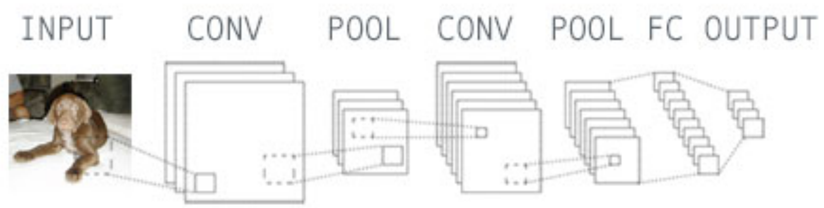
BSCS 2019 - Neural Computation

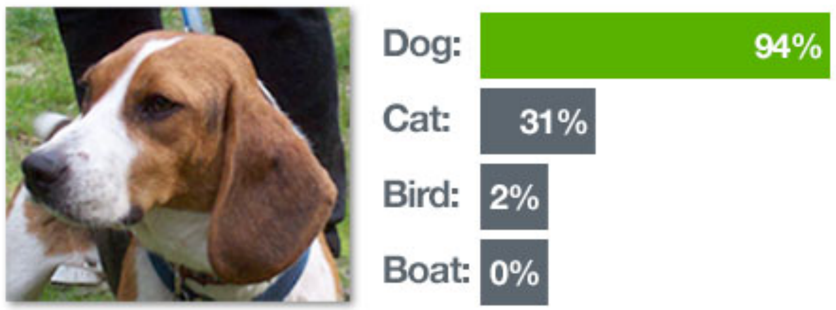
VII - Topics from the intersection of AI and neuroscience

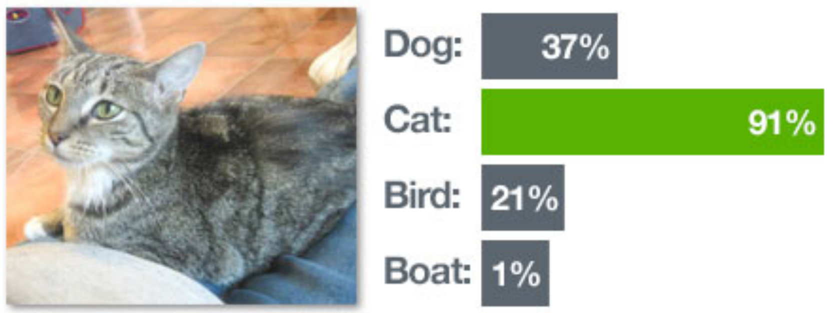
Mihály Bányai
banyai.mihaly@wigner.mta.hu
http://golab.wigner.mta.hu/people/mihaly-banyai/

