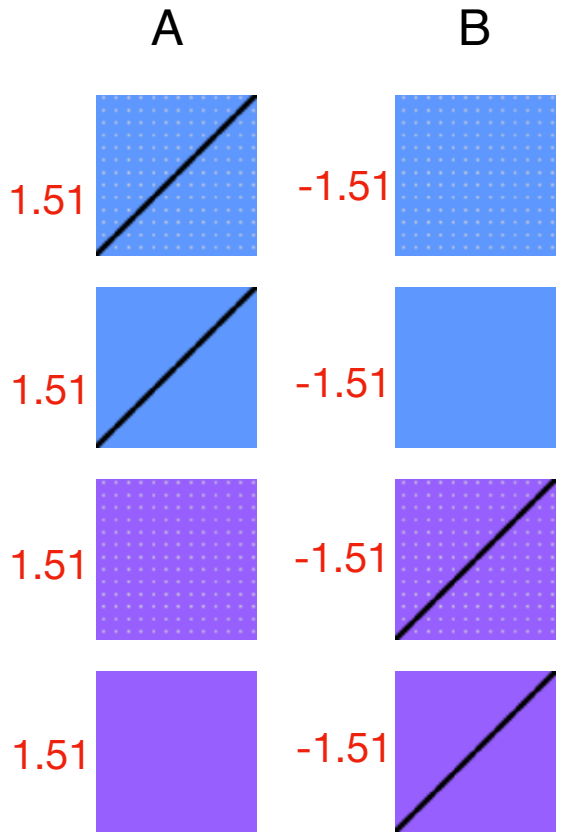
Solutions found for SHJ problems

Type I

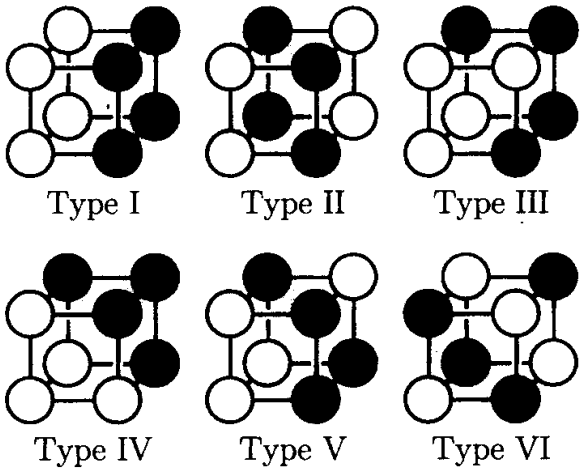


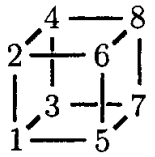
α_{color} 0.544 $\alpha_{texture}$ 0.0 α_{slash} 0.0

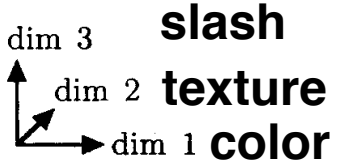
Type II



α_{color} 0.61 $\alpha_{texture}$ 0.0 α_{slash} 0.61

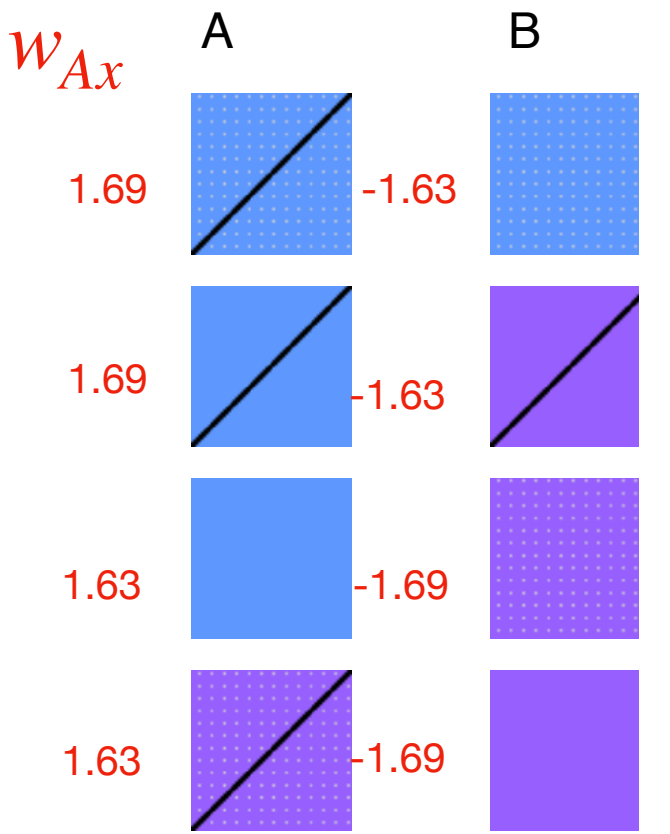






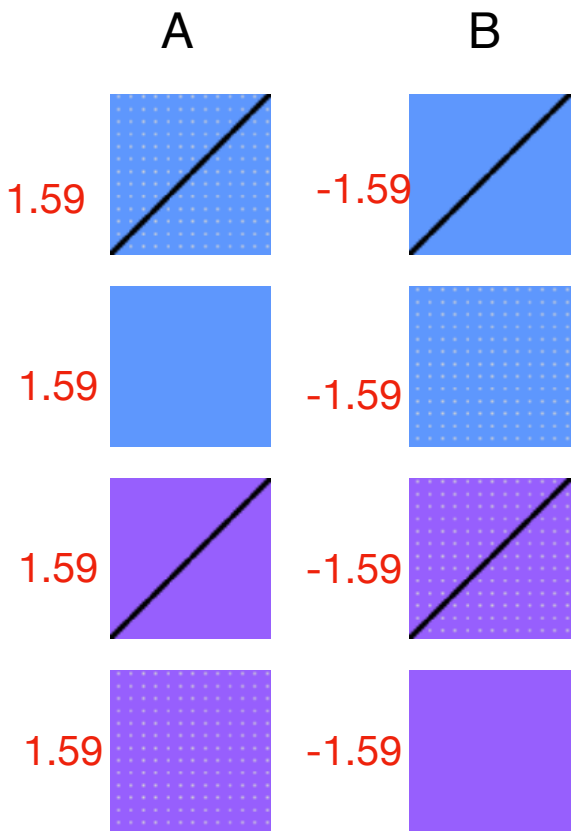
Solutions found for SHJ problems

Type III

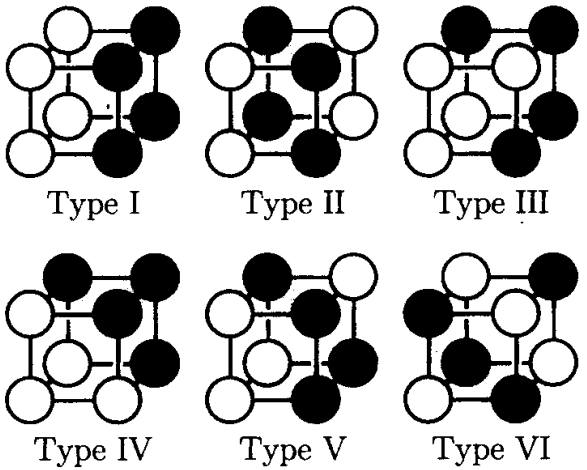


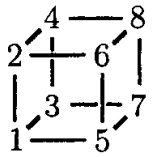
α_{color} 0.399 $\alpha_{texture}$ 0.399 α_{slash} 0.385

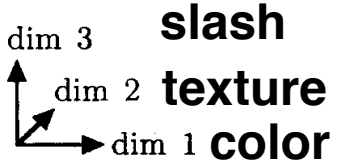
Type VI



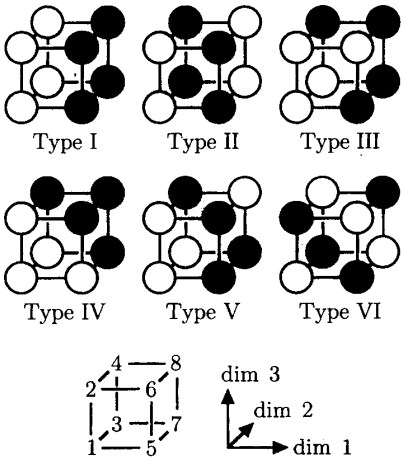
α_{color} 0.62 $\alpha_{texture}$ 0.62 α_{slash} 0.62

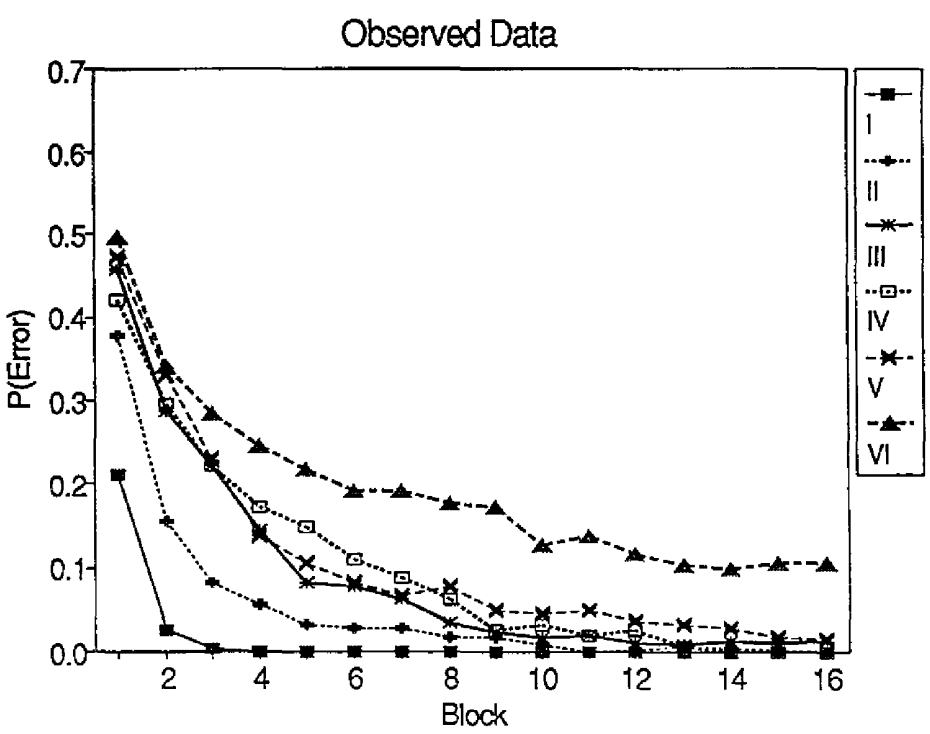


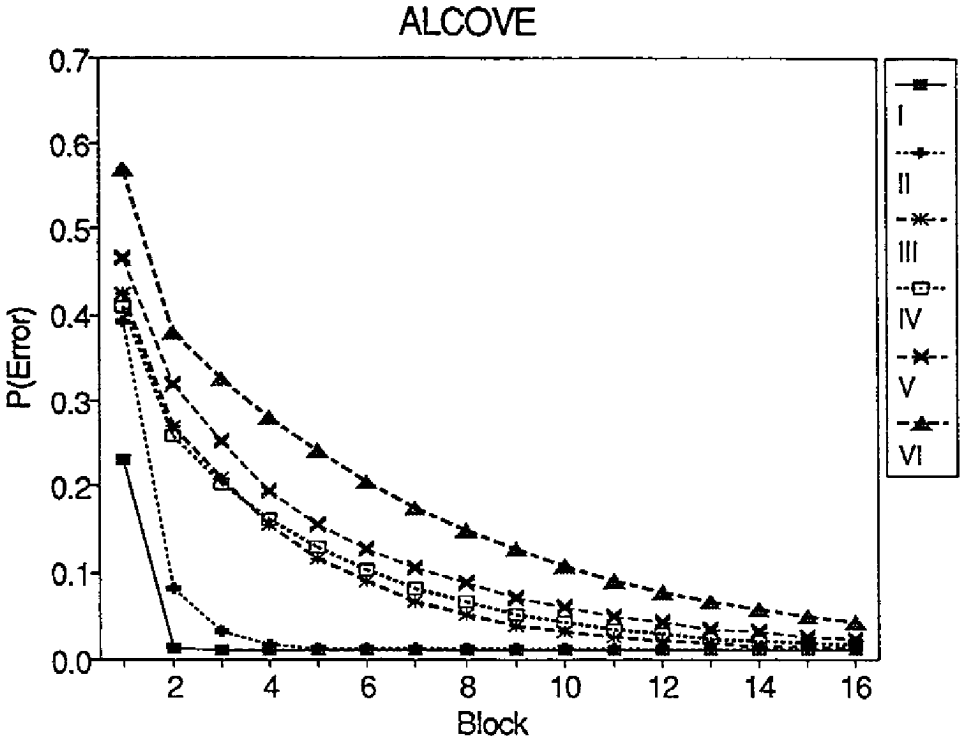




Comparing learning curves for people and ALCOVE



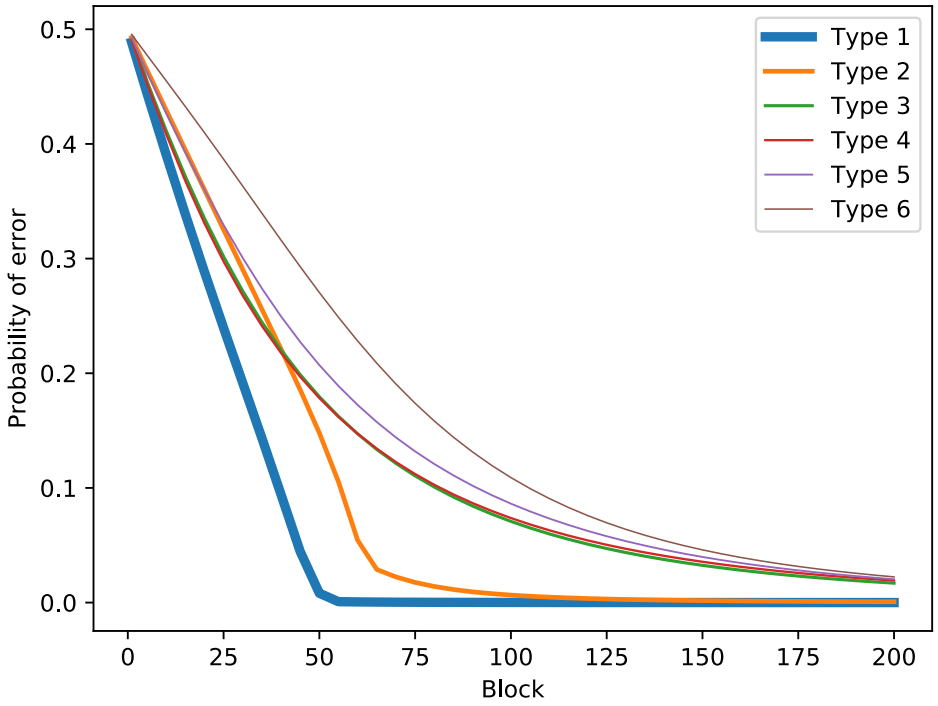




(from Nofosky et al. (1994) SHJ replication in *Memory & Cognition*)

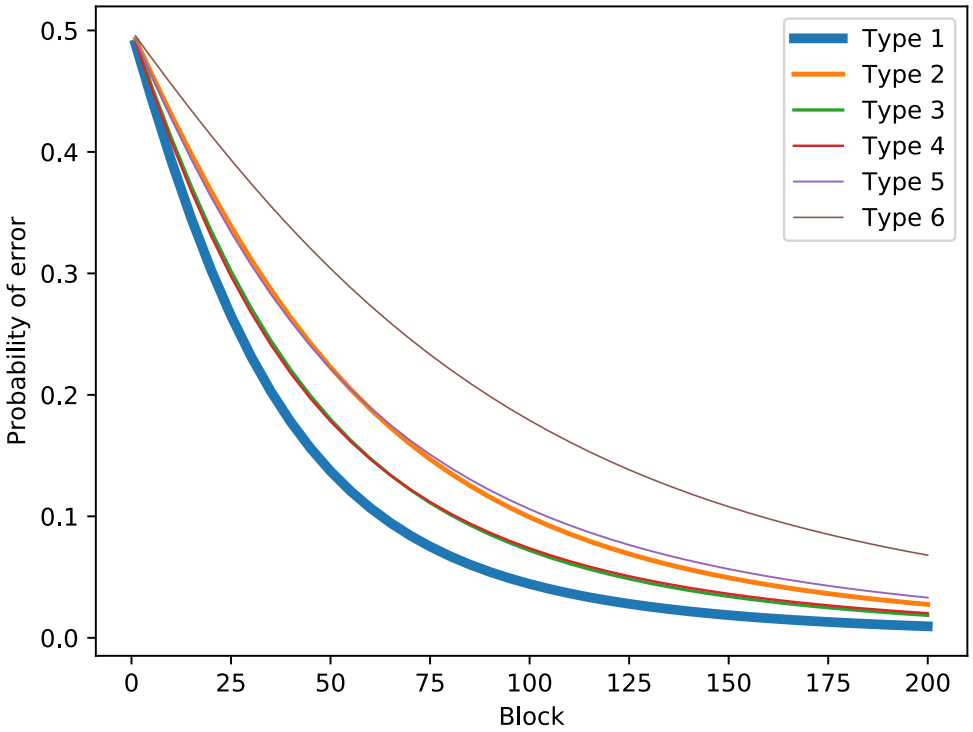
Attentional learning is critical

ALCOVE with attention weights



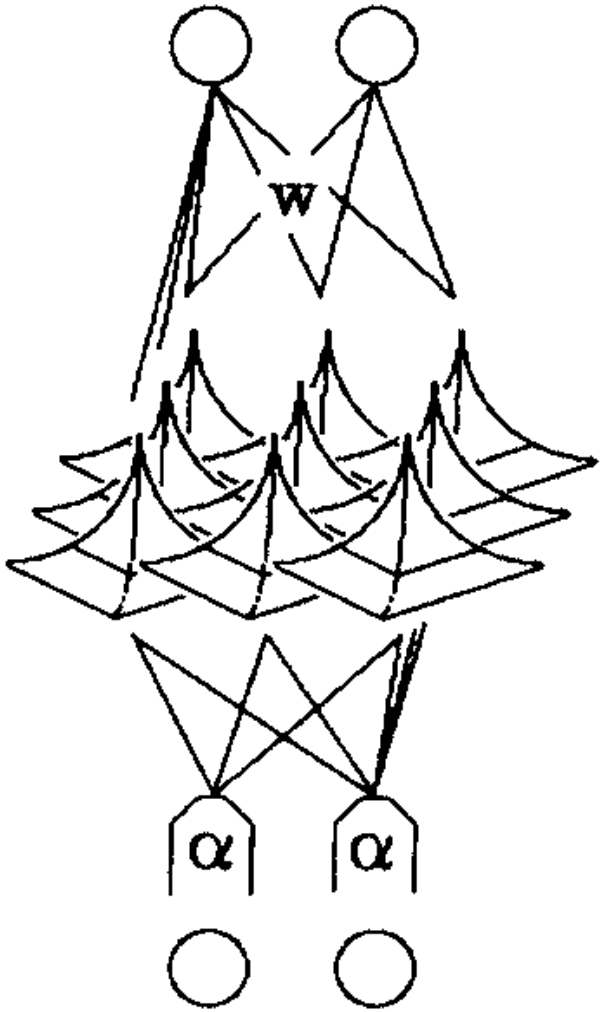
ALCOVE without attention weights

(gets Type II order wrong)

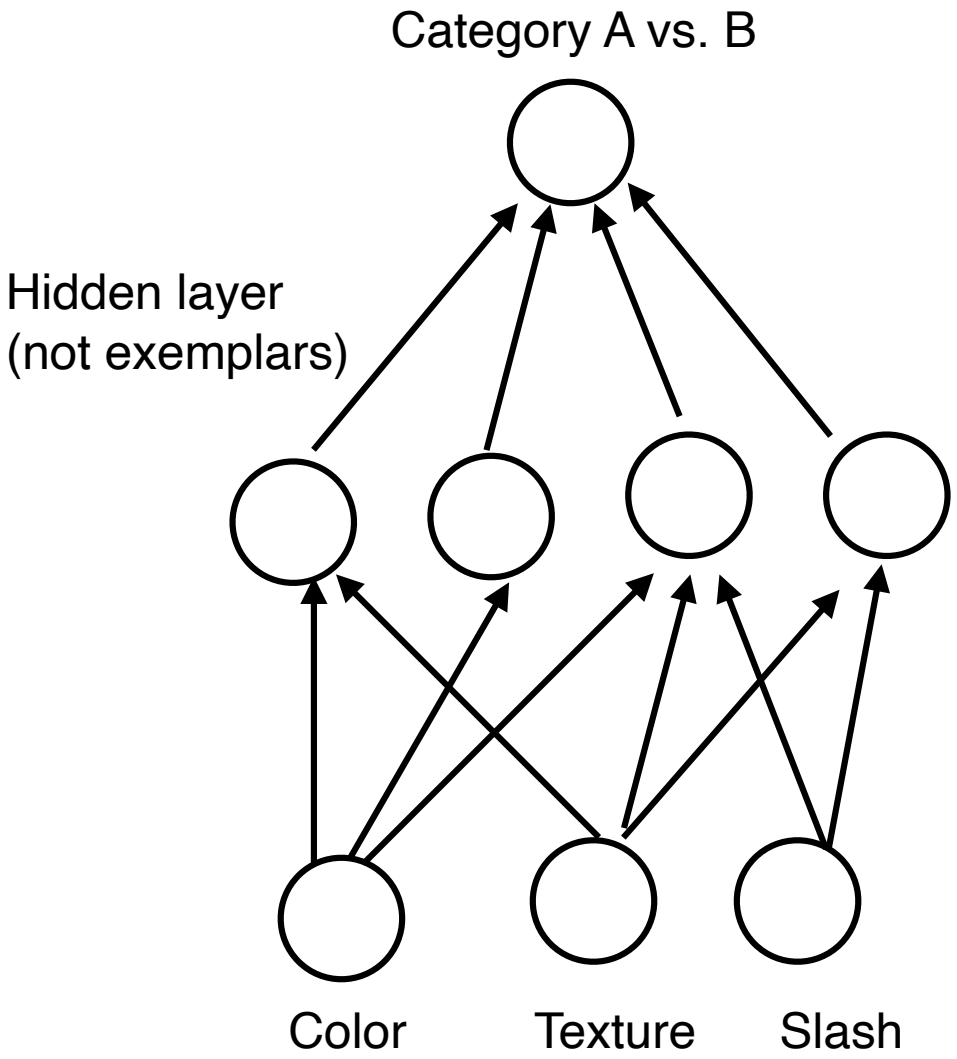


What about a standard neural network?

ALCOVE

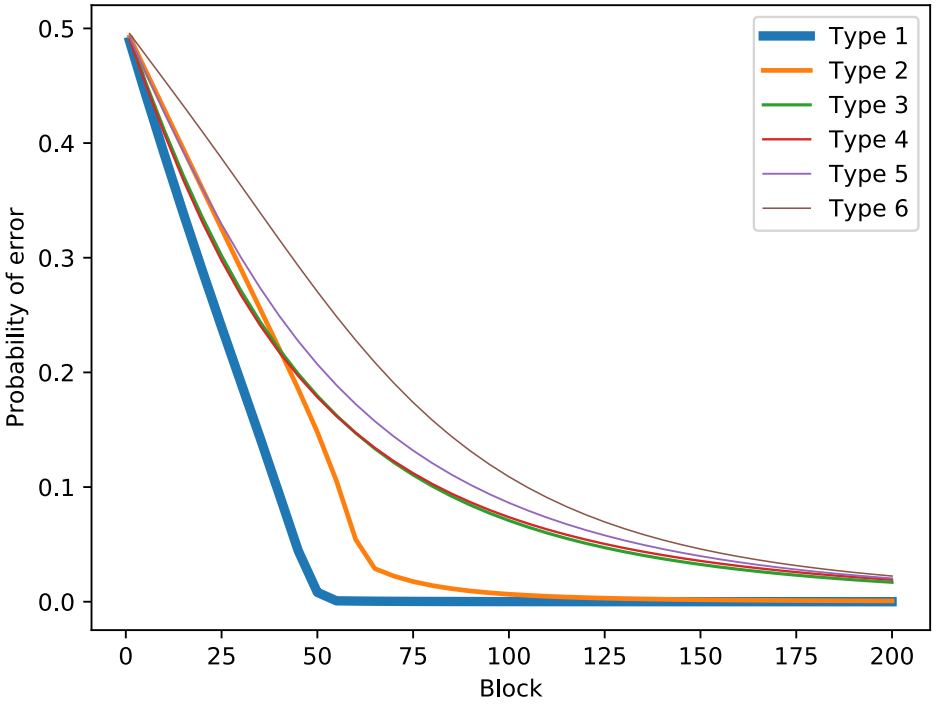


standard multi-layer network



What about a standard neural network?

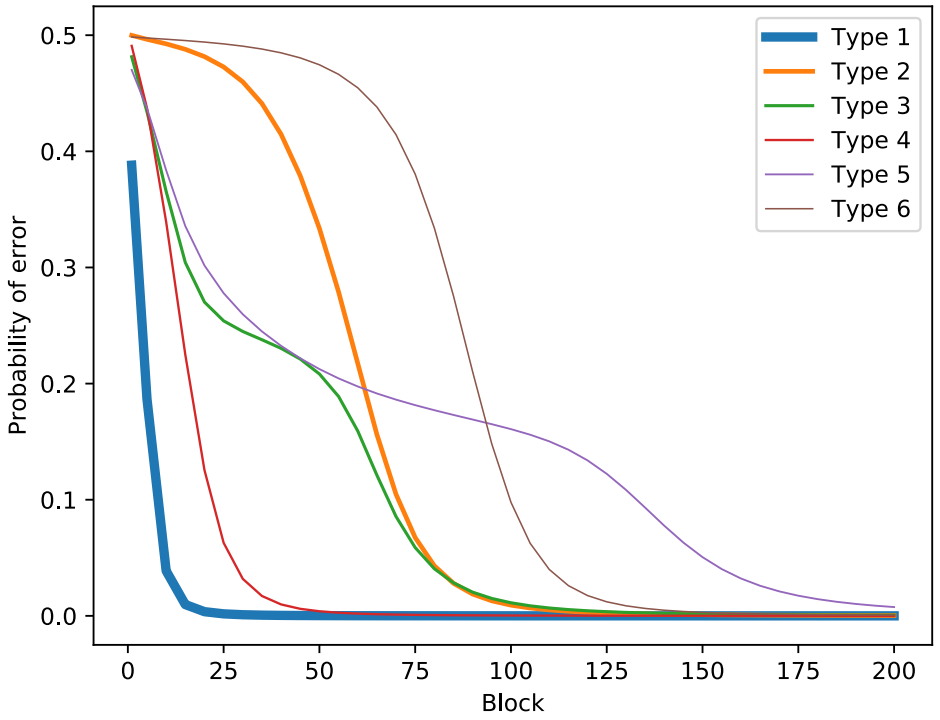
ALCOVE with attention weights



Standard multi-layer neural network

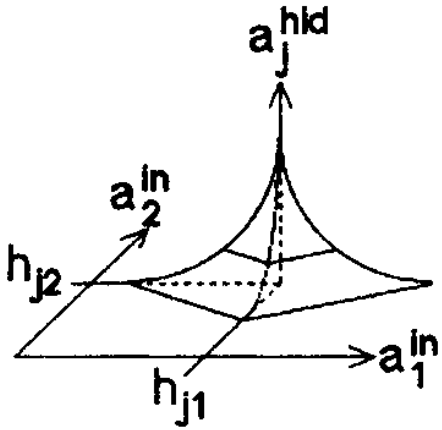
(gets Type II order wrong)

(gets Type IV order very wrong)

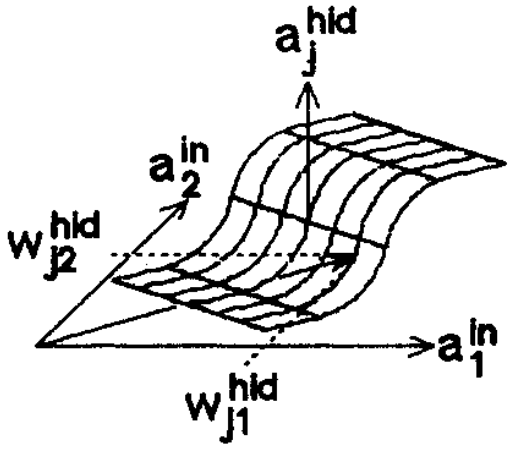


Standard nets prefer linearly separable categories, while ALCOVE prefers strong dimensional alignment

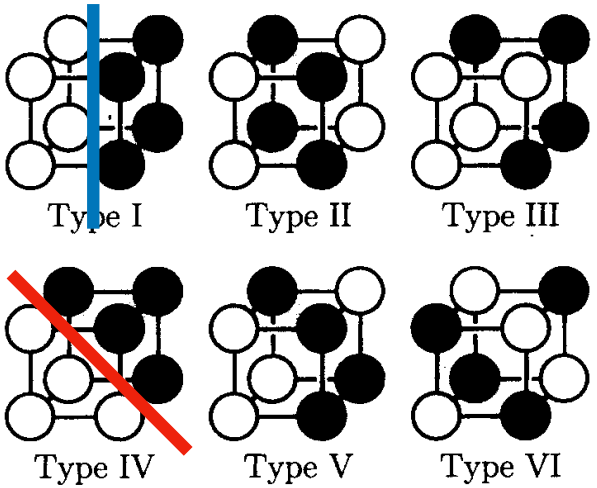
activation profile for ALCOVE hidden unit

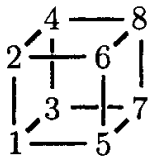


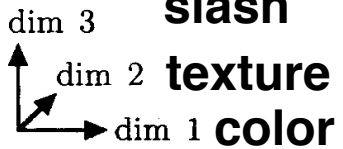
activation profile for standard neural net hidden unit



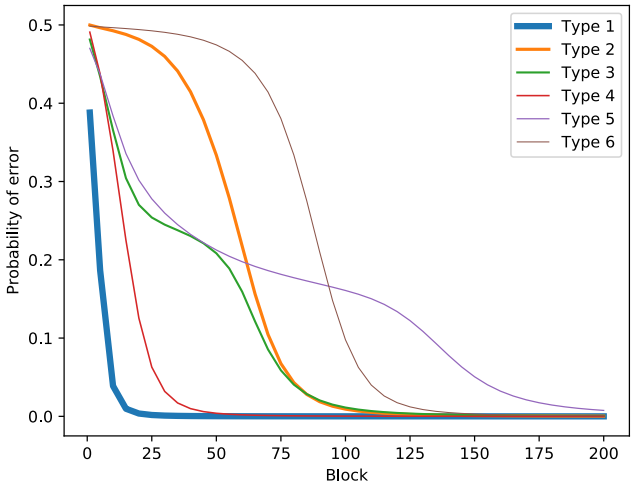
Standard multi-layer neural network







Only Type 1 and Type IV are linearly separable, and thus they are learned first with standard net



As predicted by ALCOVE, people don't necessarily favor linearly separable categories

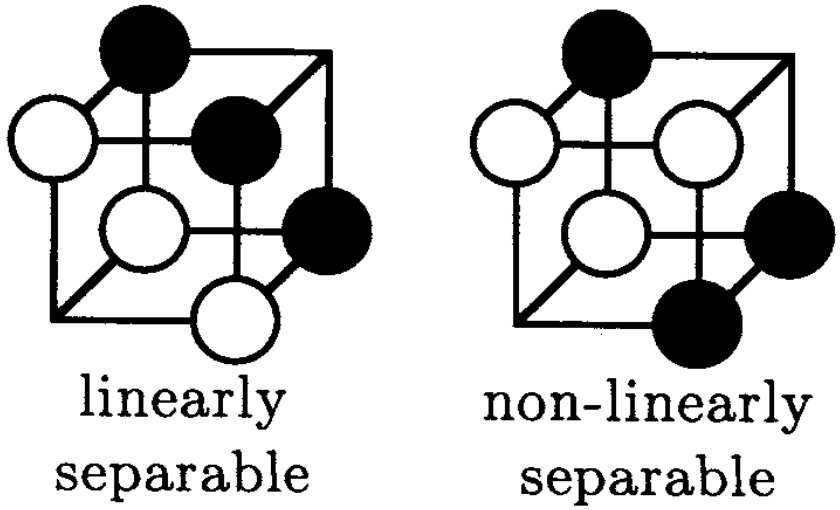


Figure 10. Category structures used by Medin and Schwanenflugel (1981, Experiment 4). (The linearly separable structure is a subset of Type IV in the Shepard, Hovland, and Jenkins, 1961, studies [cf. Figure 4], whereas the nonlinearly separable structure is the corresponding subset from Type III.)

linearly separable condition: people had 39.5 errors on average

non-linearly separable condition: people had 38.0 errors on average

(criterion was two error-free passes through stimuli)

Interlude: Bayes' rule for updating beliefs in light of data

Data (D): John is coughing

Hypotheses:

$h_1 =$ John has a cold

$h_2 =$ John has emphysema

$h_3 =$ John has a stomach flu

"Bayes' rule"

posterior: $P(h_i|D)$ — likelihood: $P(D|h_i)$ — prior: $P(h_i)$

$$P(h_i|D) = \frac{P(D|h_i)P(h_i)}{\sum_j P(D|h_j)P(h_j)}$$

Which hypotheses should we believe, and with what certainty?

We want to calculate the posterior probabilities: $P(h_1|D)$, $P(h_2|D)$, and $P(h_3|D)$

This example is from Josh Tenenbaum.

Bayesian inference

posterior ← $P(h_i|D)$ = likelihood ← $P(D|h_i)$ prior ← $P(h_i)$ / $\sum_j P(D|h_j)P(h_j)$

$$P(h_i|D) = \frac{P(D|h_i)P(h_i)}{\sum_j P(D|h_j)P(h_j)}$$

Data (D): John is coughing

Hypotheses:

h_1 = John has a cold — $P(h_1) = .75$ $P(D|h_1) = 1$

h_2 = John has emphysema — $P(h_2) = .05$ $P(D|h_2) = 1$

h_3 = John has a stomach flu — $P(h_3) = .2$ $P(D|h_3) = .2$

Prior favors h_1 and h_3, over h_2

Likelihood favors h_1 and h_2, over h_3

Posterior favors h_1, over h_2 and h_3

$$P(h_1|D) = .89 = \frac{.75(1)}{.75(1) + .05(1) + .2(.2)}$$

$$P(h_2|D) = .06$$

$$P(h_3|D) = .05$$

Base rate neglect and ALCOVE

(Experiment from Gluck and Bower 1988)

There is a common disease and rare disease with certain base rates:

$$P(C) = 0.75 \qquad P(R) = 0.25$$

The diseases lead to symptoms with a certain pattern, with s1 arising often in cases of the rare disease:

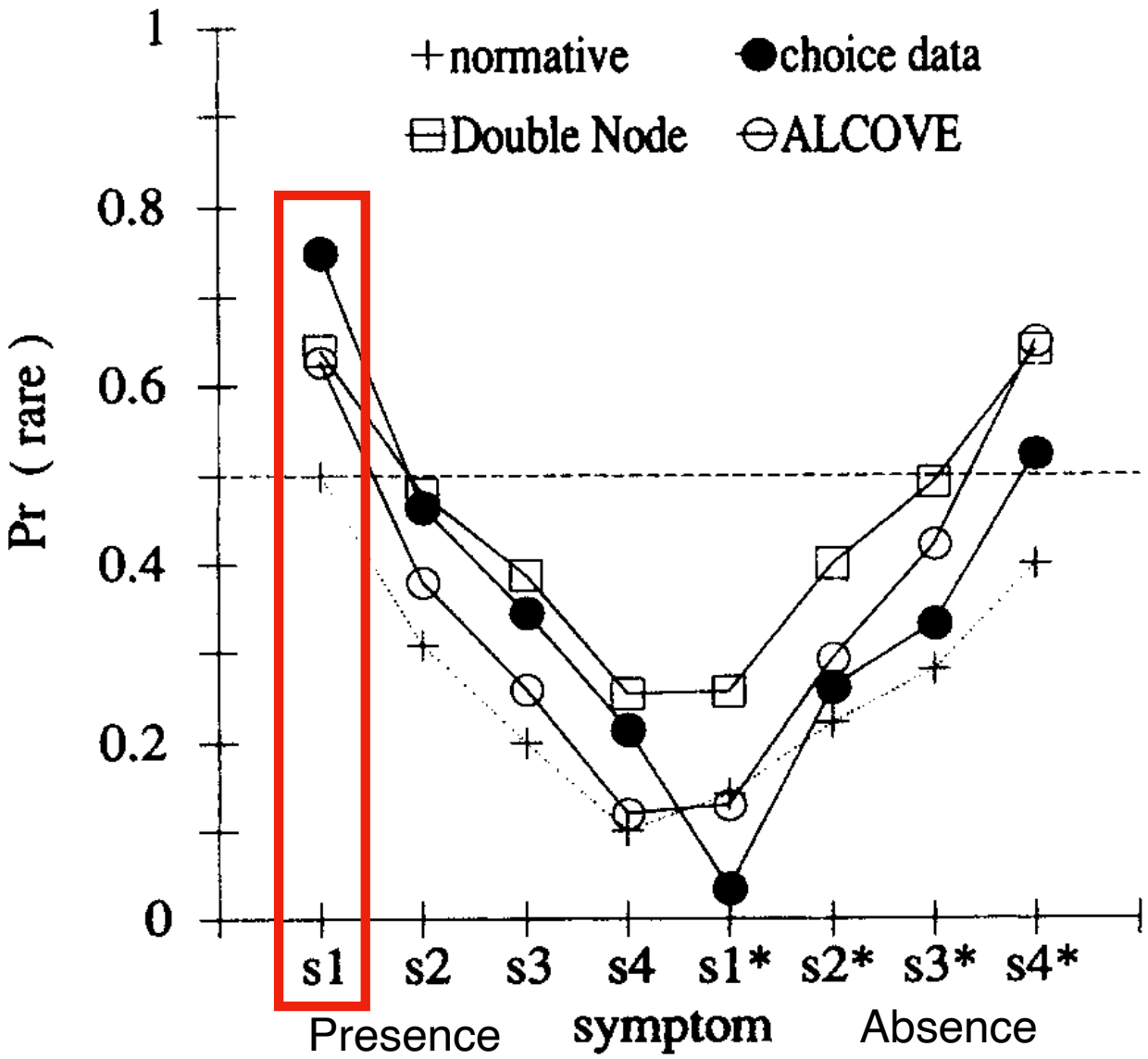
Symptom	Rare disease (R)	Common disease (C)
s1	0.6	0.2
s2	0.4	0.3
s3	0.3	0.4
s4	0.2	0.6

$$P(s1 \mid C) = 0.2$$
$$P(s1 \mid R) = 0.6$$

Both due to mismatch in base rates, the presence of s1 could mean either disease:

$$P(R \mid s1) = \frac{P(s1 \mid R)P(R)}{P(s1 \mid R)P(R) + P(s1 \mid C)P(C)} = 0.5 \qquad P(C \mid s1) = 0.5$$

Base rate neglect



Normative:

$$P(R \mid s1) = \frac{P(s1 \mid R)P(R)}{P(s1)}$$

The context model behaves like the normative model, since it stores all examples with their correct labels

Error driven learning forces exemplars to compete for the right to activate output nodes, resulting in base-rate neglect when s1 is presented in isolation

Psychological Review
1991, Vol. 98, No. 3, 409-429

0033-295X/91/$3.00

The Adaptive Nature of Human Categorization

John R. Anderson
Carnegie Mellon University

A rational model of human categorization behavior is presented that assumes that categorization reflects the derivation of optimal estimates of the probability of unseen features of objects. A Bayesian analysis is performed of what optimal estimations would be if categories formed a disjoint partitioning of the object space and if features were independently displayed within a category. This Bayesian analysis is placed within an incremental categorization algorithm. The resulting rational model accounts for effects of central tendency of categories, effects of specific instances, learning of linearly nonseparable categories, effects of category labels, extraction of basic level categories, base-rate effects, probability matching in categorization, and trial-by-trial learning functions. Although the rational model considers just 1 level of categorization, it is shown how predictions can be enhanced by considering higher and lower levels. Considering prediction at the lower, individual level allows integration of this rational analysis of categorization with the earlier rational analysis of memory (Anderson & Milson, 1989).