AI today

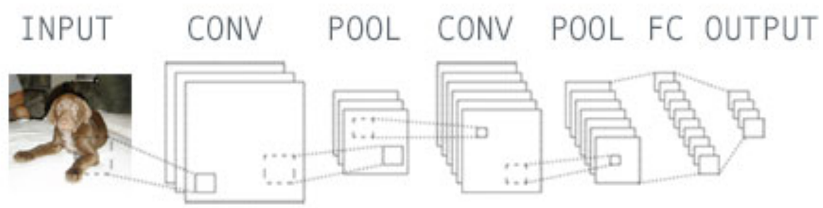
AI today

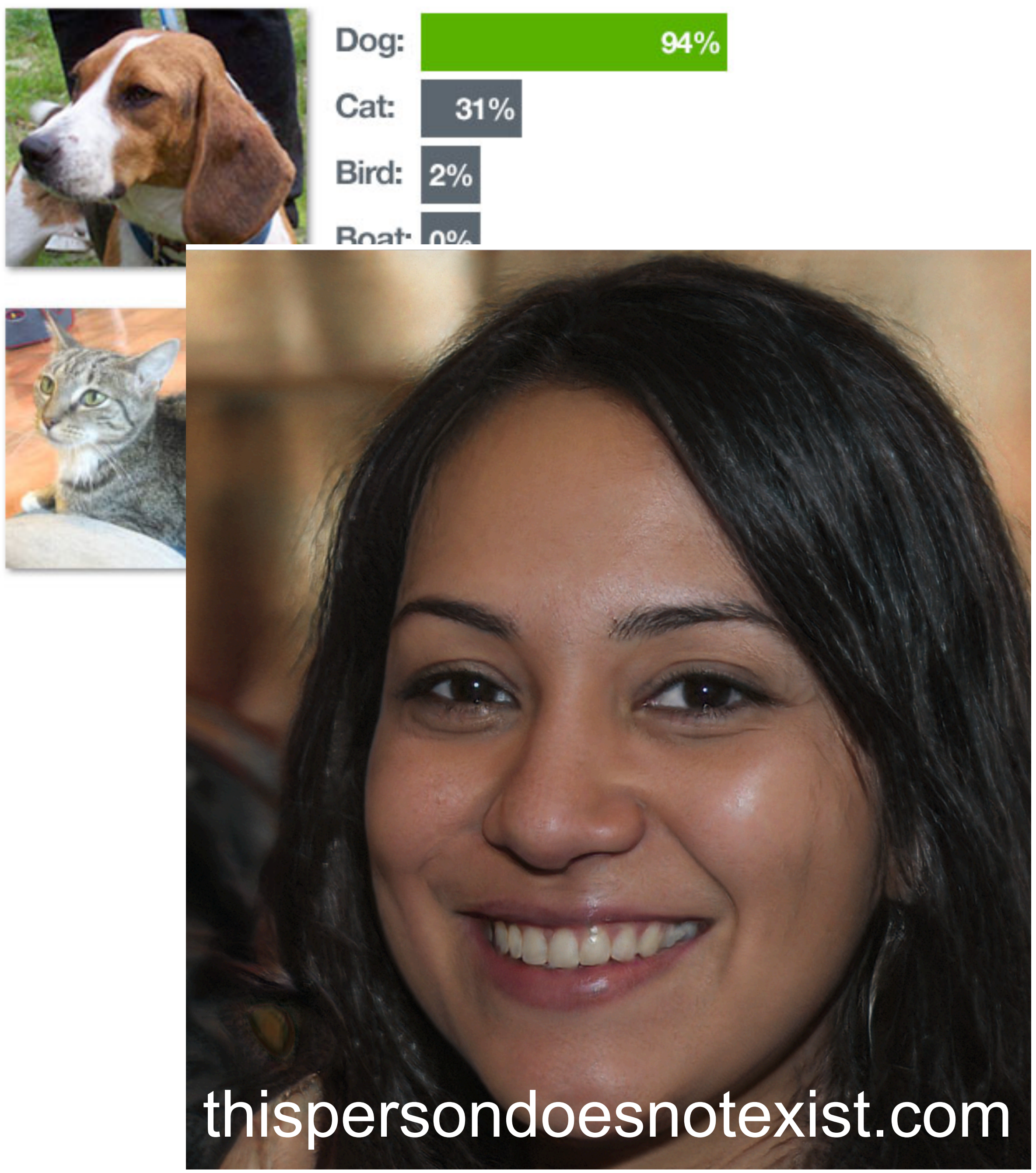

INPUT
CONV
POOL
CONV
POOL
FC
OUTPUT


Dog: 94%
Cat: 31%
Bird: 2%
Boat: 0%

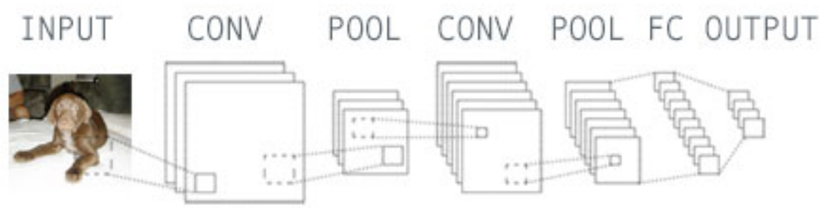

Dog: 37%
Cat: 91%
Bird: 21%
Boat: 1%

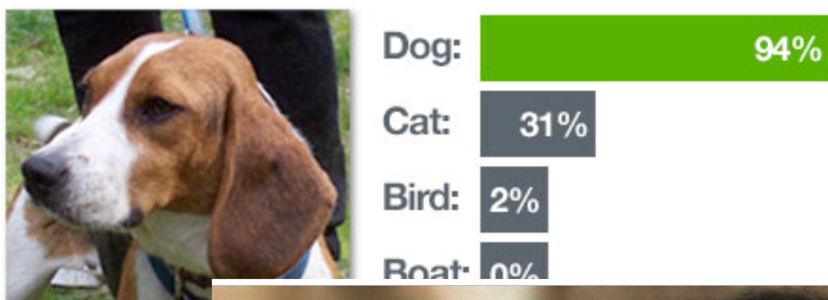
AI today

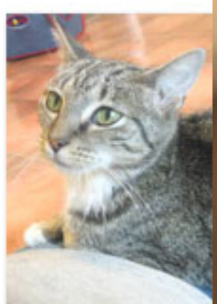

INPUT
CONV
POOL
CONV
POOL
FC
OUTPUT


Dog: 94%
Cat: 31%
Bird: 2%
thispersondoesnotexist.com

AI today


INPUT
CONV
POOL
CONV
POOL
FC
OUTPUT

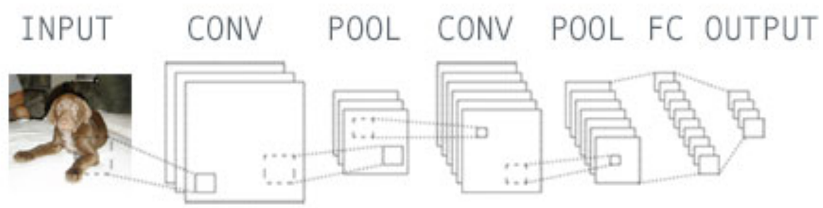