Anderson (1990) presented a rational analysis of human cognition. The term *rational* derives from similar "rational-man" analyses in economics. Rational analyses in other fields are sometimes called *adaptationist analyses.* Basically, they are efforts to explain the behavior in some domain on the assumption that the behavior is optimized with respect to some criteria of adaptive importance. This article begins with a general characterization of how one develops a rational theory of a particular cognitive phenomenon. Then I present the basic theory of categorization developed in Anderson (1990) and review the applications from that book. Since the writing of the book, the steps involved in a research program that attempts to understand cognition in terms of its adaptation to the environment:

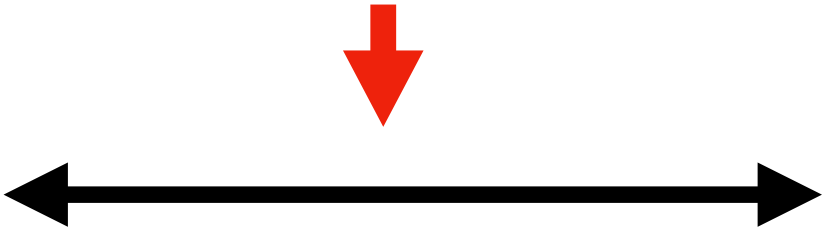
1. The first task is to specify what the system is trying to optimize. Perhaps such models are ultimately to be justified in terms of maximizing some evolutionary criterion like number of surviving offspring. However, this is not a very workable criterion in most applications. Thus, economics uses wealth as the variable to be optimized; optimal foraging theory (Stephens & Krebs, 1986) often uses caloric intake; and the rational theory of memory (Anderson & Milson, 1989) uses retrieval of relevant experiences from the past.

Is there another option besides learning a single prototype, or storing every single exemplar?

Exemplars

Another option?

Prototype



Somewhere between exemplars and prototypes…

The mind could represent a concept, like “boat”, as a collection of *several* prototypes... and even specific examples.

We want **model flexibility** depending on the **structure of the data**.

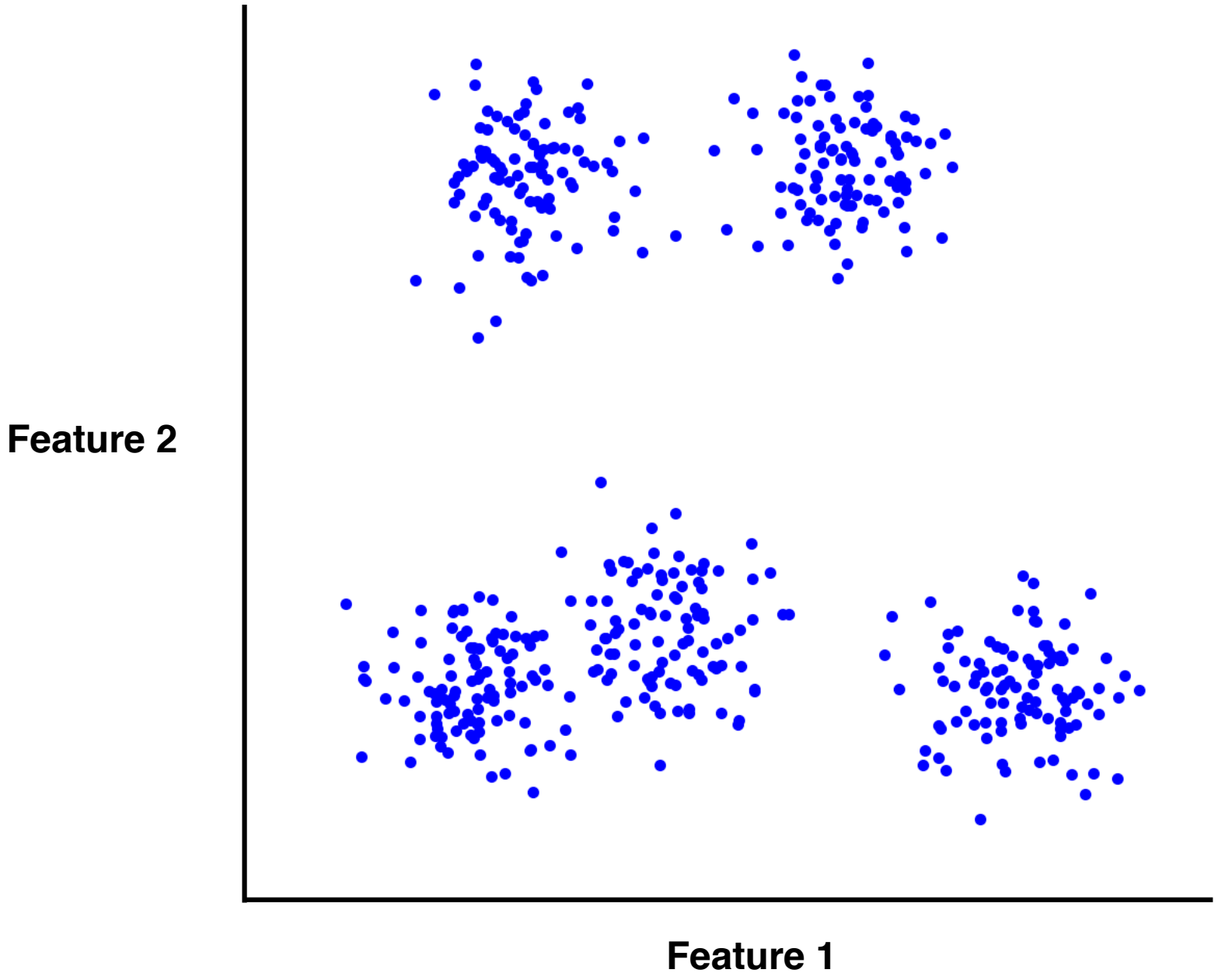
prototype 1

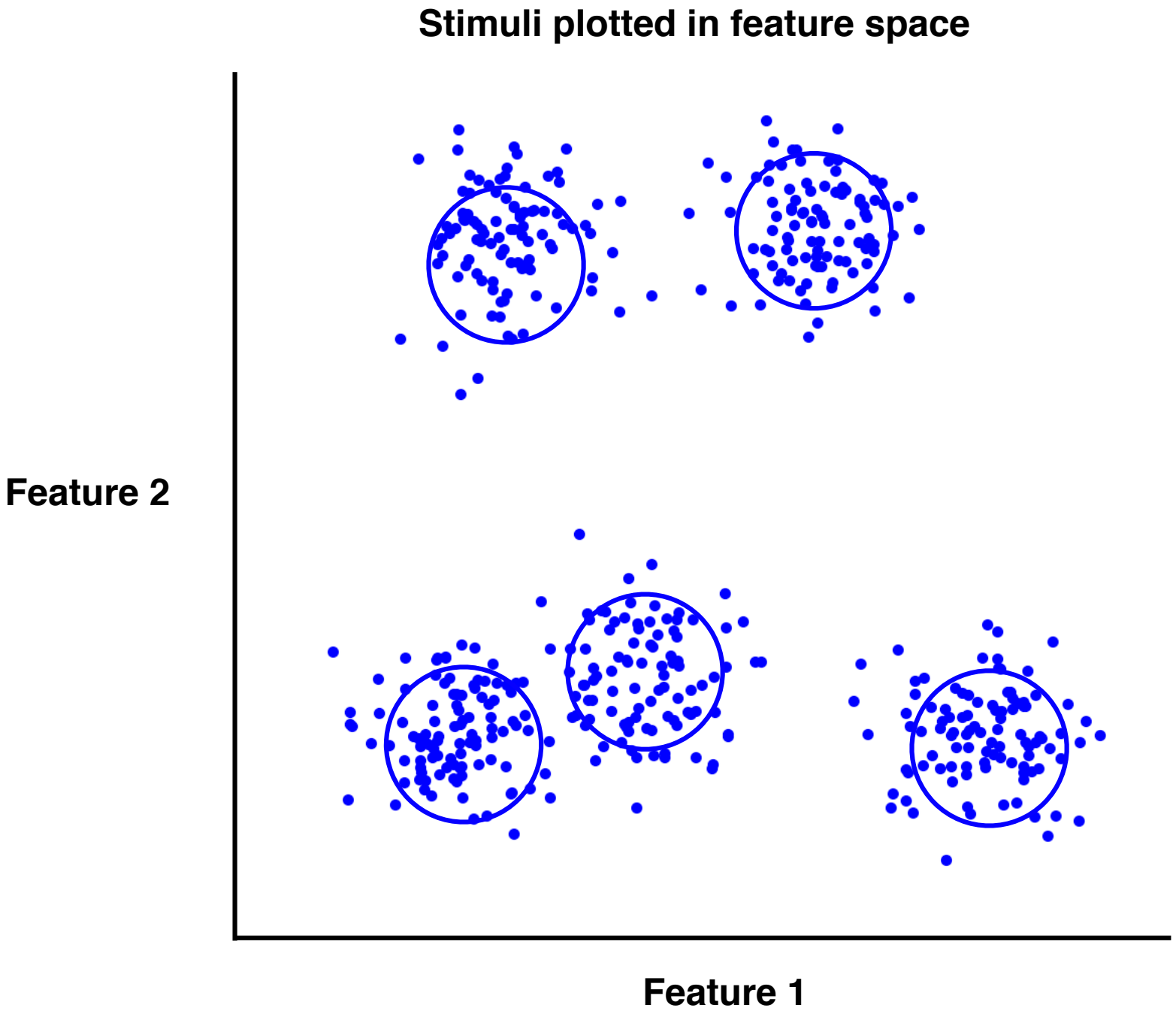
prototype 2

prototype 3

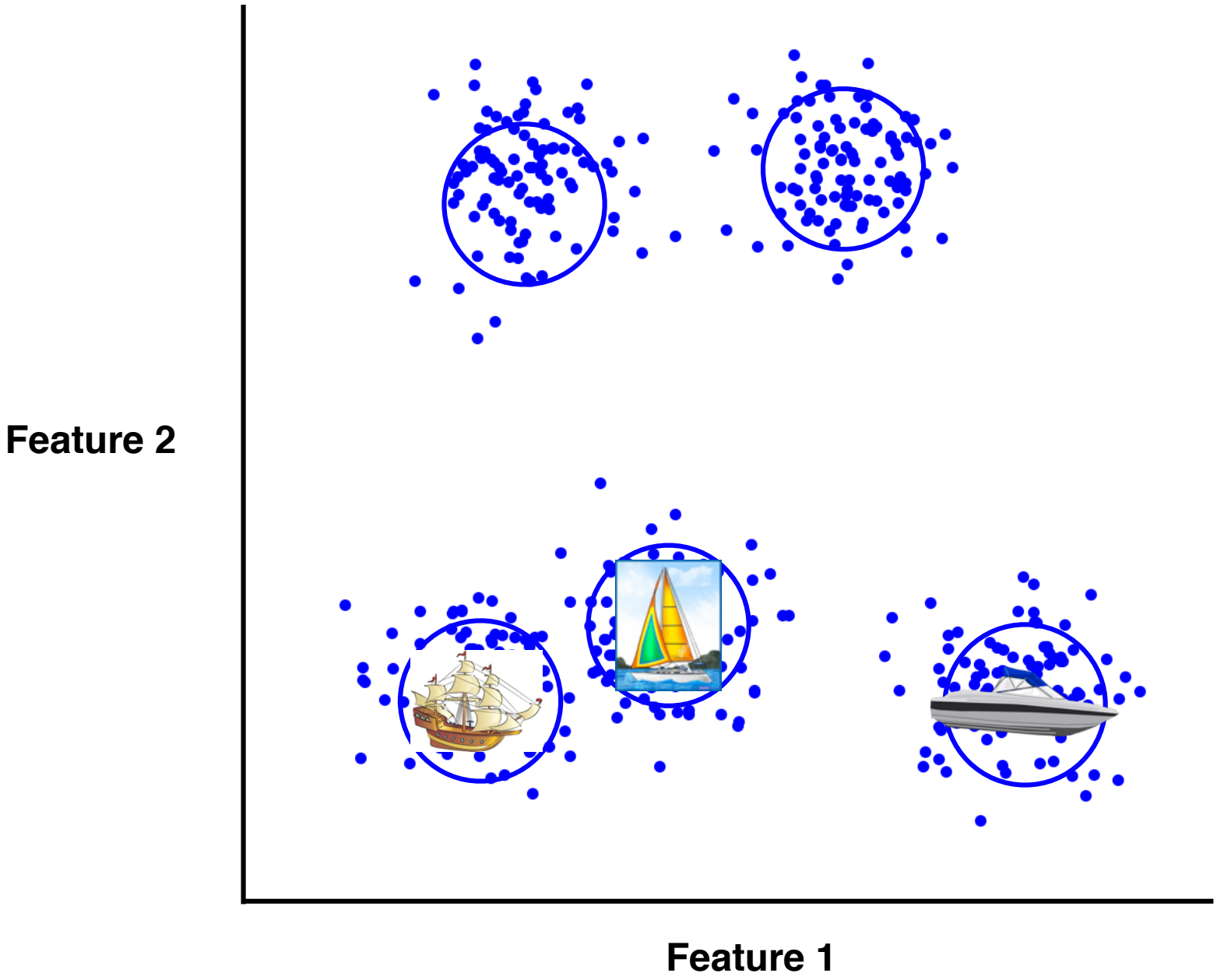
. . .

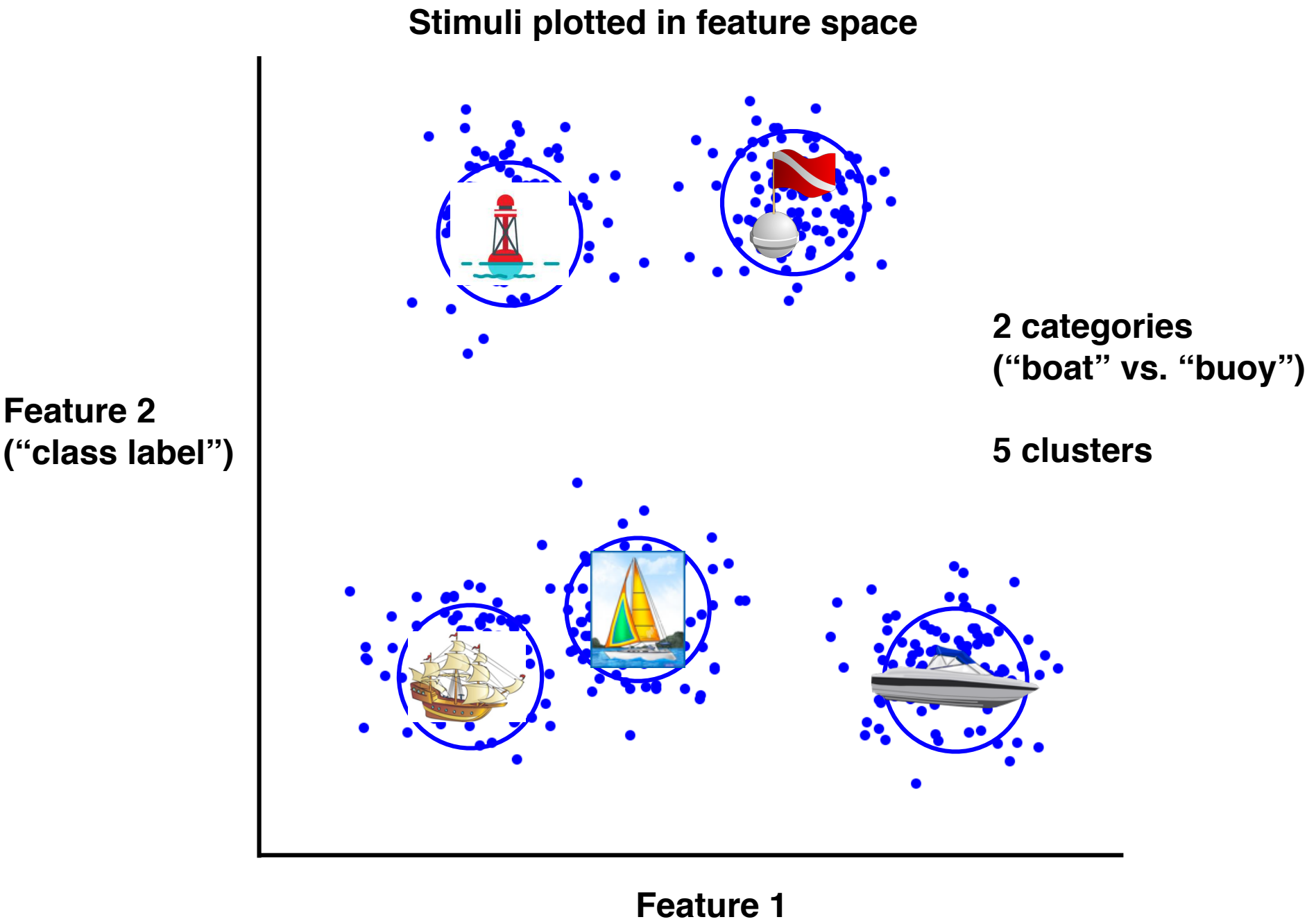
Stimuli plotted in feature space (possibly many dimensions)




Stimuli plotted in feature space
Feature 2
Feature 1

Stimuli plotted in feature space

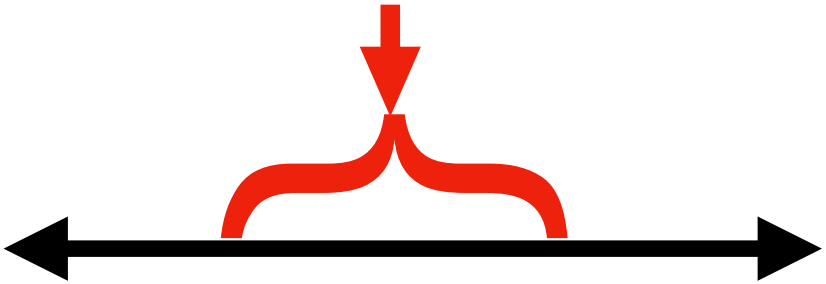



Stimuli plotted in feature space
Feature 2
("class label")
2 categories
("boat" vs. "buoy")
5 clusters
Feature 1