Dog: 94%
Cat: 31%
Bird: 2%

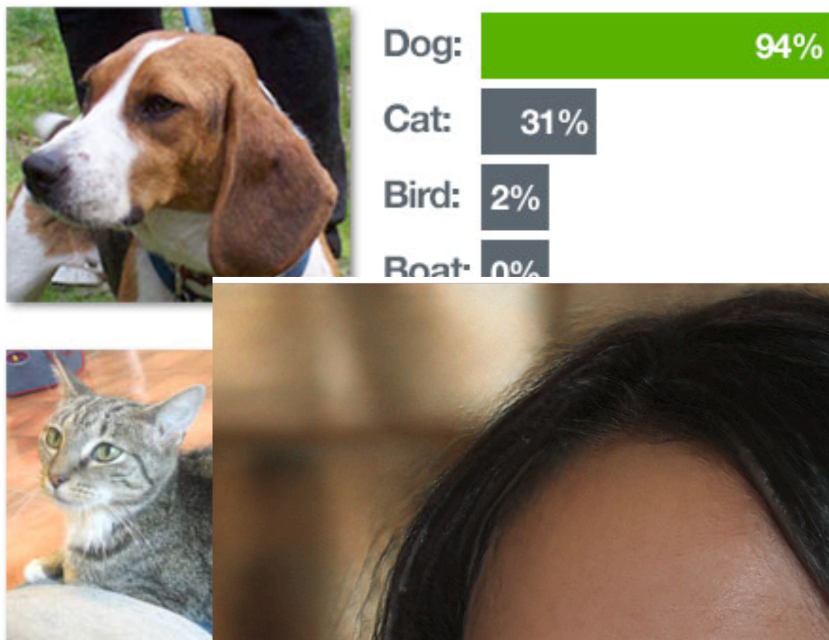



thispersondoesnotexist.com


AlphaGo
Lee Sedol
01:58:30

AI today


INPUT
CONV
POOL
CONV
POOL
FC
OUTPUT

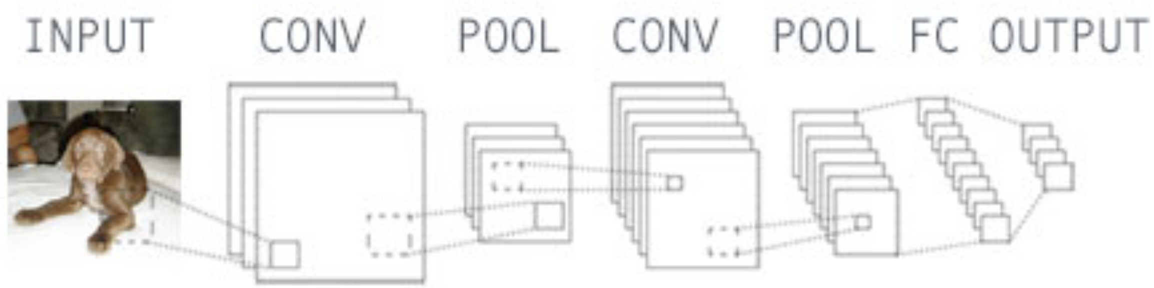

Dog: 94%
Cat: 31%
Bird: 2%

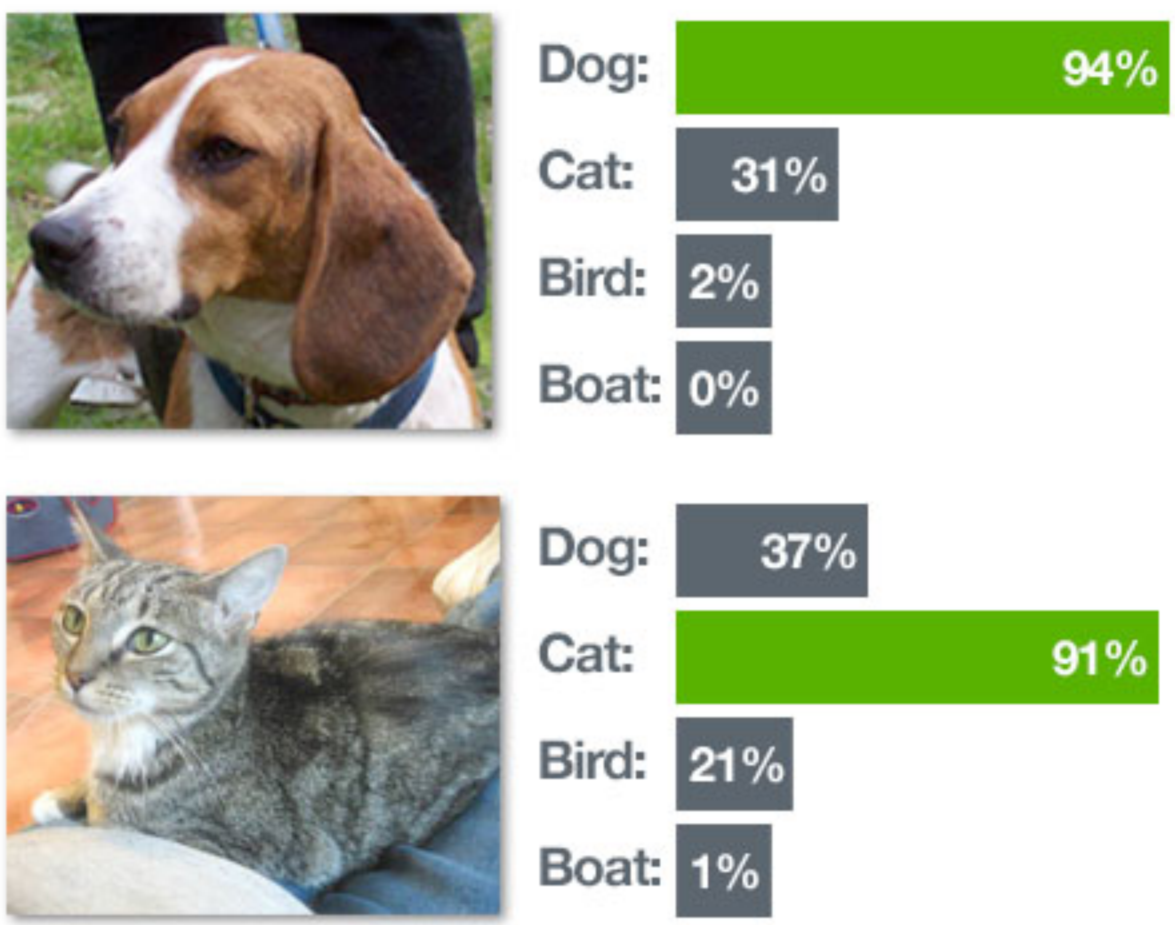

thispersondoesnotexist.com


ALPHAGO 01:58:30
AlphaGo
Lee Sedol

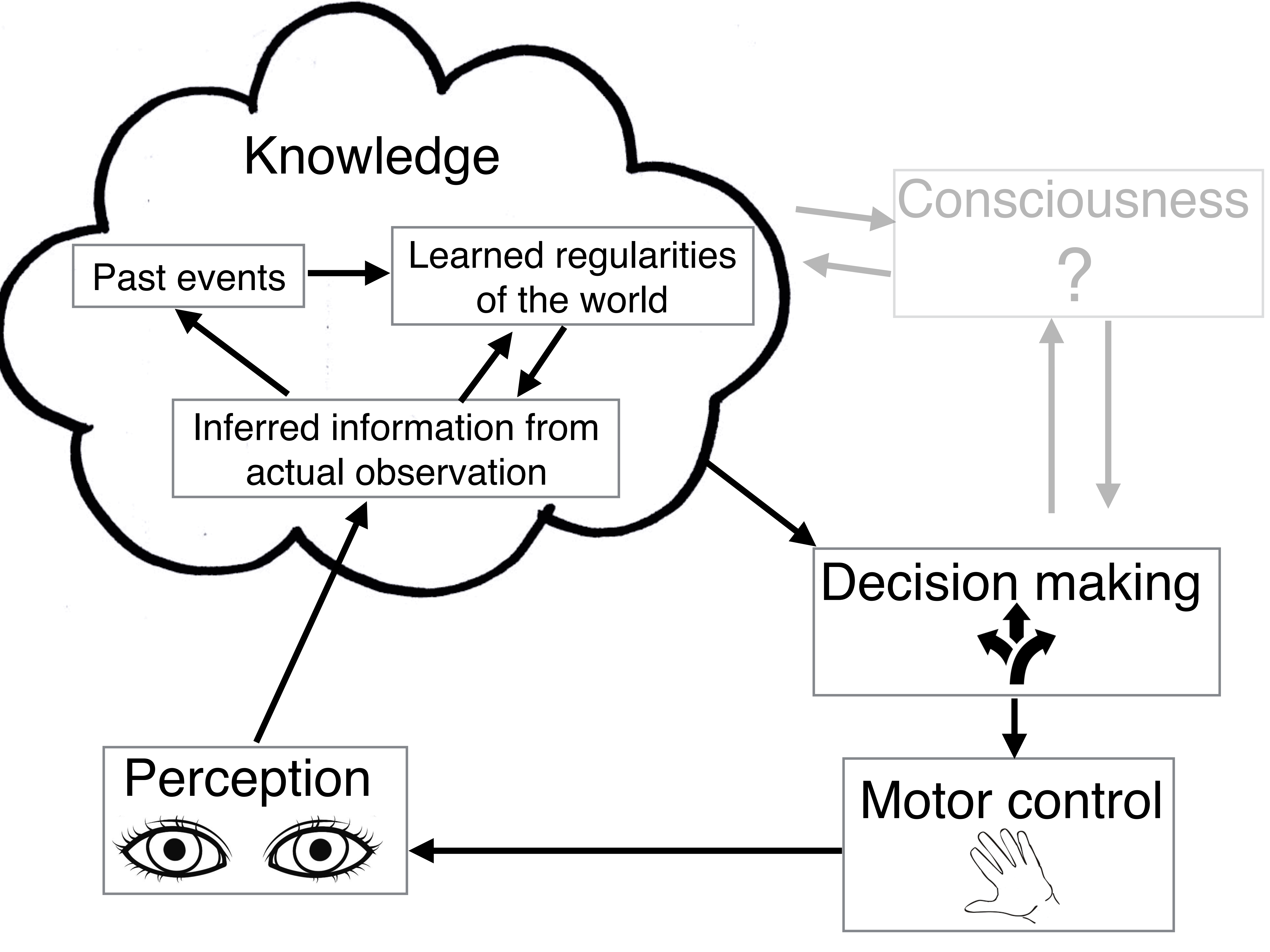

Viper

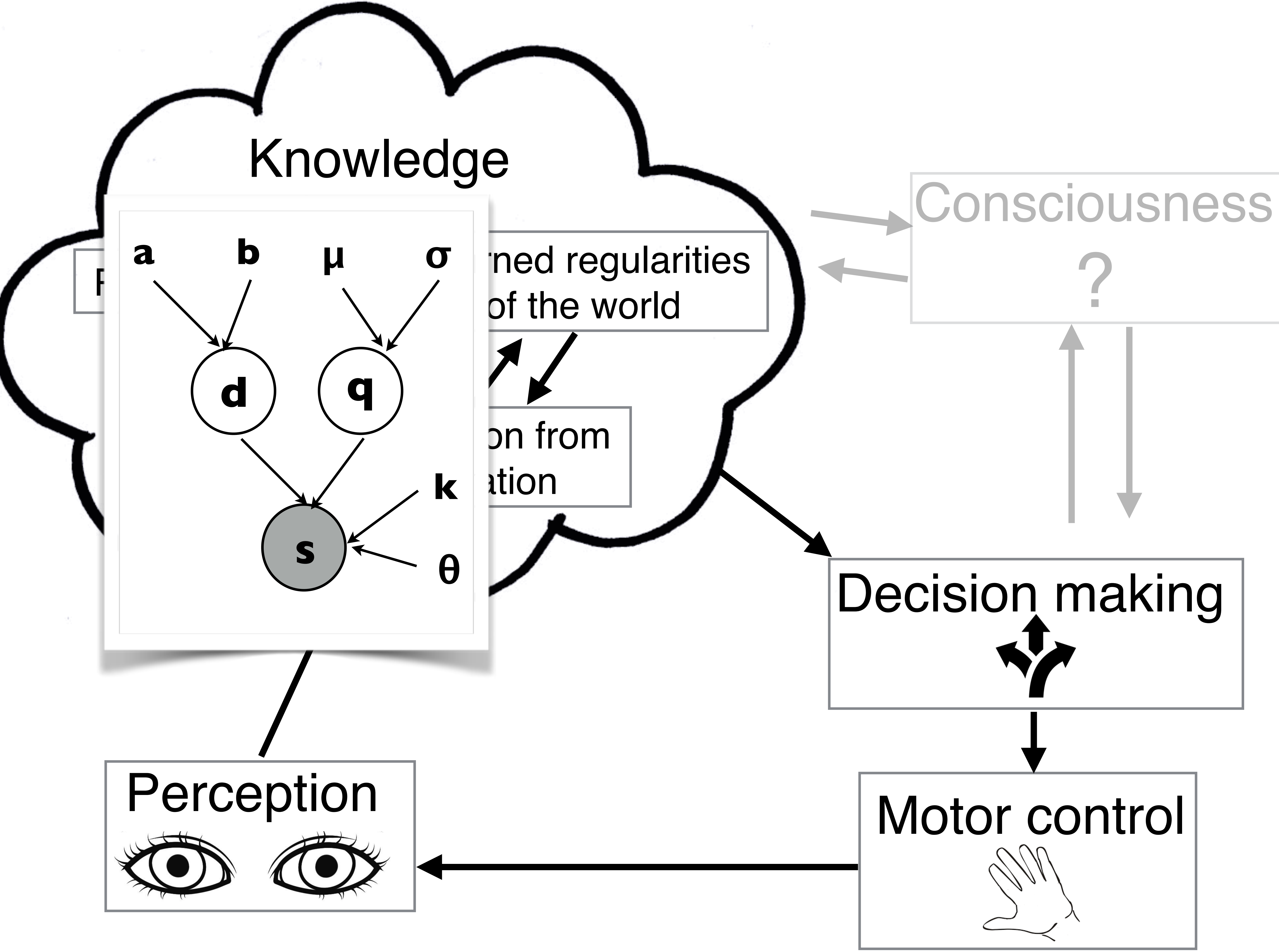
AI today

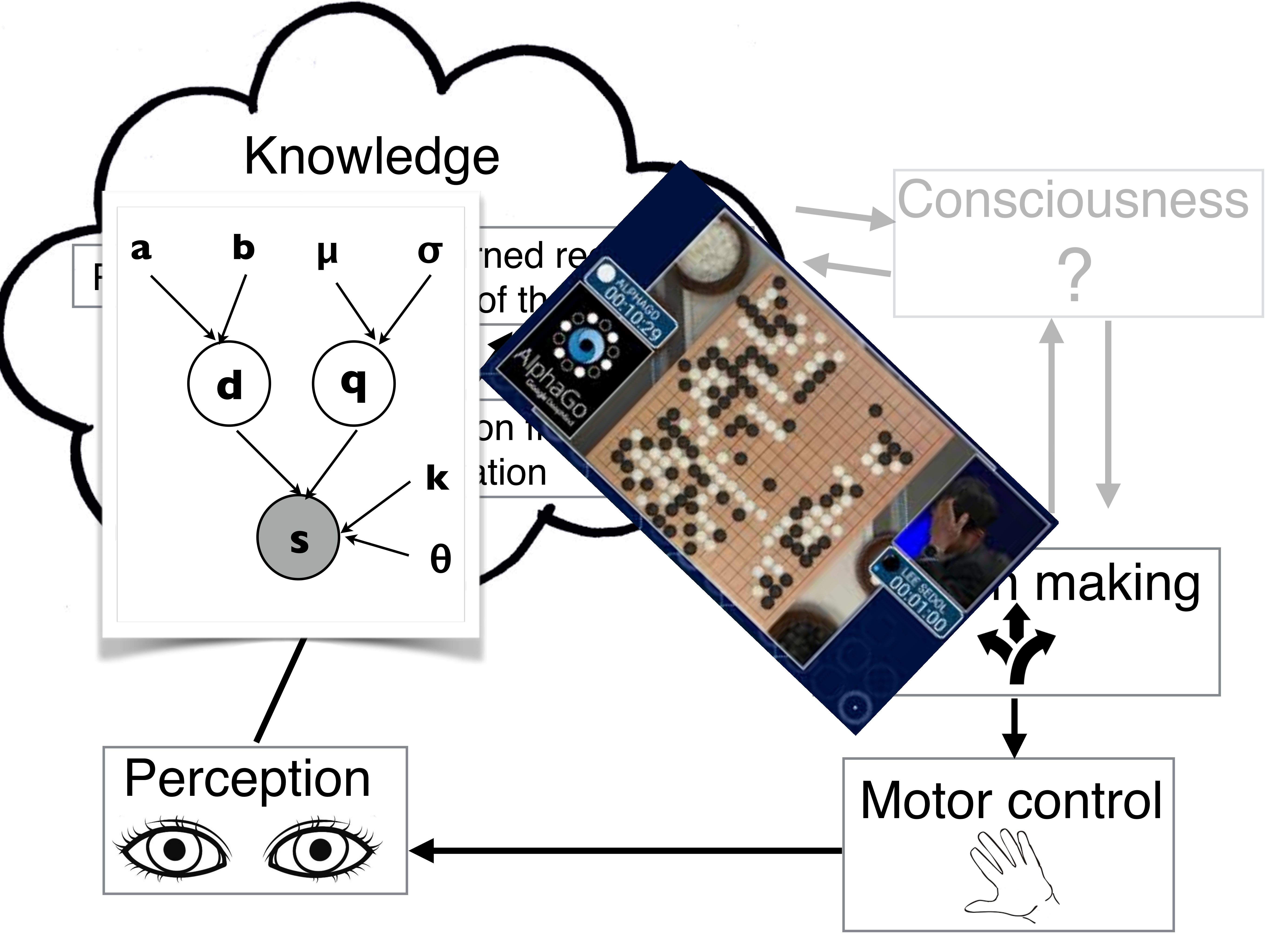

INPUT
CONV
POOL
CONV