The Rational model can discover cluster structure, interpolating between prototype and exemplar models in a principled way

Exemplars

Rational model



Prototype

Anderson's "rational model"

Base rate of CLUSTER *k* (not category) is the probability of a new stimulus belonging to that cluster

(this model is called the "Chinese Restaurant Process" or "Dirichlet Process" in statistics and machine learning)

Base rate assignment to existing cluster *k*

$$P(k) = \frac{n_k}{\alpha + n}$$

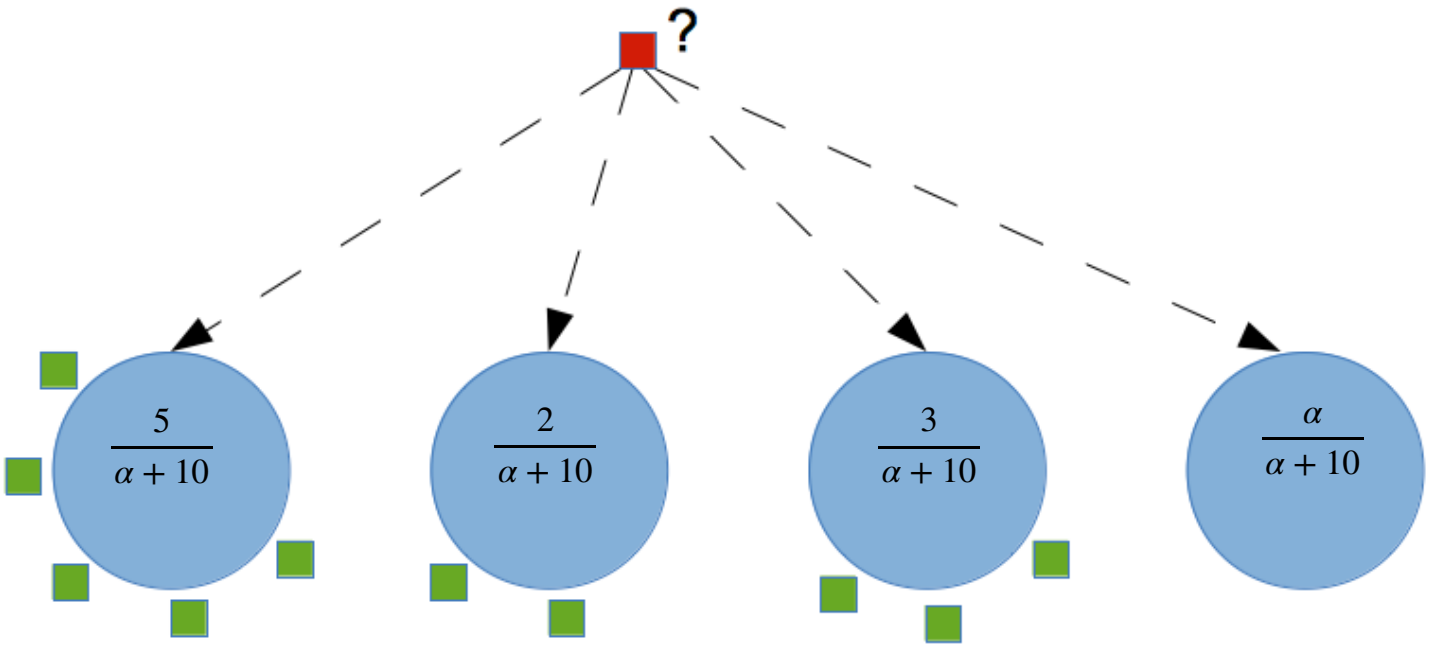
Base rate of forming a new cluster

$$P(\textbf{new}) = \frac{\alpha}{\alpha + n}$$

n_k number of stimuli in *k*

n number of stimuli total

α parameter that controls likelihood of new cluster



Rational model

Base rate of CLUSTER *k* (not category) probability for new stimuli (prior)
(this model is called the "Chinese Restaurant Process" or "Dirichlet Process" in statistics and machine learning)

Base rate assignment to existing cluster *k*

$$P(k) = \frac{n_k}{\alpha + n}$$

Base rate of forming a new cluster

$$P(\textbf{new}) = \frac{\alpha}{\alpha + n}$$

n_k number of stimuli in *k*

n number of stimuli total

α parameter that controls likelihood of new cluster

Scoring similarity of new stimulus *F* to existing cluster k (likelihood)

$$P(F \mid k) = \prod_{D_i} P(F_i \mid k)$$

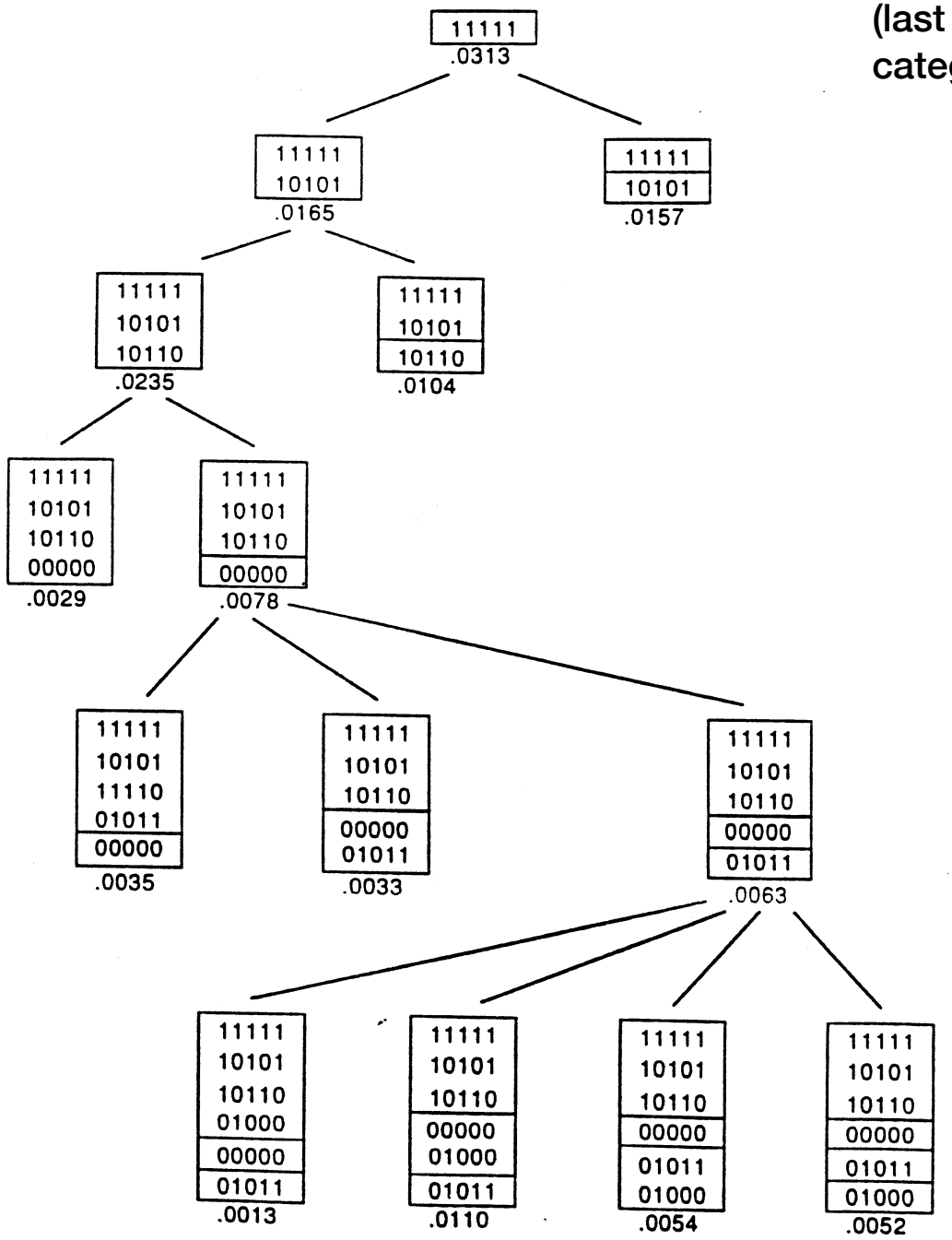
The form of this distribution depends on whether features are discrete or continuous. **Class labels** are treated just like any other feature (see paper for details)

Using Bayes' rule, we to classify the new item *F* in a cluster k (posterior)

$$P(k \mid F) = \frac{P(F \mid k)P(k)}{P(F \mid \textbf{new})P(\textbf{new}) + \sum_{k'} P(F \mid k')P(k')}$$

Rational model in action

(last dimension is category label)



Rational model in action

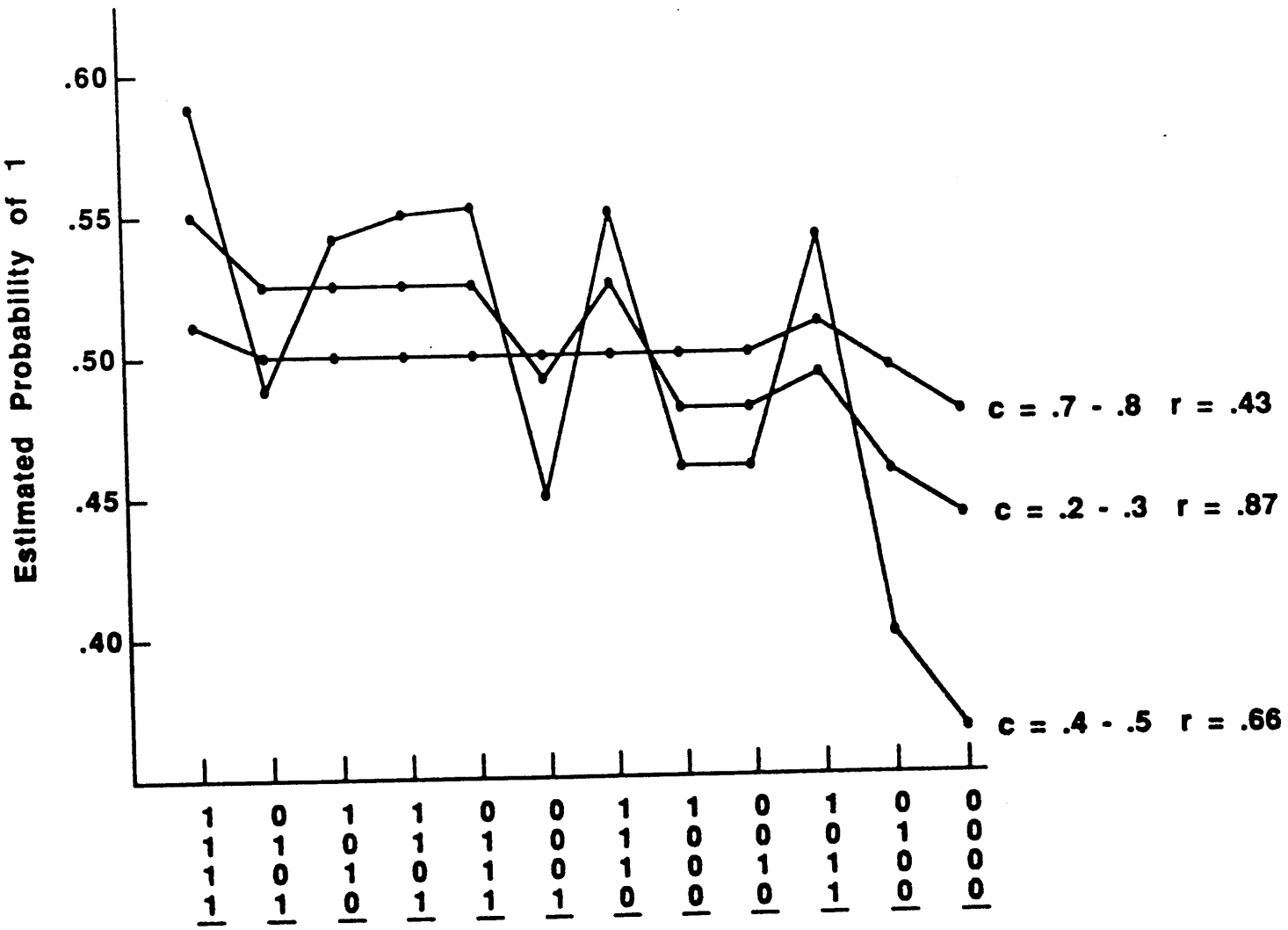
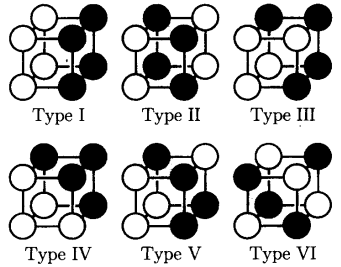
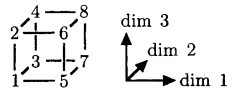


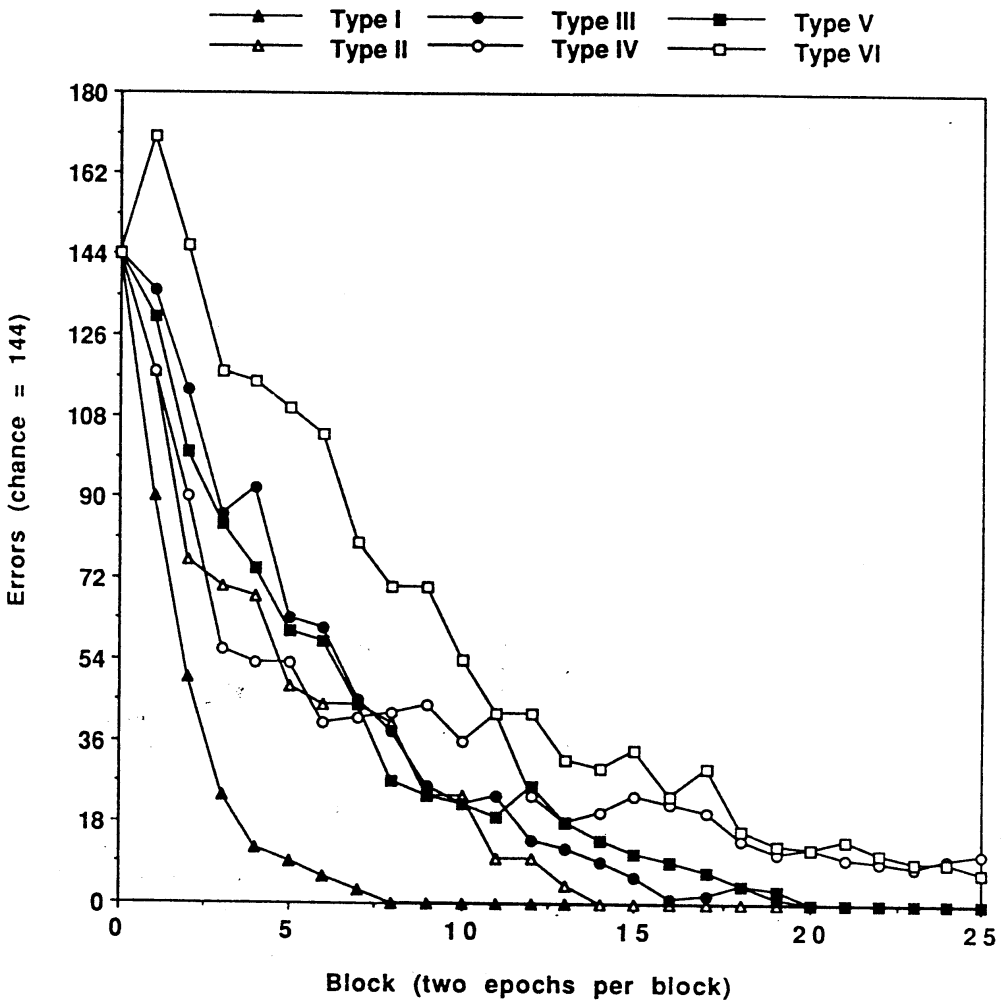
Figure 2. Estimated probability of Category 1 for the 12 test stimuli in the first experiment of Medin and Schaffer (1978). (Different functions are for different ranges of the coupling probability.)

Comparing SHJ learning curves for people and rational model

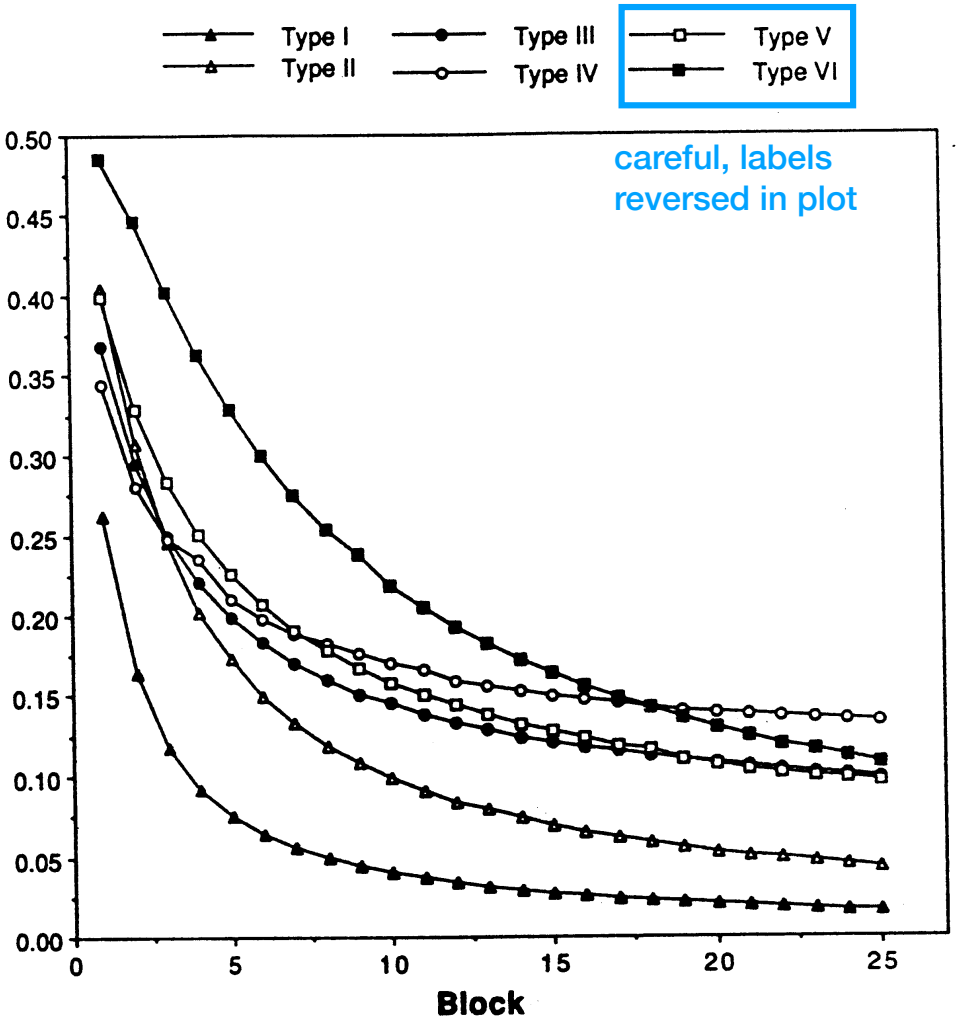




People



Model



ImageNet Classification with Deep Convolutional Neural Networks

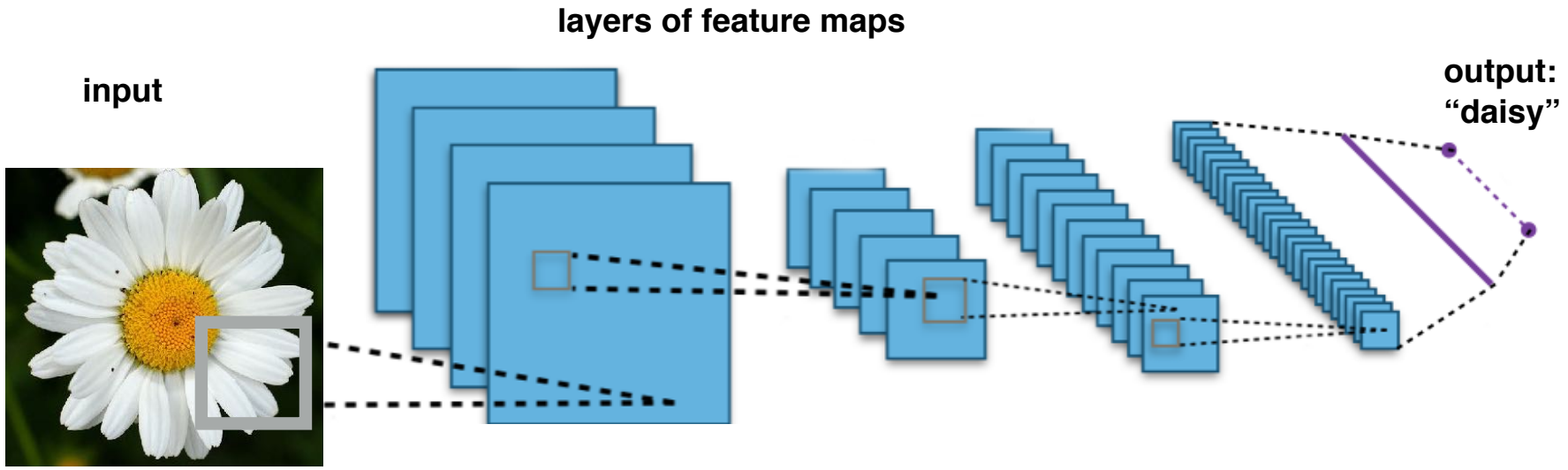
Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

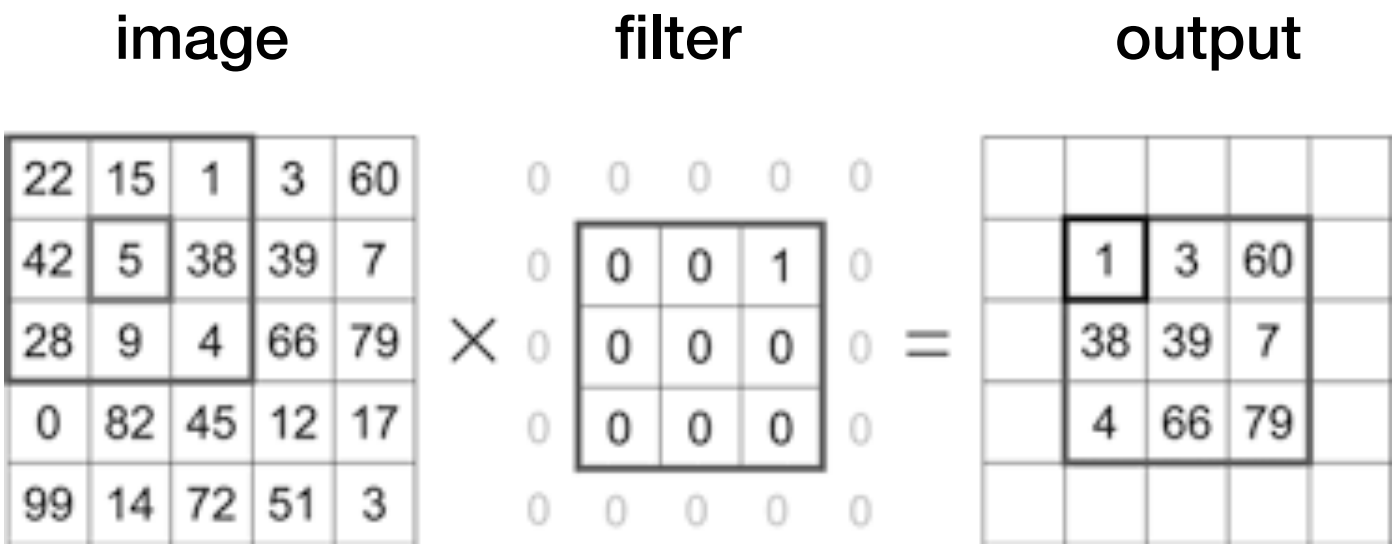
Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

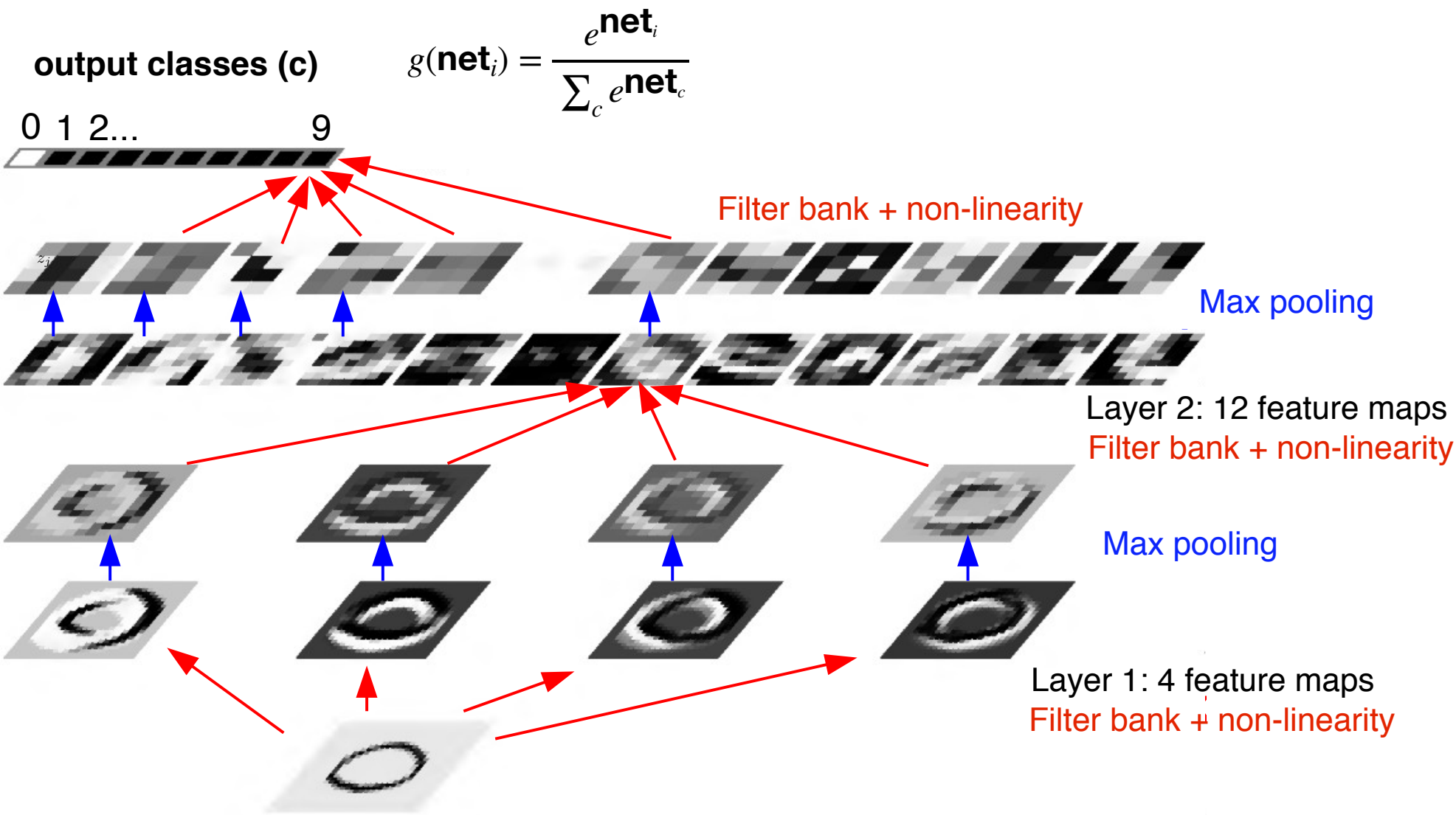


convolving an image with a filter

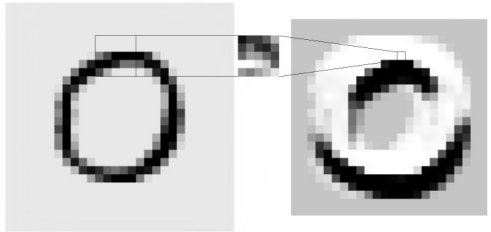
We slide the filter across the image to produce an output image



Deep convolutional neural network (convnet) for vision

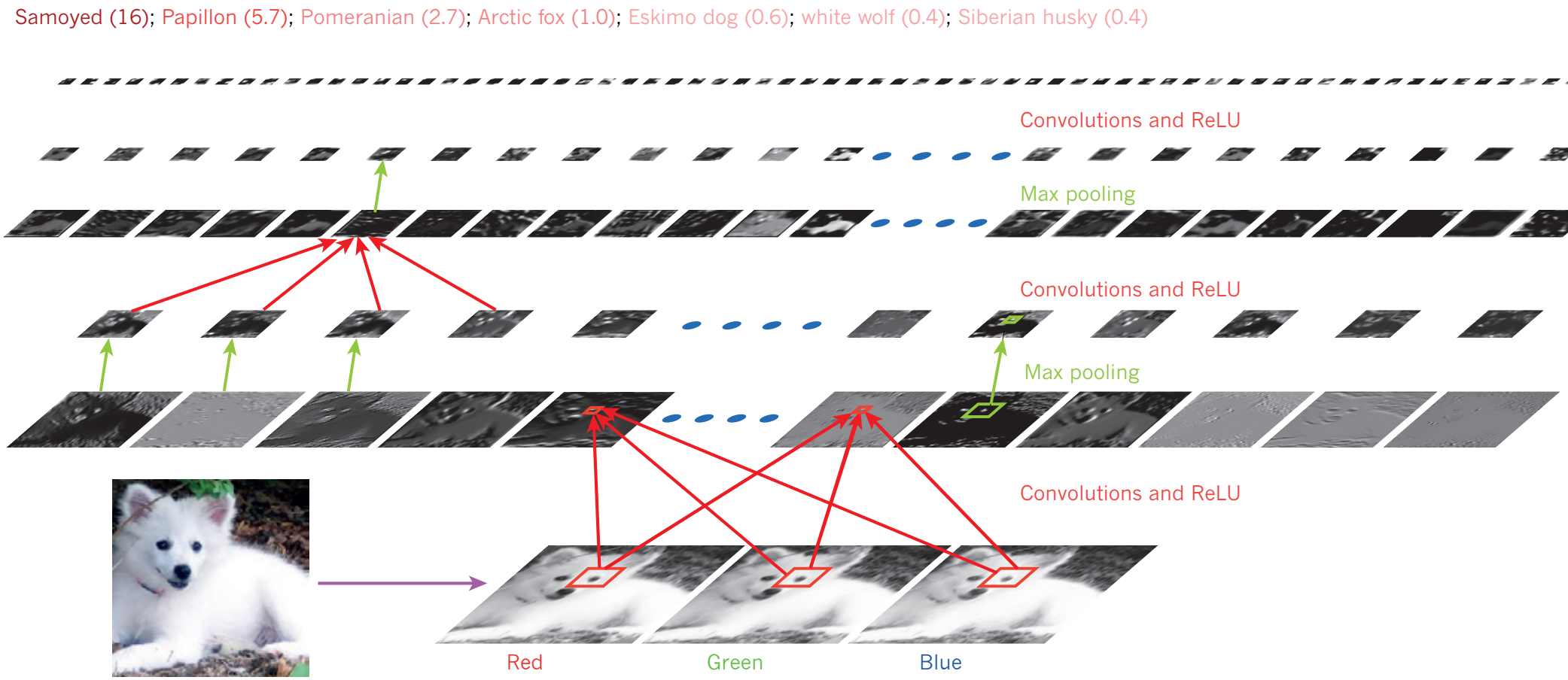


example convolution



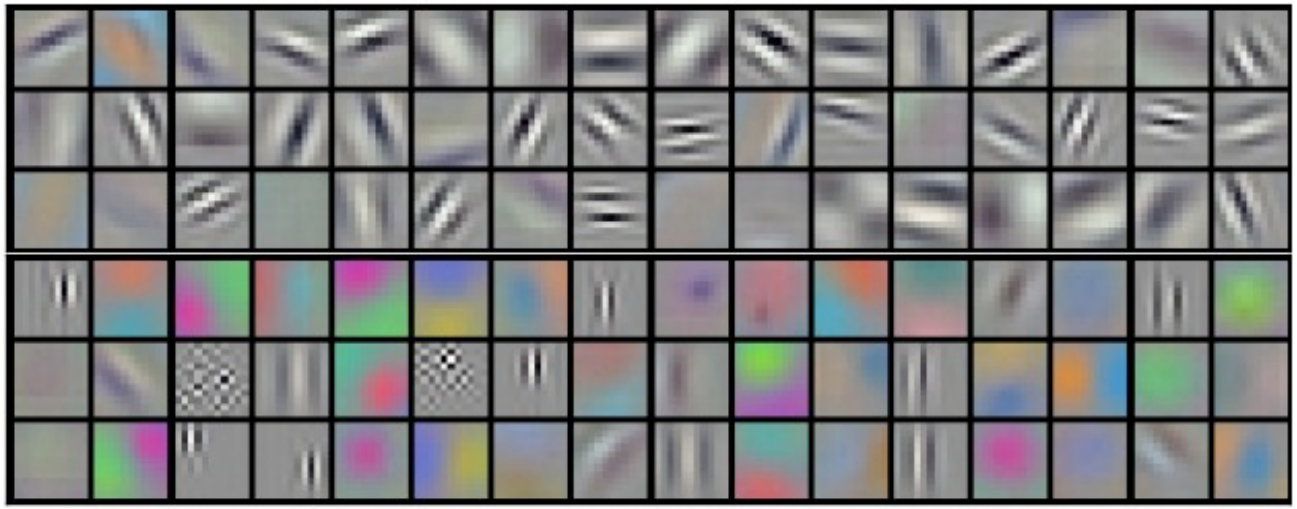
Slide courtesy of Yann LeCun

Deep convolutional neural network



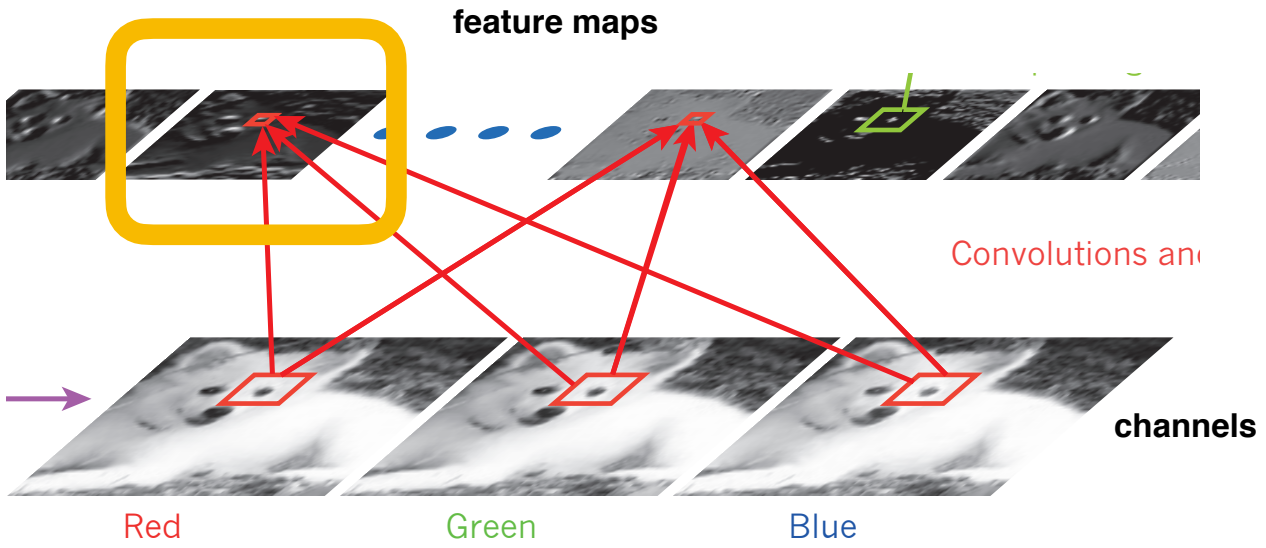
From LeCun, Bengio, & Hinton (2015).