POOL
FC
OUTPUT

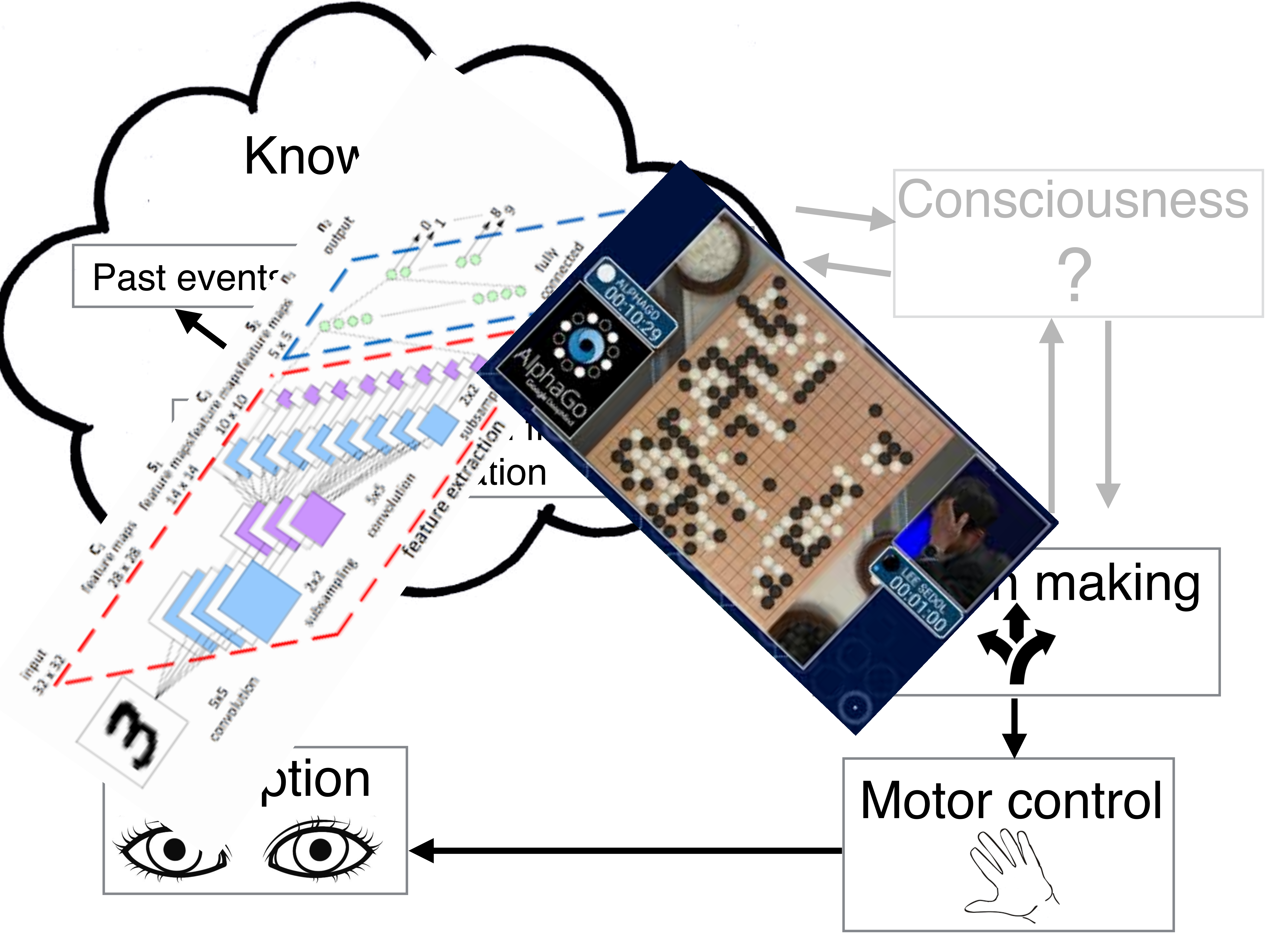

Dog: 94%
Cat: 31%
Bird: 2%
Boat: 0%
Dog: 37%
Cat: 91%
Bird: 21%
Boat: 1%


AlphaGo
Lee Sedol


Knowledge
Past events
Learned regularities of the world
Inferred information from actual observation
Consciousness
?
Decision making
Perception
Motor control


Knowledge
a
b
μ
σ
d
q
k
s
θ
rned regularities
of the world
on from
ation
Consciousness
?
Decision making
Perception
Motor control

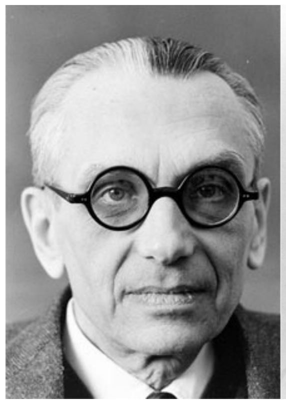
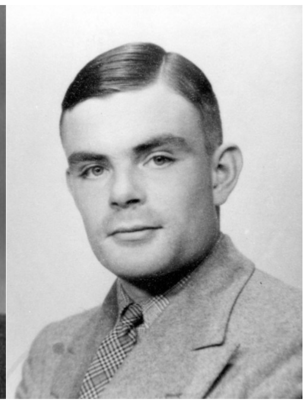