Examples of learned filters



(Krizhevsky et al., 2012)

Let's go through computing with convolutions to build a feature map…



Filter 0

Filter 1

Next layer of network

Input Volume (+pad 1) (7x7x3)

x[:,:,0]

0	0	0	0	0	0	0
0	2	0	0	0	1	0
0	2	2	0	2	0	0
0	1	0	2	0	0	0
0	2	0	2	0	1	0
0	0	0	2	0	2	0
0	0	0	0	0	0	0

Channel 0

(e.g., red)

x[:,:,1]

0	0	0	0	0	0	0
0	1	0	2	0	1	0
0	0	0	0	0	2	0
0	0	0	2	1	0	0
0	2	1	1	1	2	0
0	1	1	0	0	0	0
0	0	0	0	0	0	0

Channel 1

(e.g., green)

x[:,:,2]

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	1	1	0	0	0
0	1	1	2	2	0	0
0	2	2	1	0	0	0
0	0	0	1	1	2	0
0	0	0	0	0	0	0

Channel 2

(e.g., blue)

dot product (element-wise multiply then sum)

dot product

dot product

Filter W0 (3x3x3)

w0[:,:,0]

0	1	0
-1	1	0
0	1	1

w0[:,:,1]

-1	-1	1
0	1	0
0	1	0

w0[:,:,2]

1	-1	-1
0	0	0
0	0	0

Bias b0 (1x1x1)

b0[:,:,0]

1

Filter W1 (3x3x3)

w1[:,:,0]

0	-1	-1
0	0	1
1	1	-1

w1[:,:,1]

1	1	-1

w1[:,:,2]

0	-1	0
-1	1	0
-1	-1	0

Bias b1 (1x1x1)

b1[:,:,0]

0

sum

Output Volume (3x3x2)

o[:,:,0]

8	6	5
7	9	2
-1	7	3

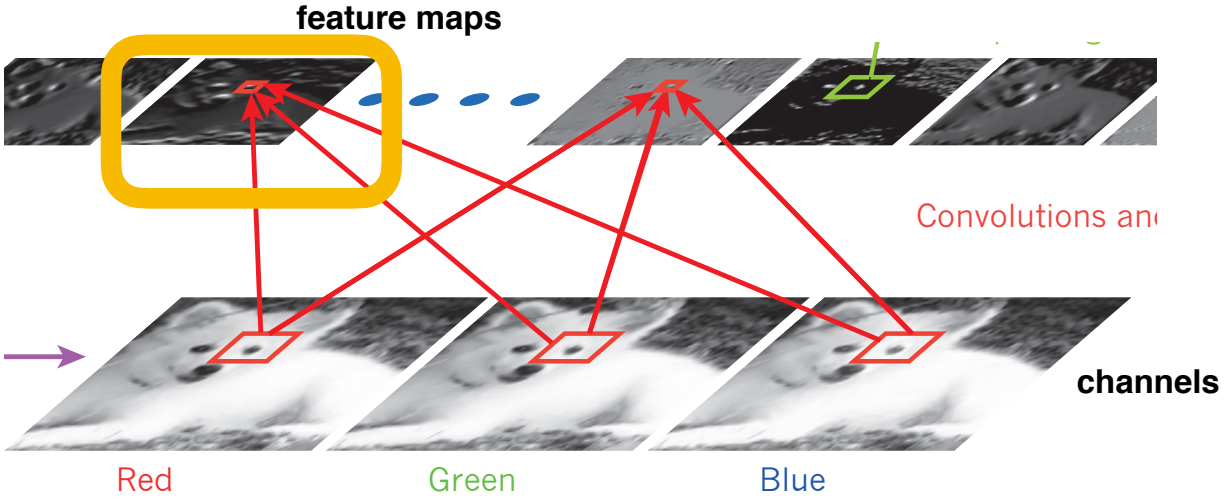
Feature map 0

o[:,:,1]

2	3	7
-2	1	-2
-5	-1	-2

Feature map 1

Credit for demo: http://cs231n.github.io/convolutional-networks/



Filter 0 **Filter 1** **Next layer of network**

Input Volume (+pad 1) (7x7x3)

`x[:,:,0]`

0	0	0	0	0	0	0
0	2	0	0	2	2	0
0	2	2	1	2	1	0
0	1	0	2	1	0	0
0	2	0	2	0	1	0
0	0	0	2	0	2	0
0	0	0	0	0	0	0

Channel 0

(e.g., red)

`x[:,:,1]`

0	0	0	0	0	0	0
0	1	0	2	0	1	0
0	0	0	0	0	2	0
0	0	0	2	1	0	0
0	2	1	1	1	2	0
0	1	1	2	2	2	0
0	0	0	0	0	0	0

Channel 1

(e.g., green)

`x[:,:,2]`

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	1	1	0	0	0
0	1	1	2	2	0	0
0	2	2	1	0	0	0
0	0	0	1	1	2	0
0	0	0	0	0	0	0

Channel 2

(e.g., blue)

Filter W0 (3x3x3)

`w0[:,:,0]`

0	1	0
-1	1	0
0	1	1

`w0[:,:,1]`

-1	-1	1
0	1	0
0	1	0

`w0[:,:,2]`

1	-1	-1
0	0	0
0	0	0

Bias b0 (1x1x1)

`b0[:,:,0]`

1

Filter W1 (3x3x3)

`w1[:,:,0]`

0	-1	-1
0	0	1
1	1	-1

`w1[:,:,1]`

0	-1	-1
-1	1	1
1	1	-1

`w1[:,:,2]`

0	-1	0
-1	1	0
-1	-1	0

Bias b1 (1x1x1)

`b1[:,:,0]`

0

Output Volume (3x3x2)

`o[:,:,0]`

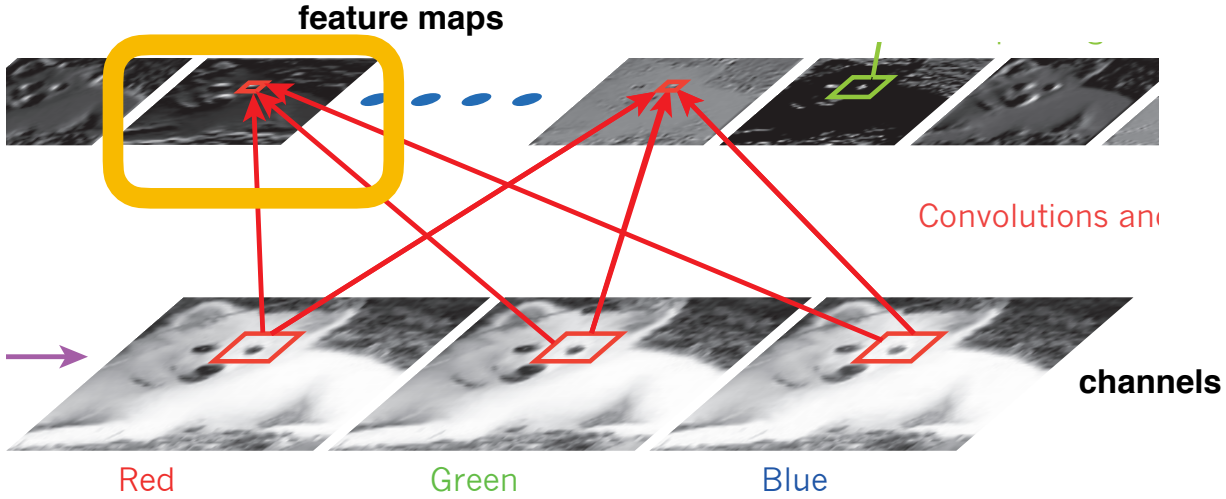
8	6	5
7	9	2
-1	7	3

Feature map 0

`o[:,:,1]`

2	3	7
-2	1	-2
-5	-1	-2

Feature map 1



Filter 0 **Filter 1** **Next layer of network**

Input Volume (+pad 1) (7x7x3)

x[:,:,0]

0	0	0	0	0	0	0
0	2	0	0	2	2	0
0	2	2	1	2	1	0
0	1	0	2	1	0	0
0	2	0	2	0	1	0
0	0	0	2	0	2	0
0	0	0	0	0	0	0

Channel 0

(e.g., red)

x[:,:,1]

0	0	0	0	0	0	0
0	1	0	2	0	1	0
0	0	0	0	0	2	0
0	0	0	2	1	0	0
0	2	1	1	1	2	0
0	1	1	2	2	2	0
0	0	0	0	0	0	0

Channel 1

(e.g., green)

x[:,:,2]

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	1	1	0	0	0
0	1	1	2	2	0	0
0	2	2	1	0	0	0
0	0	0	1	1	2	0
0	0	0	0	0	0	0

Channel 2

(e.g., blue)

Filter W0 (3x3x3)

w0[:,:,0]

0	1	0
-1	1	0
0	1	1

w0[:,:,1]

-1	-1	1
0	1	0
0	1	0

w0[:,:,2]

1	-1	-1
0	0	0
0	0	0

Bias b0 (1x1x1)

b0[:,:,0]

1

Filter W1 (3x3x3)

w1[:,:,0]

0	-1	-1
0	0	1
1	1	-1

w1[:,:,1]

0	-1	-1
-1	1	1
1	1	-1

w1[:,:,2]

0	-1	0
-1	1	0
-1	-1	0

Bias b1 (1x1x1)

b1[:,:,0]

0

Output Volume (3x3x2)

o[:,:,0]

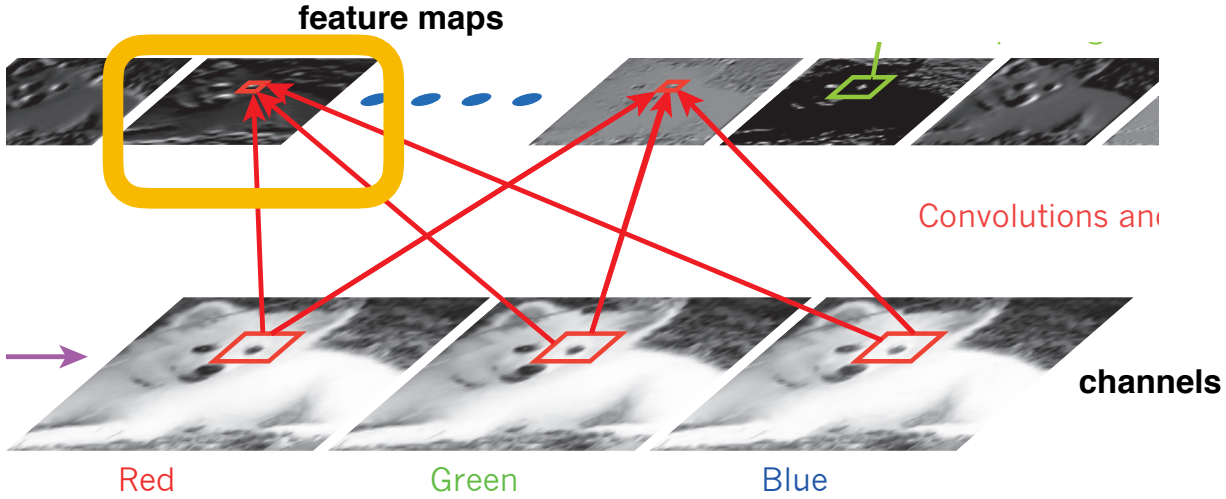
8	6	5
7	9	2
-1	7	3

Feature map 0

o[:,:,1]

2	3	7
-2	1	-2
-5	-1	-2

Feature map 1



Filter 0 **Filter 1** **Next layer of network**

Input Volume (+pad 1) (7x7x3)

x[:,:,0]

0	0	0	0	0	0	0
0	2	0	0	2	2	0
0	2	2	1	2	1	0
0	1	0	2	1	0	0
0	2	0	2	0	1	0
0	0	0	2	0	2	0
0	0	0	0	0	0	0

x[:,:,1]

0	0	0	0	0	0	0
0	1	0	2	0	1	0
0	0	0	0	0	2	0
0	0	0	2	1	0	0
0	2	1	1	1	2	0
0	1	1	2	2	2	0
0	0	0	0	0	0	0

x[:,:,2]

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	1	1	0	0	0
0	1	1	2	2	0	0
0	2	2	1	0	0	0
0	0	0	1	1	2	0
0	0	0	0	0	0	0

Channel 0 (e.g., red)

Channel 1 (e.g., green)

Channel 2 (e.g., blue)

Filter W0 (3x3x3)

w0[:,:,0]

0	1	0
-1	1	0
0	1	1

w0[:,:,1]

-1	-1	1
0	1	0
0	1	0

w0[:,:,2]

1	-1	-1
0	0	0
0	0	0

Bias b0 (1x1x1)

b0[:,:,0]

1

Filter W1 (3x3x3)

w1[:,:,0]

0	-1	-1
0	0	1
1	1	-1

w1[:,:,1]

0	-1	-1
-1	1	1
1	1	-1

w1[:,:,2]

0	-1	0
-1	1	0
-1	-1	0

Bias b1 (1x1x1)

b1[:,:,0]

0

Output Volume (3x3x2)

o[:,:,0]

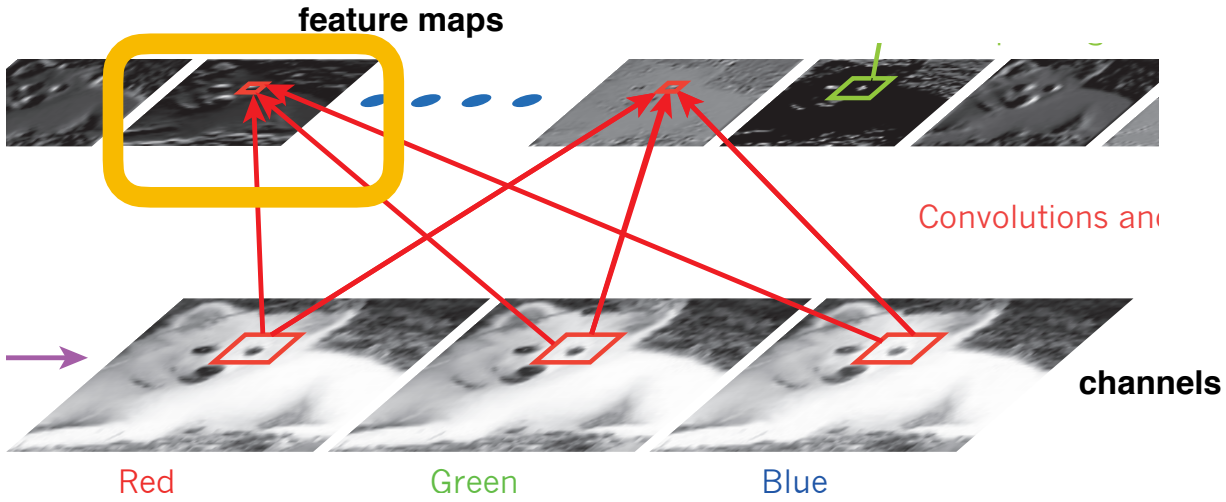
8	6	5
7	9	2
-1	7	3

Feature map 0

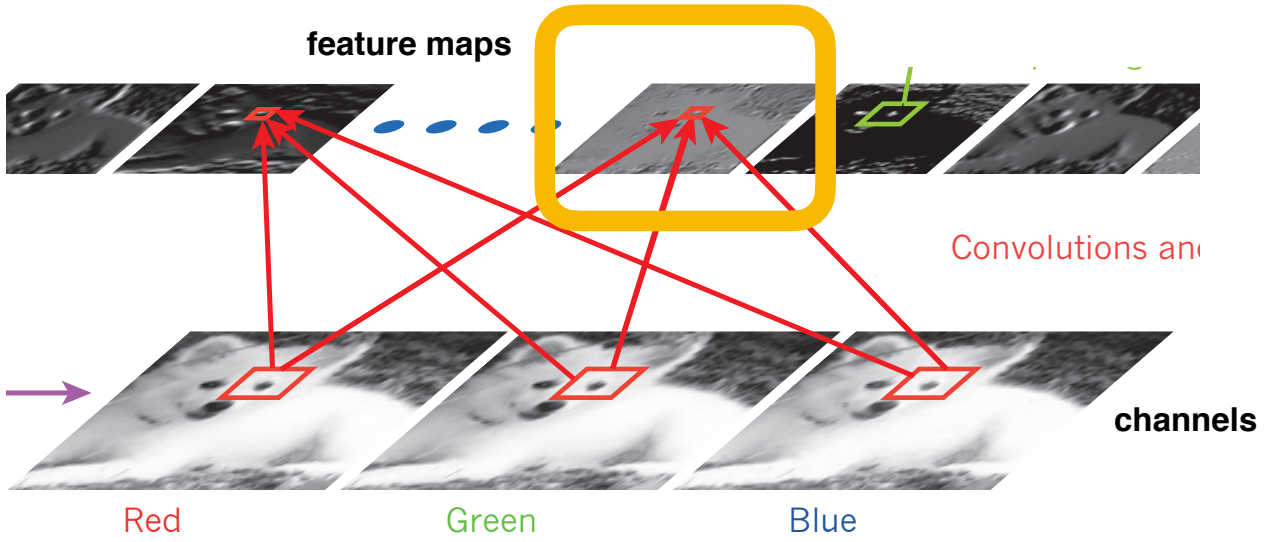
o[:,:,1]

2	3	7
-2	1	-2
-5	-1	-2

Feature map 1



Let's skip to compute the next feature map…



Filter 0 **Filter 1** **Next layer of network**

Input Volume (+pad 1) (7x7x3)

Filter W0 (3x3x3)

Filter W1 (3x3x3)

Output Volume (3x3x2)

Channel 0 (e.g., red)

Channel 1 (e.g., green)

Channel 2 (e.g., blue)

x[:,:,0]

0	0	0	0	0	0	0
0	2	0	0	2	2	0
0	2	2	1	2	1	0
0	1	0	2	1	0	0
0	2	0	2	0	1	0
0	0	0	2	0	2	0
0	0	0	0	0	0	0

x[:,:,1]

0	0	0	0	0	0	0
0	1	0	2	0	1	0
0	0	0	0	0	2	0
0	0	0	2	1	0	0
0	2	1	1	1	2	0
0	1	1	2	2	2	0
0	0	0	0	0	0	0

x[:,:,2]

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	1	1	0	0	0
0	1	1	2	2	0	0
0	2	2	1	0	0	0
0	0	0	1	1	2	0
0	0	0	0	0	0	0

w0[:,:,0]

0	1	0
-1	1	0
0	1	1

w0[:,:,1]

-1	-1	1
0	1	0
0	1	0

w0[:,:,2]

1	-1	-1
0	0	0
0	0	0

Bias b0 (1x1x1)

b0[:,:,0]

1

w1[:,:,0]

0	-1	-1
0	0	1
1	1	-1

w1[:,:,1]

0	-1	-1
-1	1	1
1	1	-1

w1[:,:,2]

0	-1	0
-1	1	0
-1	-1	0

Bias b1 (1x1x1)

b1[:,:,0]

0

o[:,:,0]

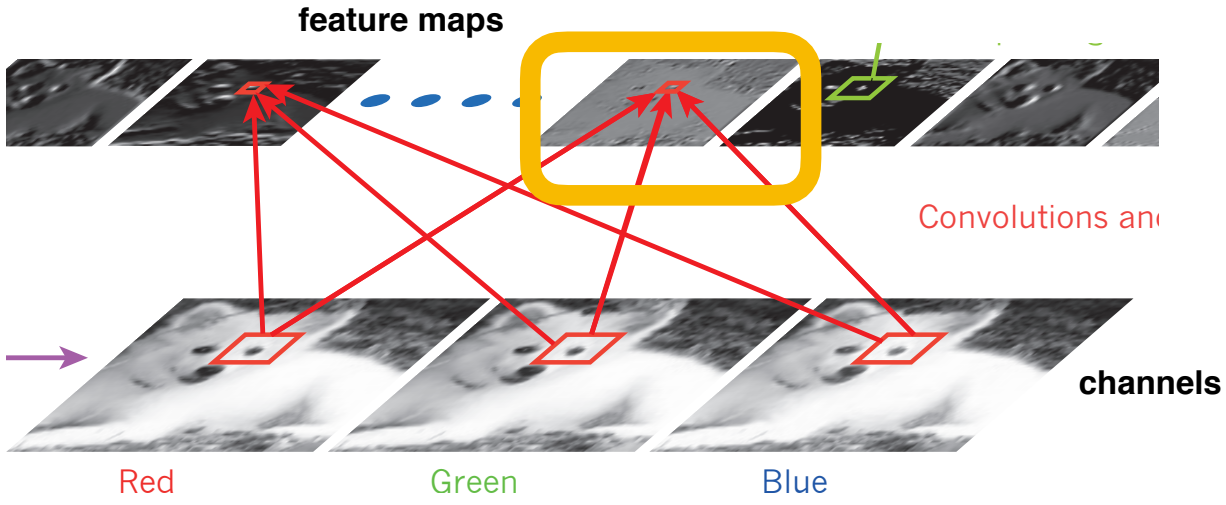
8	6	5
7	9	2
-1	7	3

Feature map 0

o[:,:,1]

2	3	7
-2	1	-2
-5	-1	-2

Feature map 1



Channel 0
(e.g., red)

Channel 1
(e.g., green)

Channel 2
(e.g., blue)

Filter 0

Filter 1

Next layer of network

Input Volume (+pad 1) (7x7x3)

`x[:,:,0]`

0	0	0	0	0	0	0
0	2	0	0	2	2	0
0	2	2	1	2	1	0
0	1	0	2	1	0	0
0	2	0	2	0	1	0
0	0	0	2	0	2	0
0	0	0	0	0	0	0