Knowledge
a
b
μ
σ
d
q
k
s
θ
rned re
of th
on
ation
ALPHAGO
00:10:29
AlphaGo
LEE SEDOL
00:01:00
Consciousness
?
n making
Perception
Motor control



Know
Past events
Consciousness
?
tion
n making
ption
Motor control
feature extraction
AlphaGo
ALPHAGO 00:10:29
LEE SEDOL 00:01:00

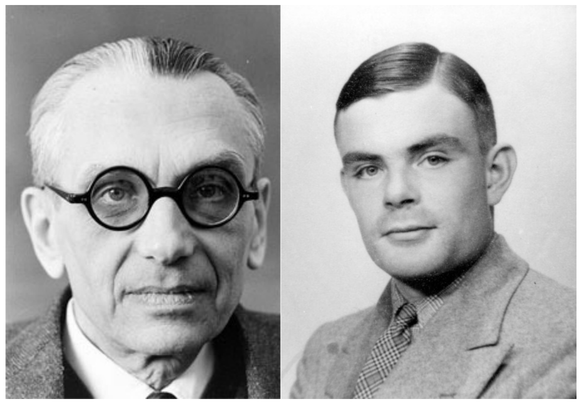
The beginning of AI

results in logic

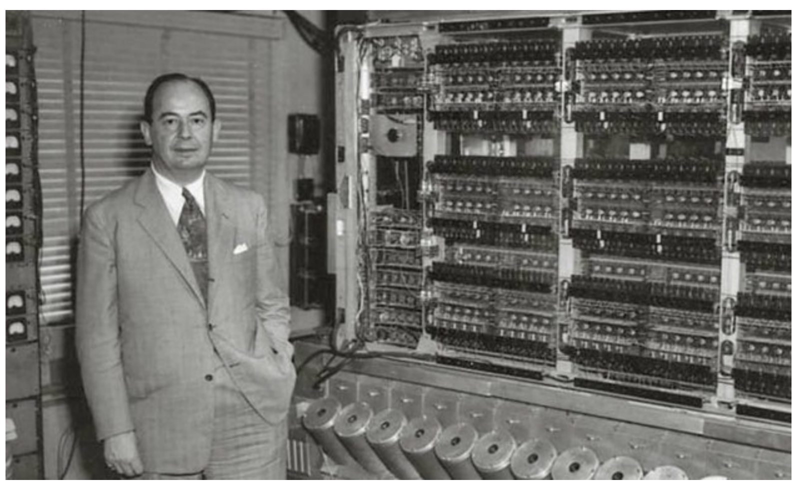


The beginning of AI

results in logic

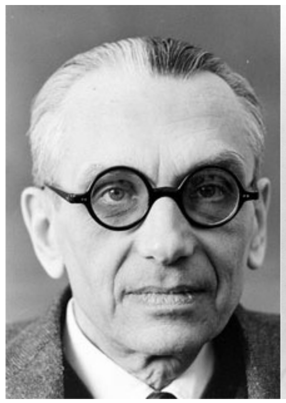
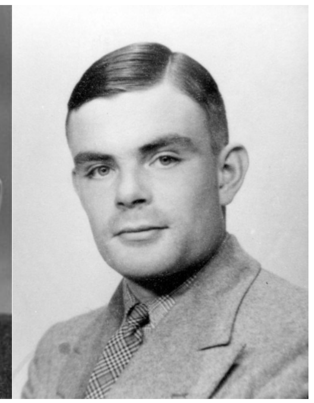


digital computer

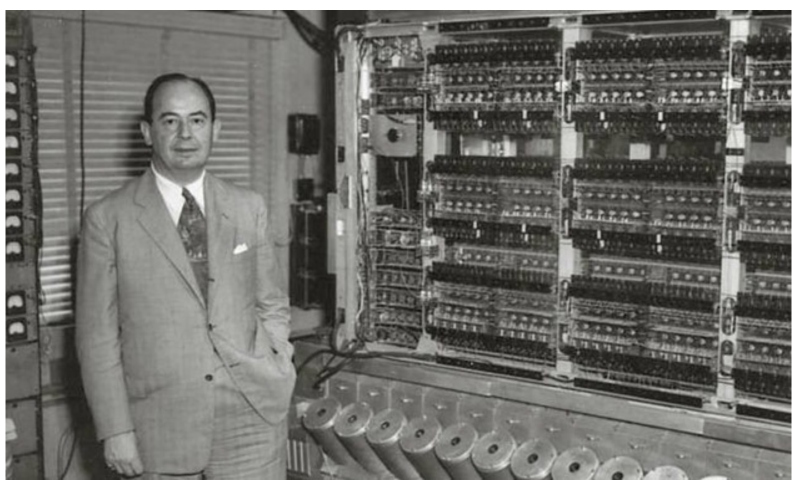


The beginning of AI

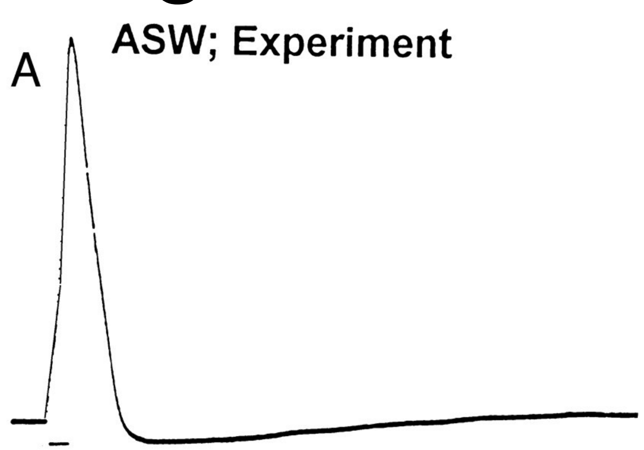
results in logic

digital computer

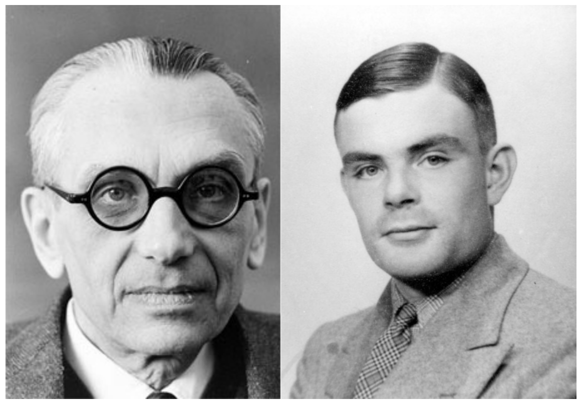


neural recordings

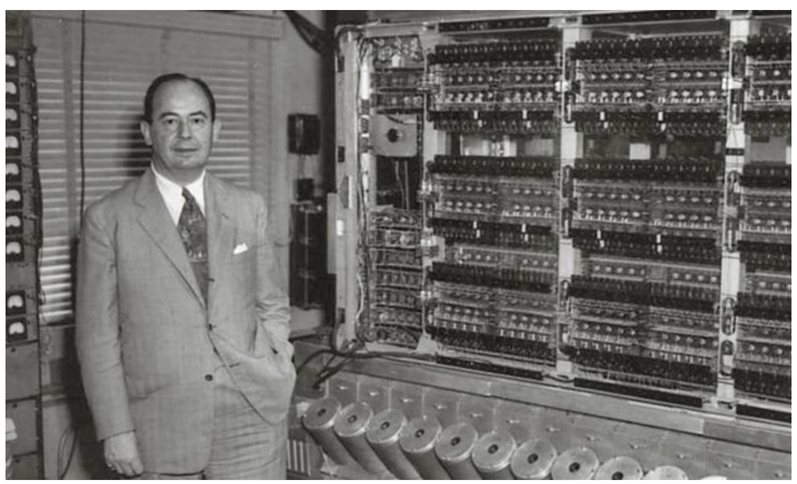



The beginning of AI

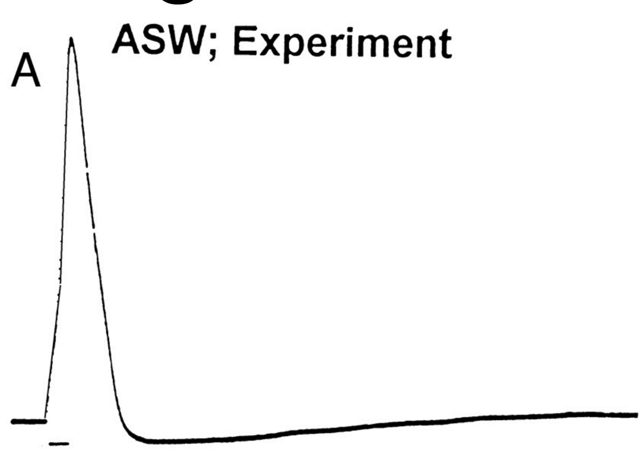
results in logic



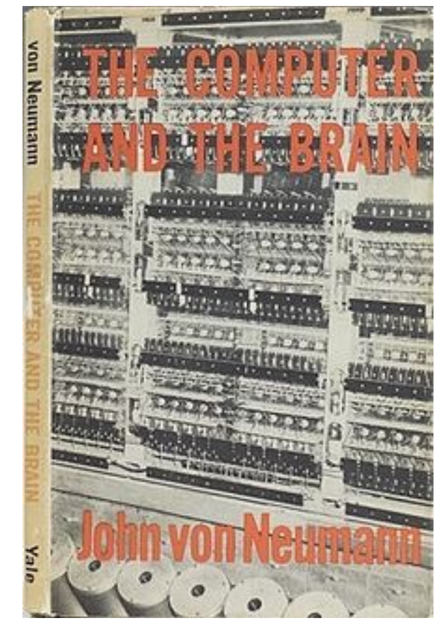
digital computer



neural recordings

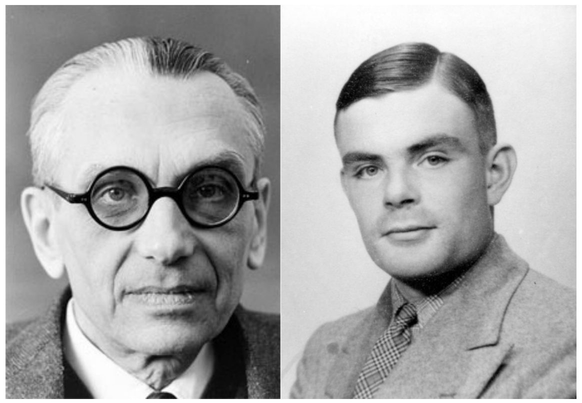


cybernetics

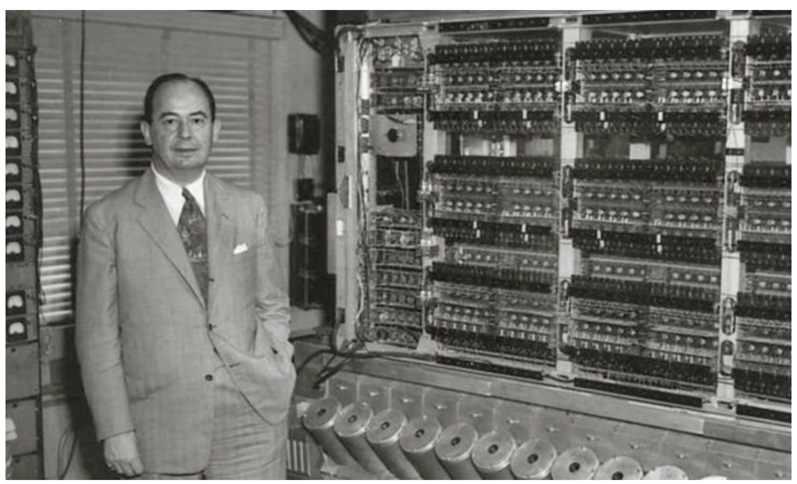


The beginning of AI

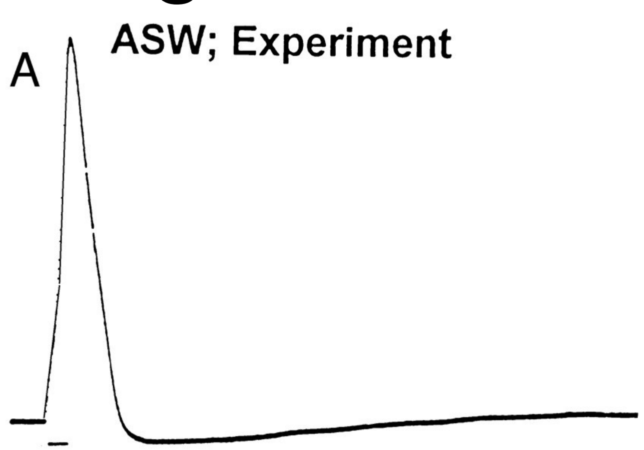