`x[:,:,1]`

0	0	0	0	0	0	0
0	1	0	2	0	1	0
0	0	0	0	0	2	0
0	0	0	2	1	0	0
0	2	1	1	1	2	0
0	1	1	2	2	2	0
0	0	0	0	0	0	0

`x[:,:,2]`

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	1	1	0	0	0
0	1	1	2	2	0	0
0	2	2	1	0	0	0
0	0	0	1	1	2	0
0	0	0	0	0	0	0

Filter W0 (3x3x3)

`w0[:,:,0]`

0	1	0
-1	1	0
0	1	1

`w0[:,:,1]`

-1	-1	1
0	1	0
0	1	0

`w0[:,:,2]`

1	-1	-1
0	0	0
0	0	0

Bias b0 (1x1x1)

`b0[:,:,0]`

1

Filter W1 (3x3x3)

`w1[:,:,0]`

0	-1	-1
0	0	1
1	1	-1

`w1[:,:,1]`

0	-1	-1
-1	1	1
1	1	-1

`w1[:,:,2]`

0	-1	0
-1	1	0
-1	-1	0

Bias b1 (1x1x1)

`b1[:,:,0]`

0

Output Volume (3x3x2)

`o[:,:,0]`

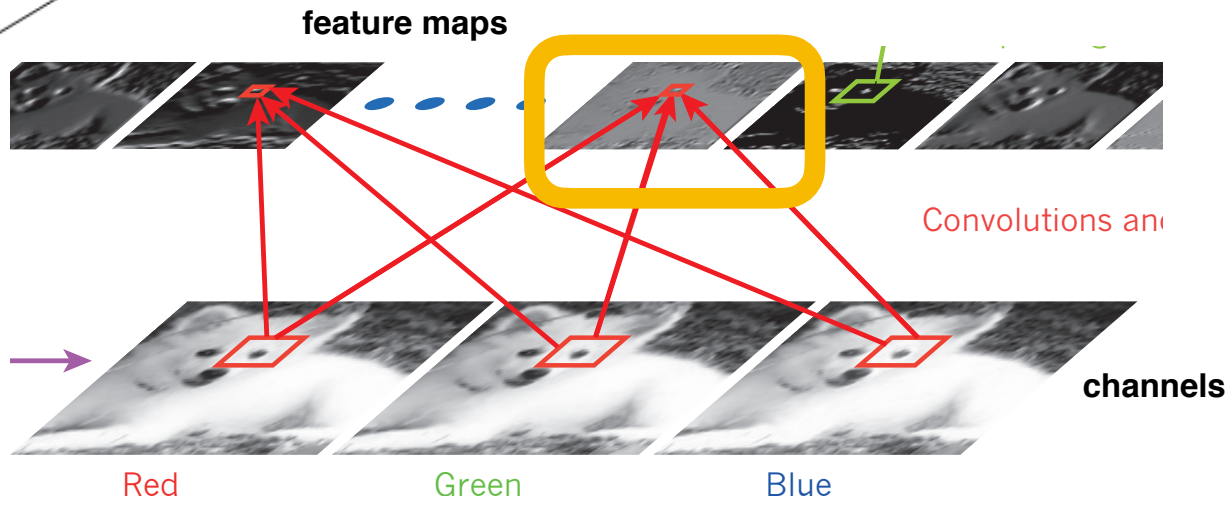
8	6	5
7	9	2
-1	7	3

Feature map 0

`o[:,:,1]`

2	3	7
-2	1	-2
-5	-1	-2

Feature map 1

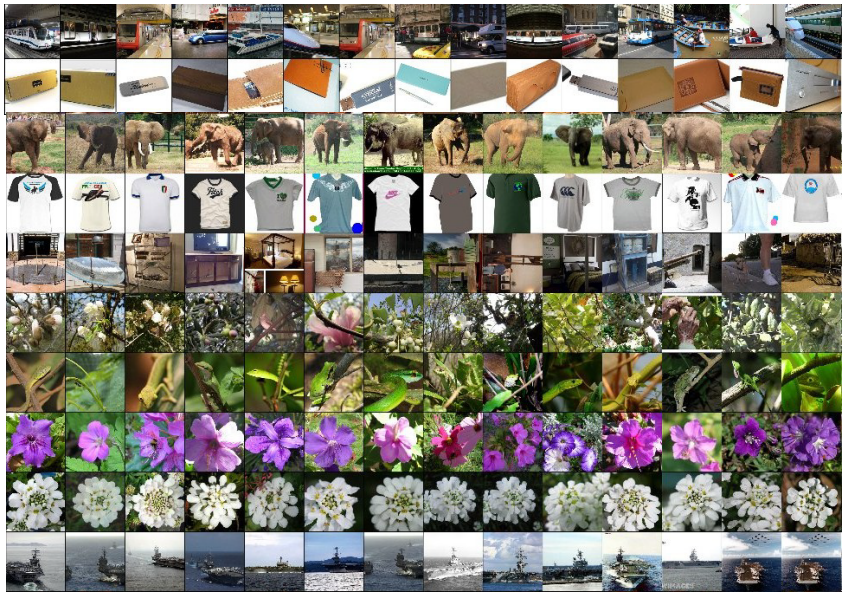


ConvNets are trained via gradient descent

ConvNets are trained on a lot of data

Training data (ImageNet)

- Usually trained on ImageNet
  - 1.2 million images with labels
  - 1000 categories



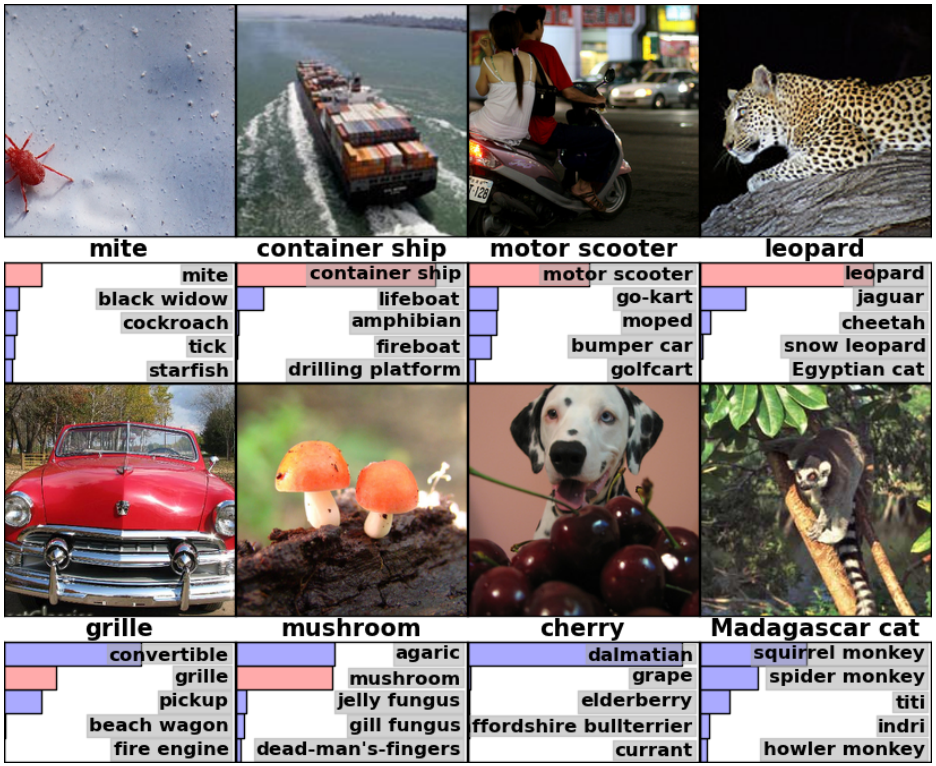
These models have millions of learnable parameters

AlexNet
8 layers
~60 million parameters

GoogLeNet
22 layers
~5 million parameters

Example results on test images

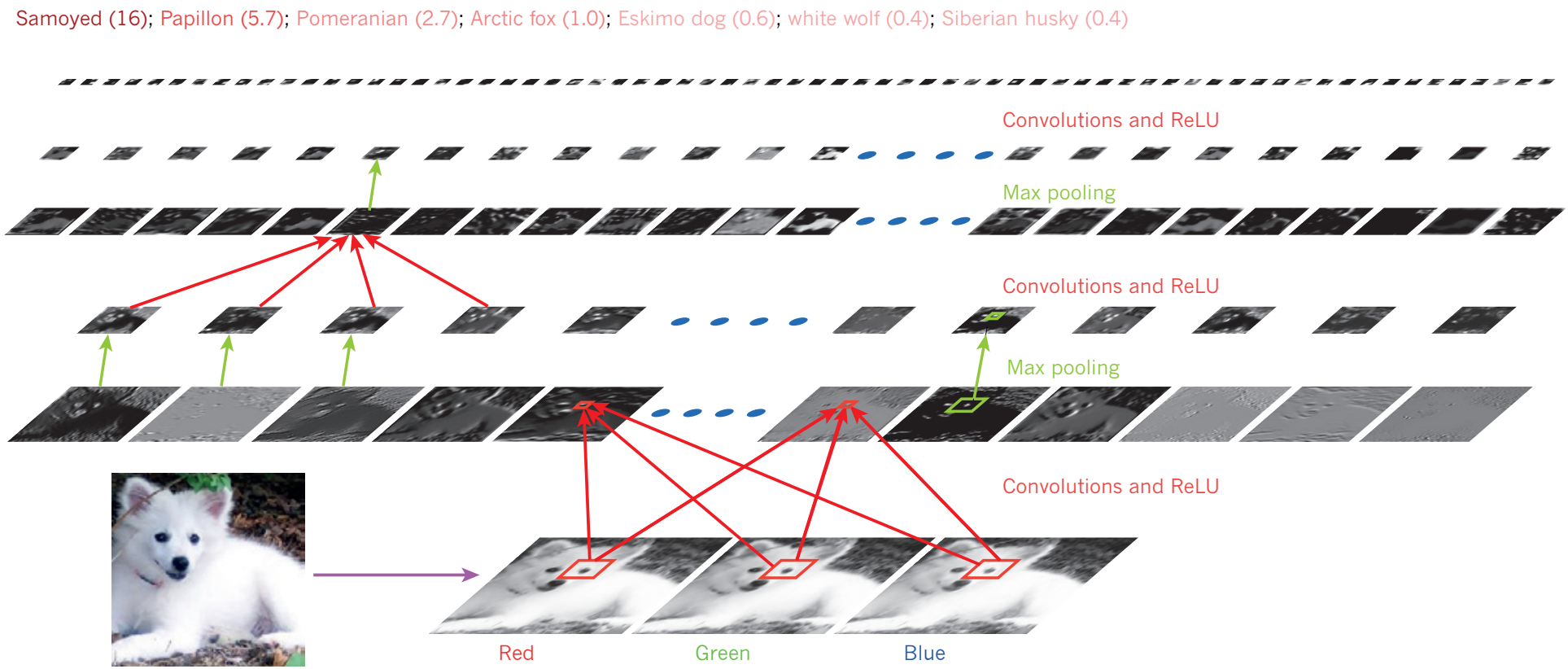
object recognition



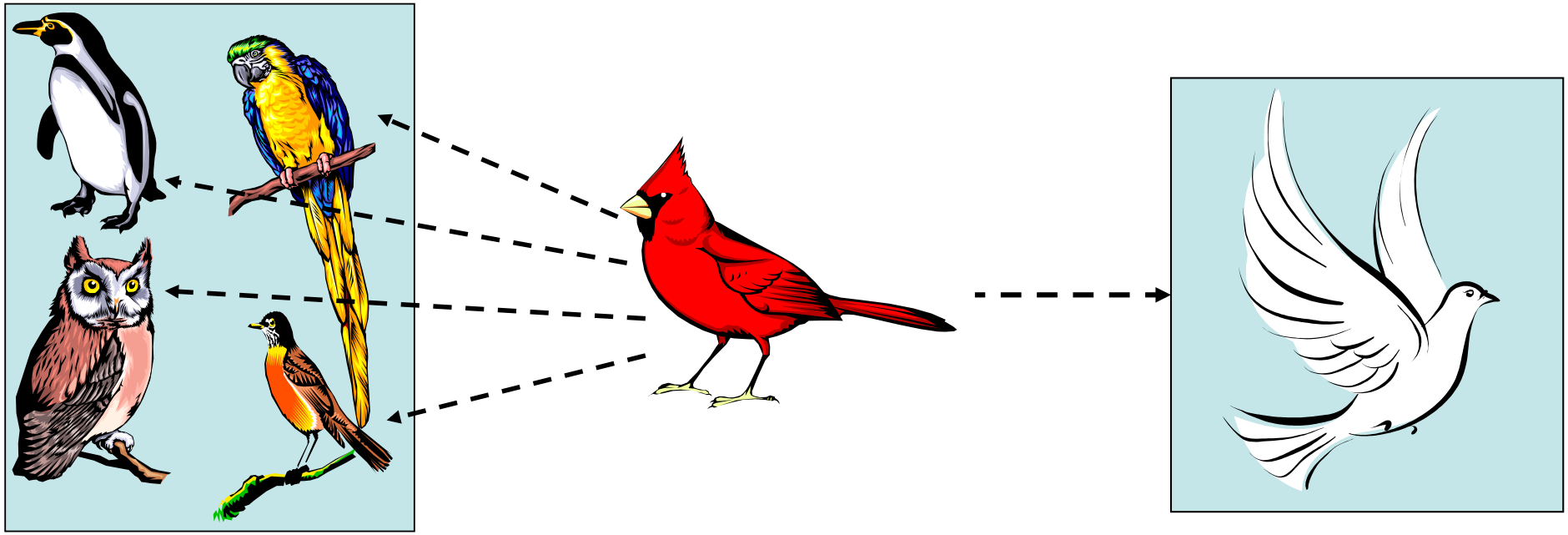
finding similar images

Is it a prototype model? Is it an exemplar model?

- well, it's not really either type of model. nor is it a mixture model like the rational model.
- It could choose to use its millions of parameters to store individual examples, abstract prototypes, or learn clusters



Review: Similar in what *respect*? What counts as a feature?



Birds
You've Seen

Bird?

Prototypical
Bird

- What counts as a feature? (Murphy & Medin, 1985)
  - To change the importance of age, we could include features for "around 10 years ago," "around 100 years ago," "1000 years ago", etc.
  - To change importance of size, we could include "smaller than the earth," "smaller than a country", "smaller than a city," etc.
- AlexNet doesn't seem to suffer from this philosophical issue — it learns the features it needs to solve the classification problem.

Typicality ratings are highly reliable and predict many aspects of categorization.

Category: furniture

Member	Rank	Score (1 [typical]... 7 [not typical])
chair	1.5	1.04
sofa	1.5	1.04
couch	3.5	1.10
table	3.5	1.10
easy chair	5	1.33
dresser	6.5	1.37
rocking chair	6.5	1.37
coffee table	8	1.38
rocker	9	1.42
love seat	10	1.44
...		
stove	50	5.40
counter	51	5.44
clock	52	5.48
drapes	53	5.67
refrigerator	54	5.70
picture	55	5.75
closet	56	5.95
vase	57	6.23
ashtray	58	6.35
fan	59	6.49
telephone	60	6.68

(Rosch, 1975)

Instructions:

...'Which breeds of dogs are real "doggy dogs"? To me, a German Shepard is very doggy, but a Pekinese is not. "Rate the extent to which each instance represents your idea of image of the meaning of the category"'

Goodness ratings are **highly consistent** across participant

Can you use ConvNets to predict category typicality?

Lake, B. M., Zaremba, W., Fergus, R., & Gureckis, T. M. (2015). Deep Neural Networks Predict Category Typicality Ratings for Images.

typical dog

atypical dog

• For people, typicality influences performance in practically any category-related task

- speed of categorization
- ease of production
- ease of learning
- usefulness for inductive inference

• No known model successfully predicts typicality ratings from raw images -- How do convnets perform?

Category: Banana (ρ=0.82)

How well does this picture fit your idea or image of the category? (rated on 1-7 scale)

Human typicality ratings

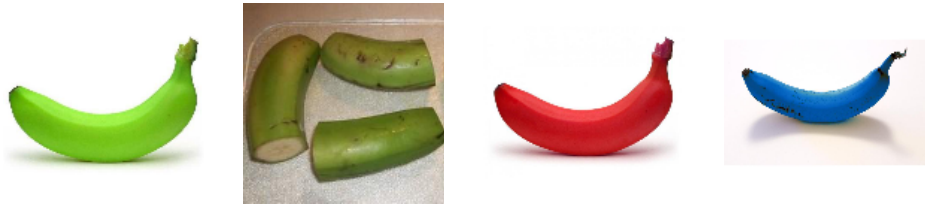
Most typical →

[97.8, 6.8] [98.0, 6.8] [96.6, 6.8] [99.7, 6.6]

[96.9, 6.6] [99.3, 6.0] [78.6, 5.8] [99.5, 5.5]

[12.1, 5.3] [59.7, 4.4] [2.9, 4.3] [46.1, 4.1]

[14.0, 4.1] [0.2, 3.6] [2.3, 2.5] [1.3, 2.4]



Least typical

rating key: [machine (0-100), human (1-7)]

Category: Bathtub (ρ=0.68)

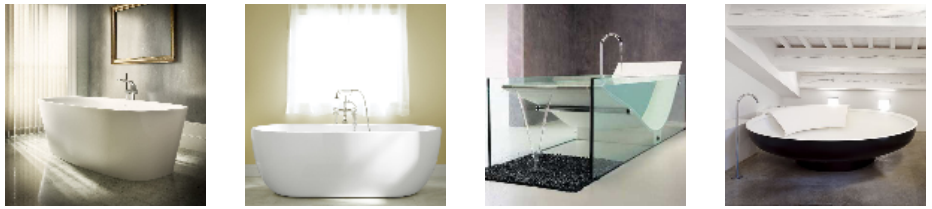
Human typicality ratings

Most typical →

[60.6, 6.6] [58.5, 6.6] [57.3, 6.6] [66.5, 6.2]

[72.0, 6.1] [80.7, 6.0] [9.5, 5.9] [35.4, 5.7]

[67.6, 5.6] [63.0, 5.2] [9.8, 3.2] [16.4, 3.1]



[1.0, 3.0] [1.5, 2.9] [1.0, 2.8] [9.1, 2.4]

Least typical

Convnet typicality ratings

[80.7, 6.0] [72.0, 6.1] [67.6, 5.6] [66.5, 6.2]

[63.0, 5.2] [60.6, 6.6] [58.5, 6.6] [57.3, 6.6]

[35.4, 5.7] [16.4, 3.1] [9.8, 3.2] [9.5, 5.9]

[9.1, 2.4] [1.5, 2.9] [1.0, 2.8] [1.0, 3.0]

rating key: [machine (0-100), human (1-7)]

Category: Envelope (ρ=0.79)

Human typicality ratings

Most typical →

Least typical

Convnet typicality ratings

rating key: [machine (0-100), human (1-7)]

Category: Teapots (ρ=0.38)