results in logic



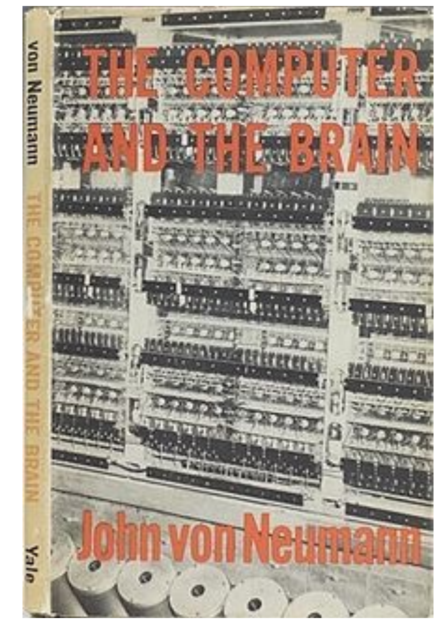
digital computer

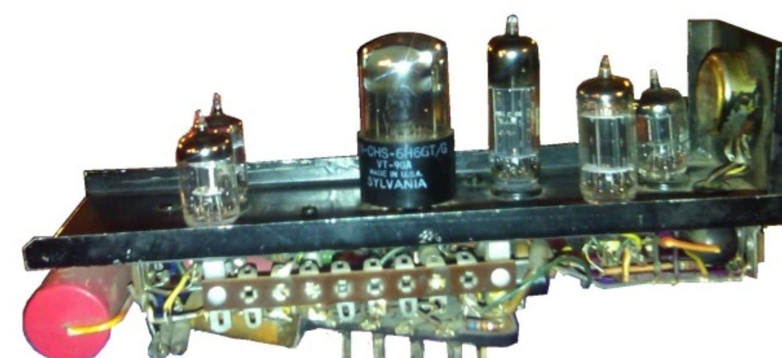


neural recordings



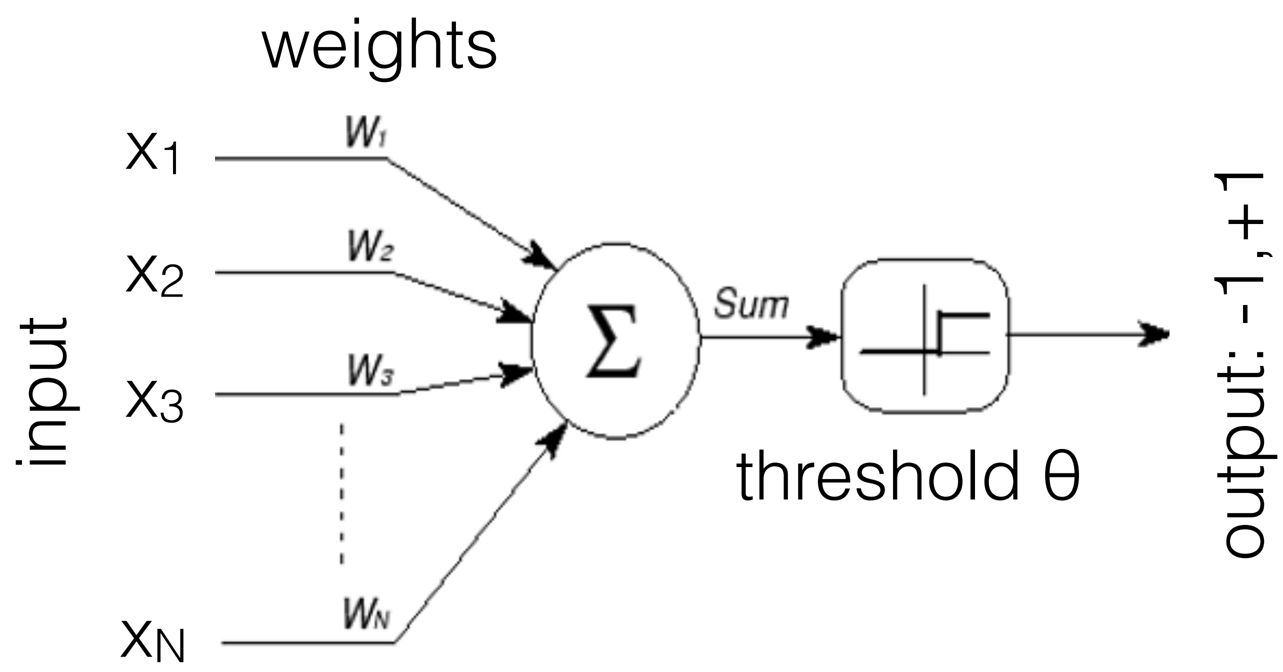
cybernetics





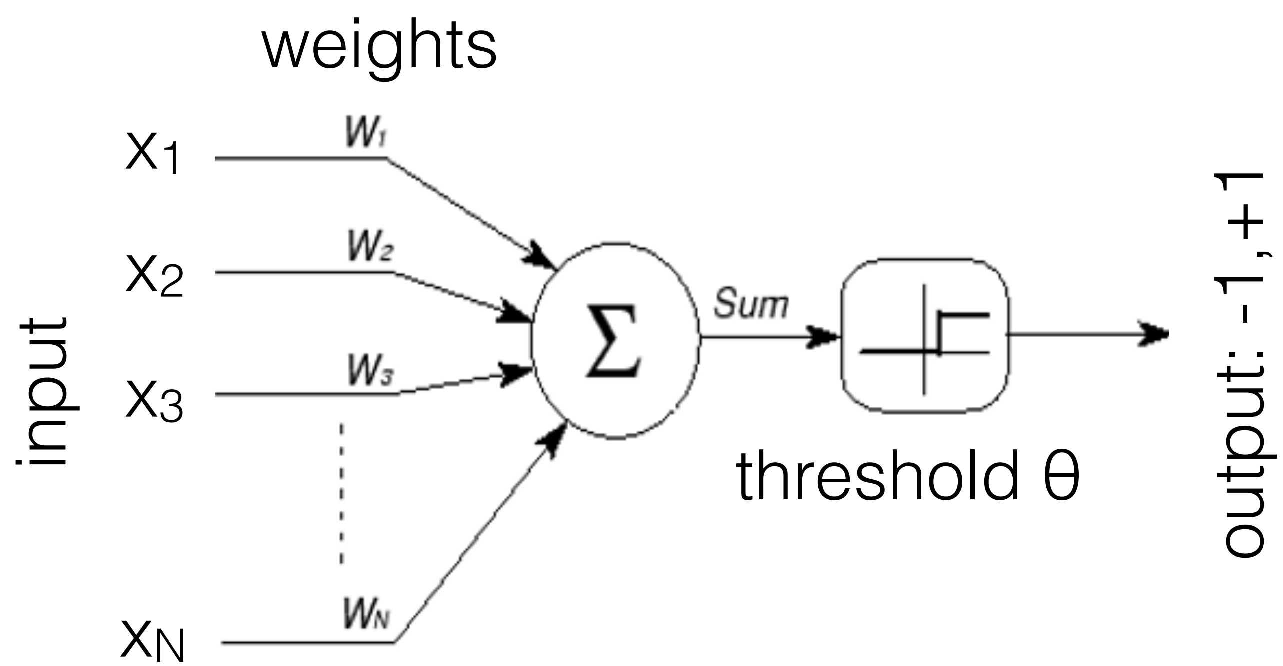
artificial neuron

Artificial neuron



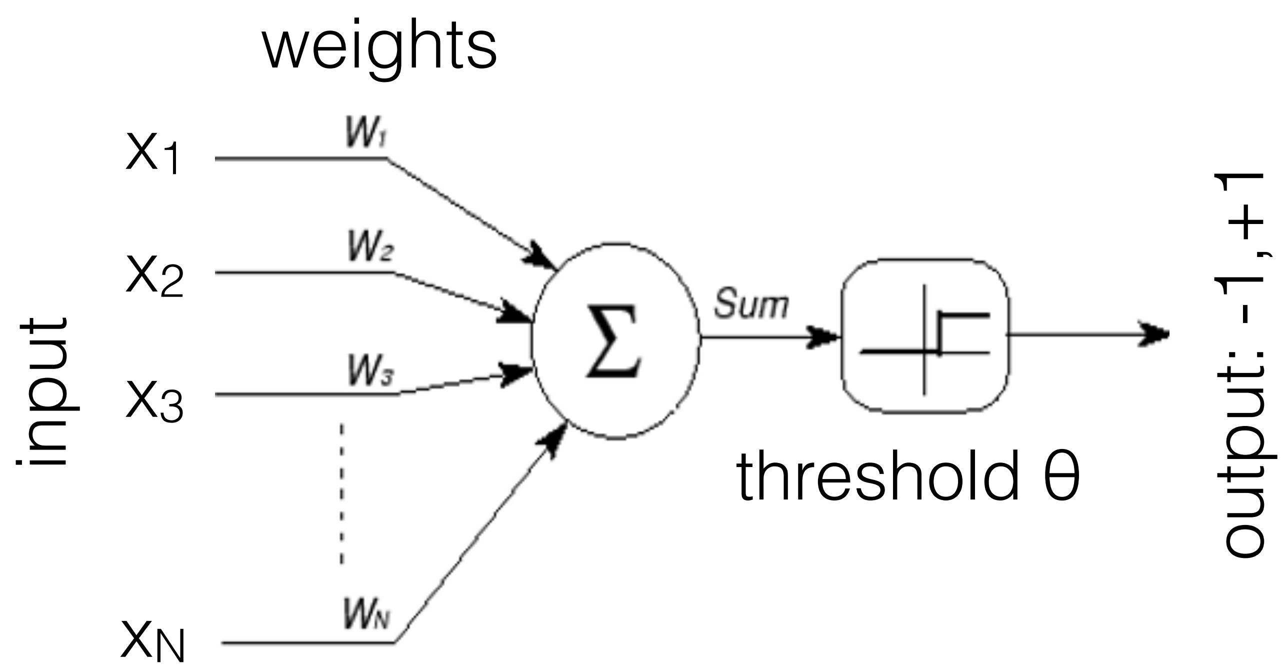
- we compare the weighted sum of inputs to a threshold

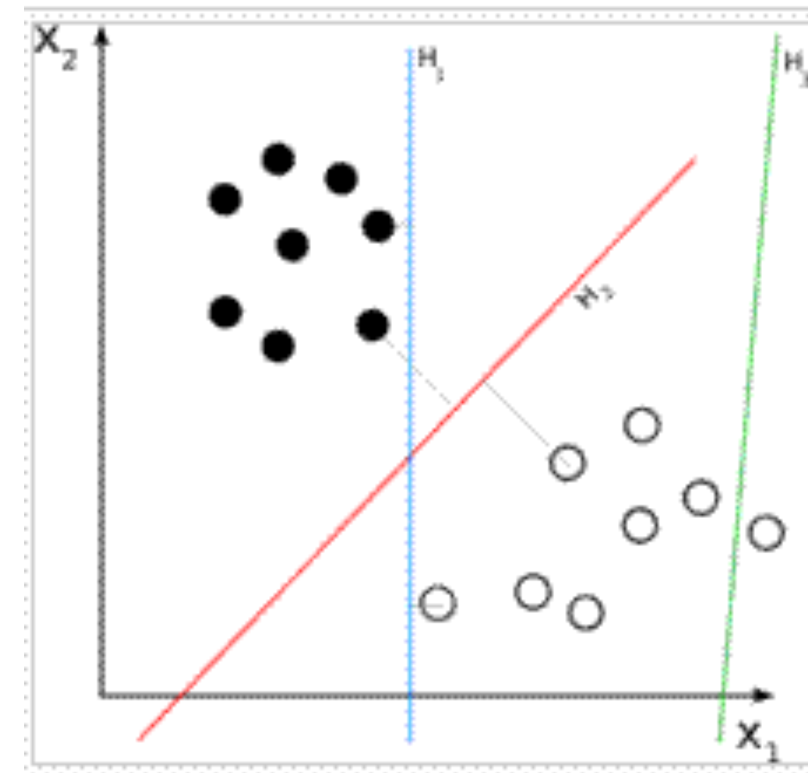
Artificial neuron



- we compare the weighted sum of inputs to a threshold
- two possible output values - a binary classification of input combinations