Human typicality ratings

Most typical →

Least typical

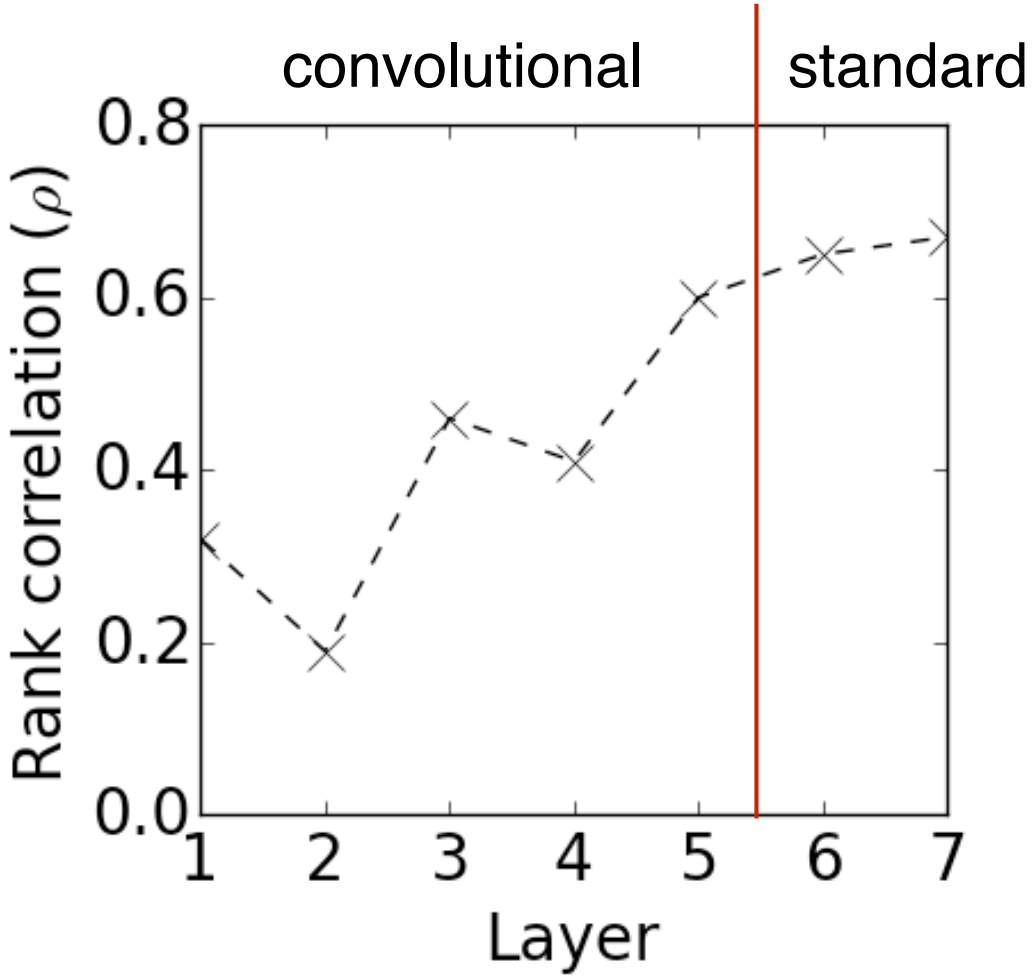
Convnet typicality ratings

rating key: [machine (0-100), human (1-7)]

Summary of typicality predictions

	Rank Correlation
Banana	0.82
Bathtub	0.68
Coffee Mug	0.62
Envelope	0.79
Pillow	0.67
Soap dispenser	0.74
Table lamp	0.69
Teapot	0.38
Average	**0.67**

Prediction quality varies as a function of network depth.



Can a "Deep-ALCOVE" predict SHJ results?

Modeling artificial category learning from pixels: Revisiting Shepard, Hovland, and Jenkins (1961) with deep neural networks

Alexa R. Tartaglini (art481@nyu.edu)
Wai Keen Vong (waikeen.vong@nyu.edu)
Brenden M. Lake (brenden@nyu.edu)
Center for Data Science, 60 5th Ave,
New York University, NY, 10011

Abstract

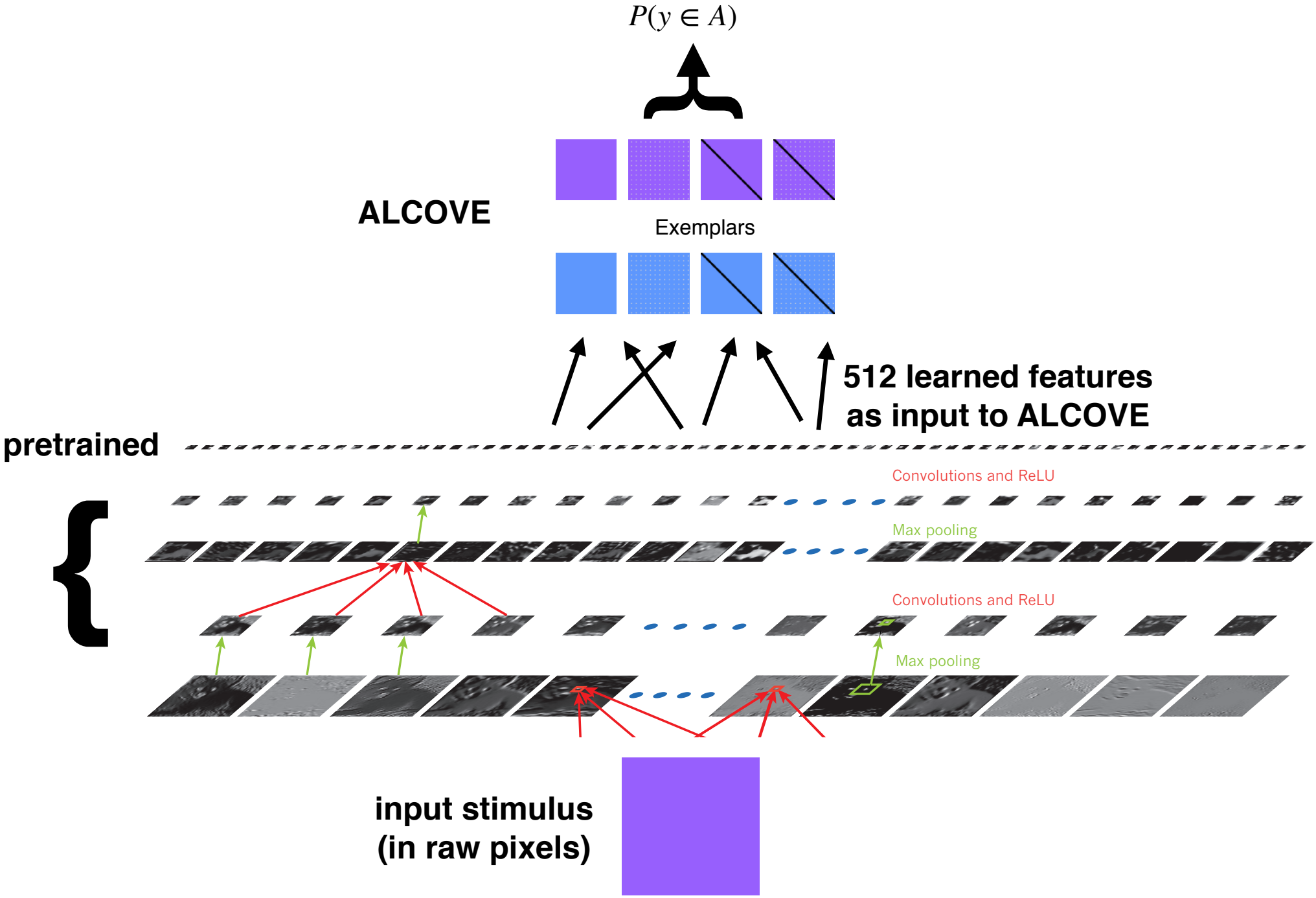
Recent work has paired classic category learning models with convolutional neural networks (CNNs), allowing researchers to study categorization behavior from raw image inputs. However, this research typically uses naturalistic images, which assess participant responses to existing categories; yet, much of traditional category learning research has focused on using novel, artificial stimuli to examine the learning process behind how people acquire categories. In this work, we pair a CNN with ALCOVE (Kruschke, 1992), a well-known exemplar model of categorization, and attempt to examine whether this model can reproduce the classic type ordering effect from Shepard, Hovland, and Jenkins (1961) on raw images rather than abstract features. We examine this question with a variety of CNN architectures and image datasets and compare ALCOVE-CNN to two other models that lacked certain key features of ALCOVE. We found that our ALCOVE-CNN model could reproduce the type ordering effect more often than the other models we tested, but in limited situations. Our results showed that success varied greatly across the various configurations we tested, suggesting that the feature representations from CNNs provide strong constraints in properly capturing this effect.

Keywords: category learning; convolutional neural networks; exemplar models; attention

and categorization. For instance, CNNs pre-trained for image classification have shown success in predicting category typicality ratings (Lake, Zaremba, Fergus, & Gureckis, 2015) and similarity ratings from natural images (Peterson, Abbott, & Griffiths, 2018). More recently, researchers have begun to combine CNNs with classic prototype and exemplar models of categorization (Battleday, Peterson, & Griffiths, 2020; Guest & Love, 2019; Singh, Peterson, Battleday, & Griffiths, 2020; Nosofsky, Meagher, & Kumar, 2020), usually with the aim of predicting human categorization decisions for images of common categories such as animals and vehicles.

These successes in predicting human judgments on natural images do not, however, imply that such an approach will be a successful psychological model of categorization as studied in its most classic setting: artificial category learning tasks conducted in the lab. This is the question we take up in our work here. One of the most influential findings in this vein is the classic experiment by Shepard et al. (1961) which has been replicated many times (Nosofsky, Gluck, Palmeri, McKinley, & Glauthier, 1994; Crump, McDonnell, & Gureckis, 2013). Shepard et al. (1961) showed that with

Can a “Deep-ALCOVE” predict SHJ results?



Example ConvNet results on raw SHJ stimuli

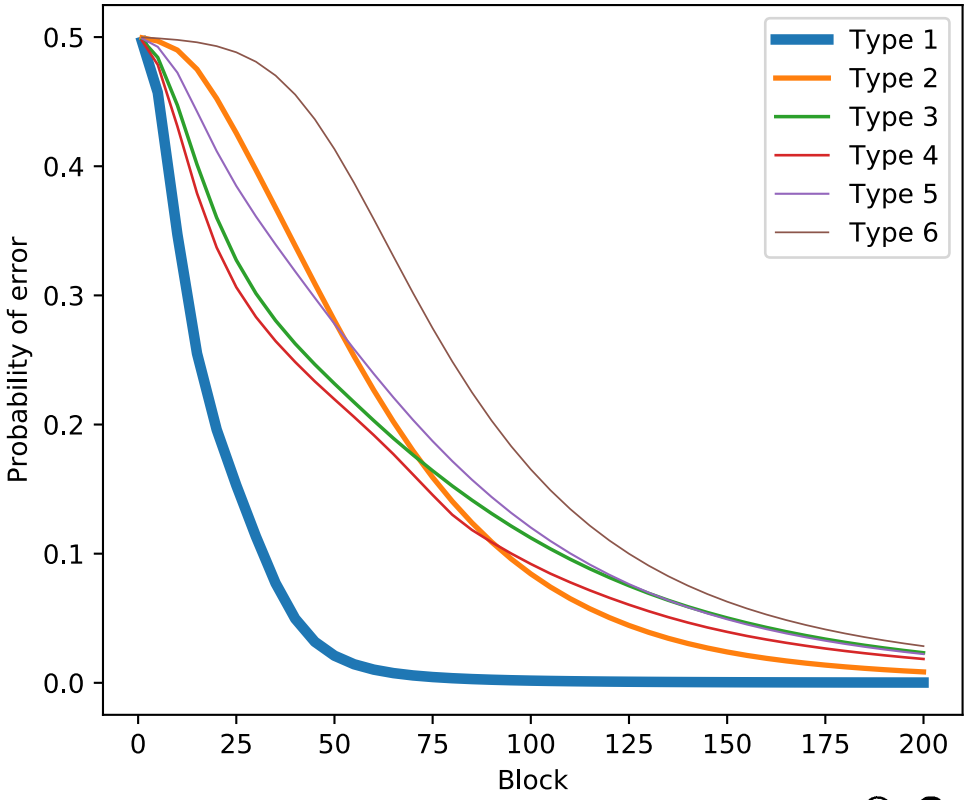
Deep ConvNet + ALCOVE

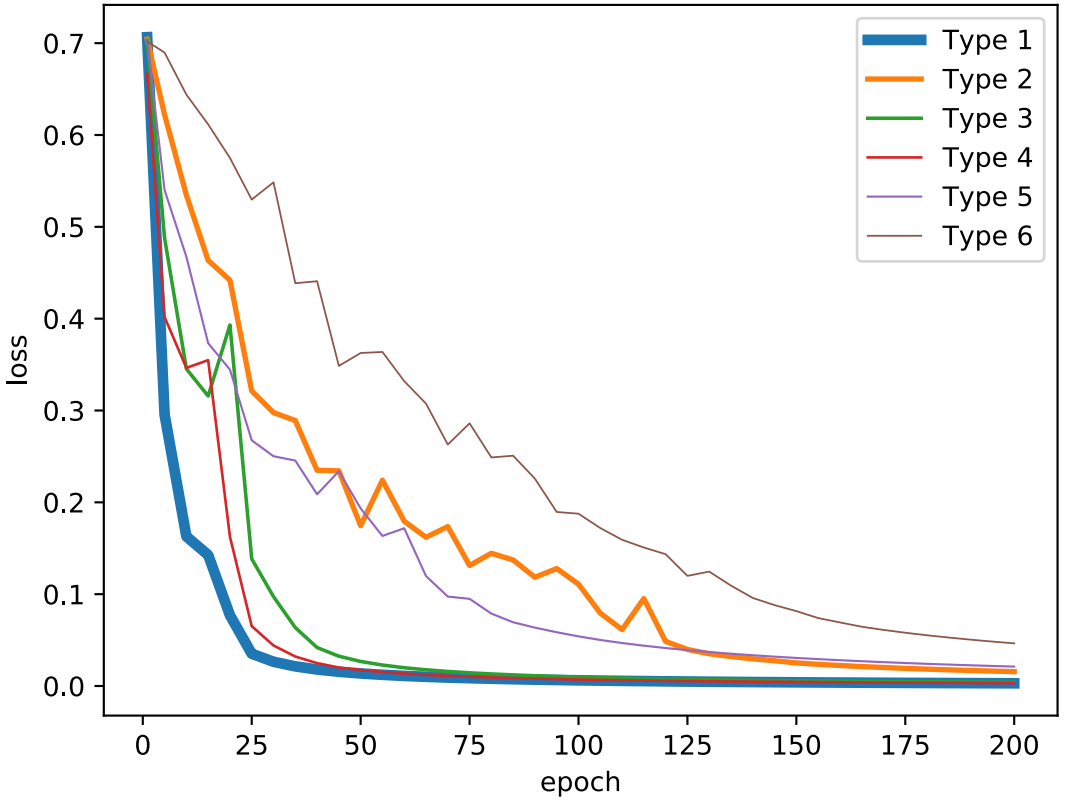
Deep ConvNet + standard neural net

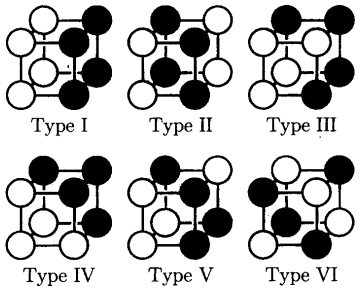
(gets Type II order wrong)

(gets Type III order wrong)

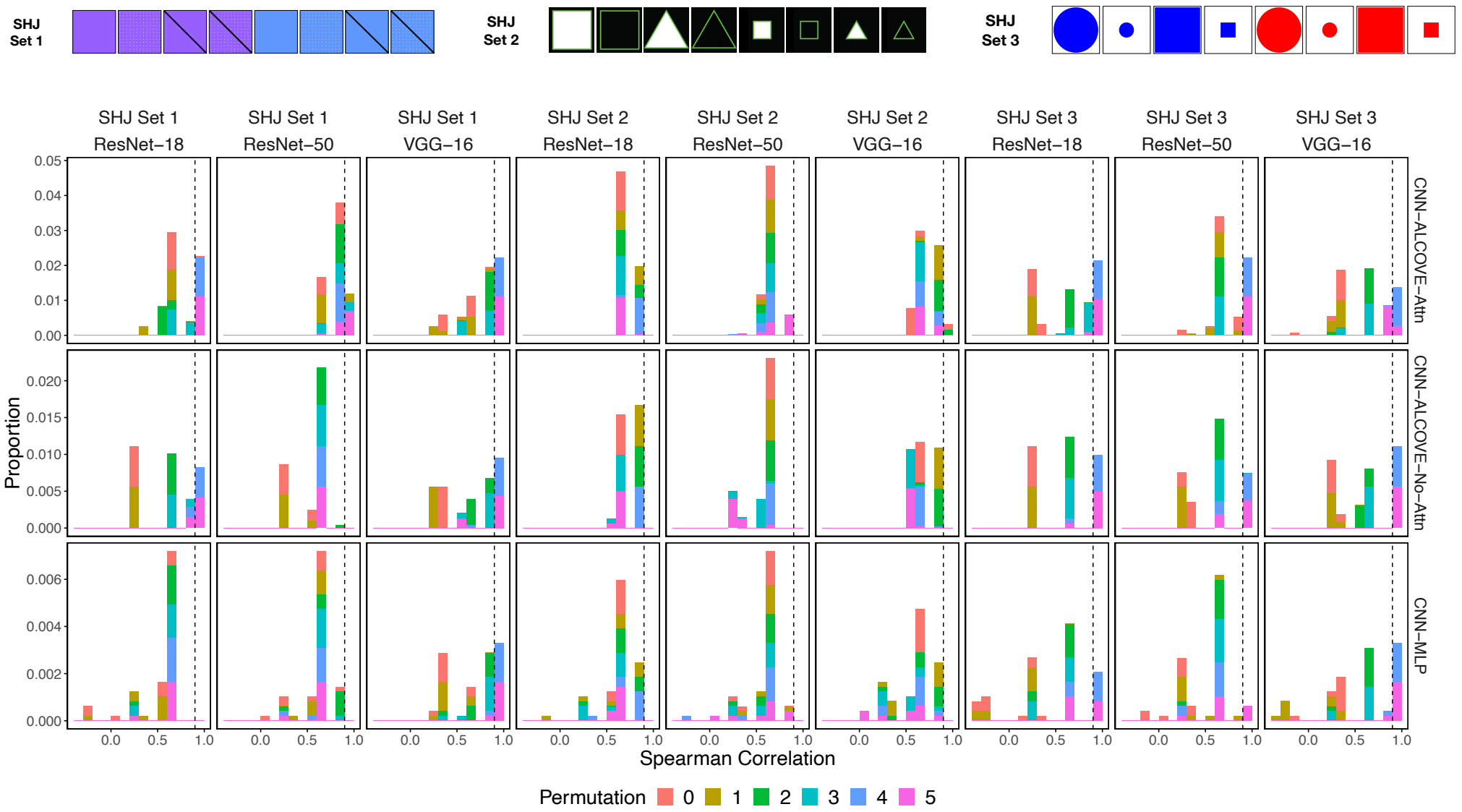
(gets Type IV order wrong)







But Deep-ALCOVE does not robustly predict the right ordering across different key factors



Some conclusions

- We still don't know exactly what these recent advances in machine learning mean for the cognitive science of concepts
- ConvNets — the dominant algorithm for object classification from images — don't fit neatly into current psychological theories of concepts
- Alexa's results point to difficulties in applying ConvNets out of the box to interface with raw experimental stimuli
- My view is that interactions between these two fields are critical for advancing both