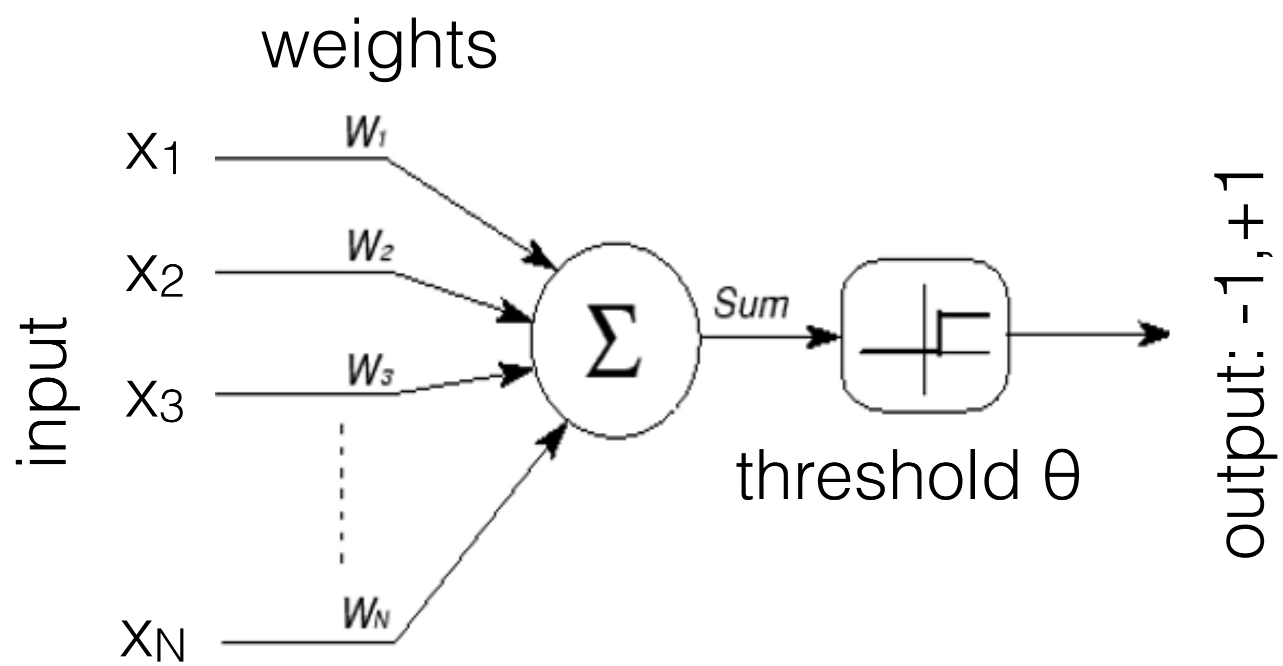
Artificial neuron

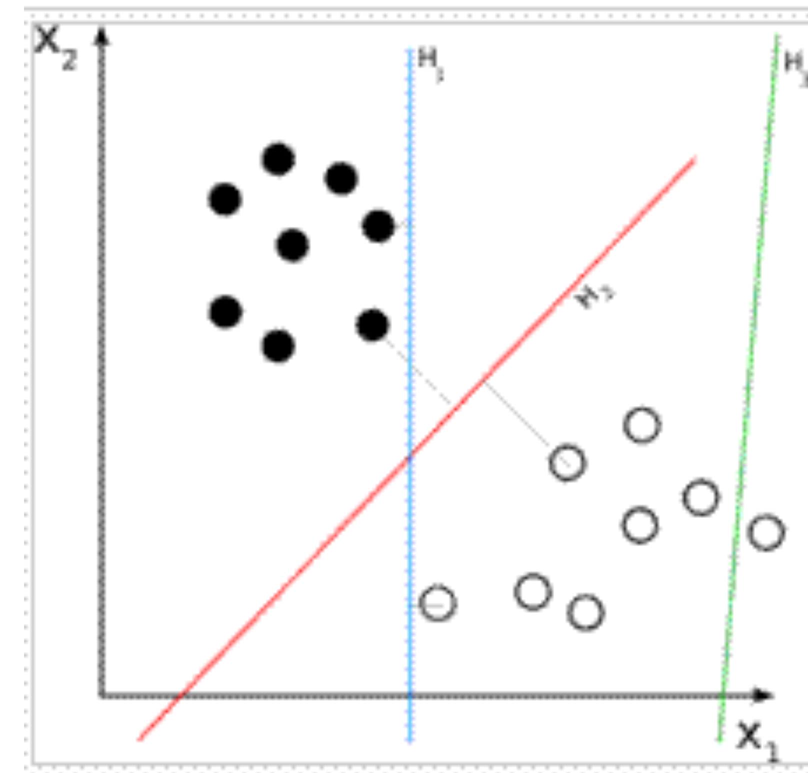




- we compare the weighted sum of inputs to a threshold
- two possible output values - a binary classification of input combinations
- linear separation

Artificial neuron

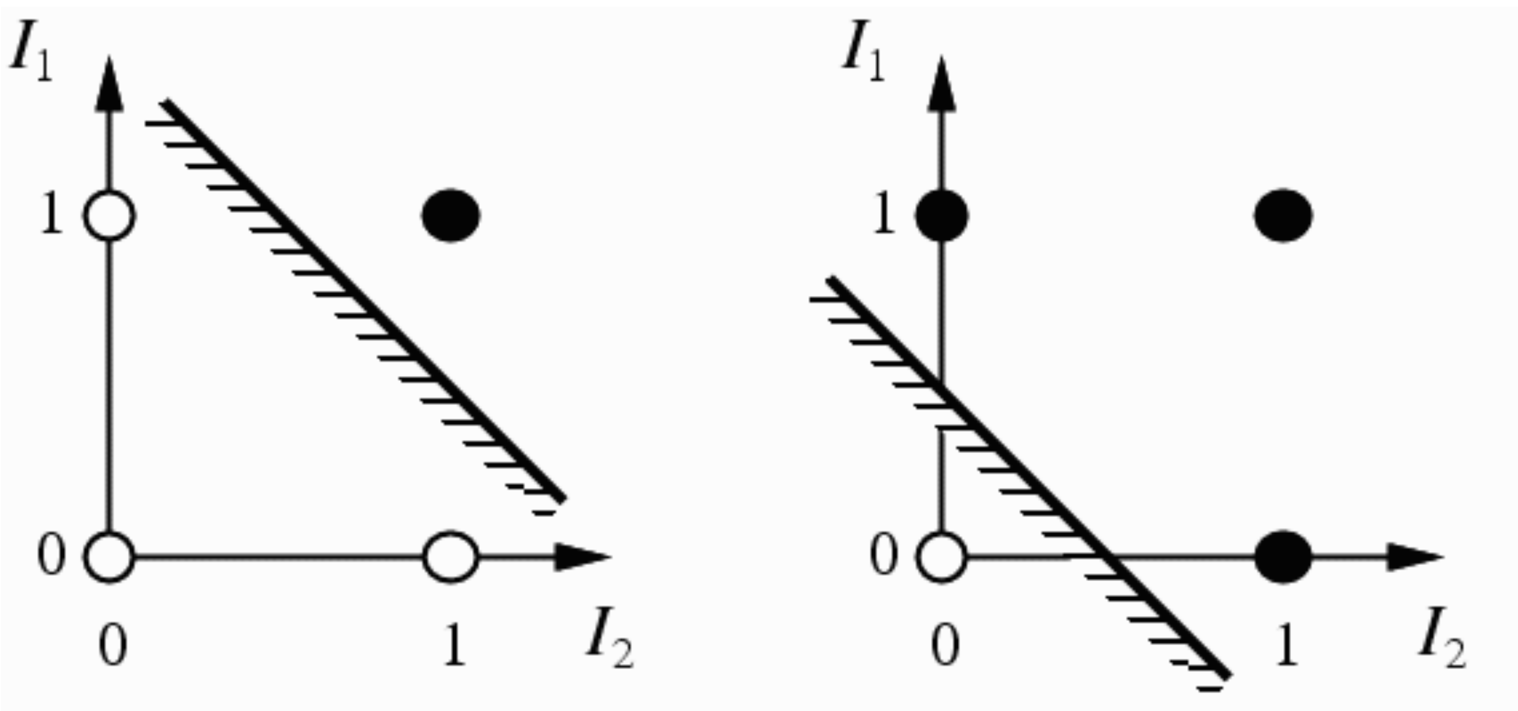




- we compare the weighted sum of inputs to a threshold
- two possible output values - a binary classification of input combinations
- linear separation

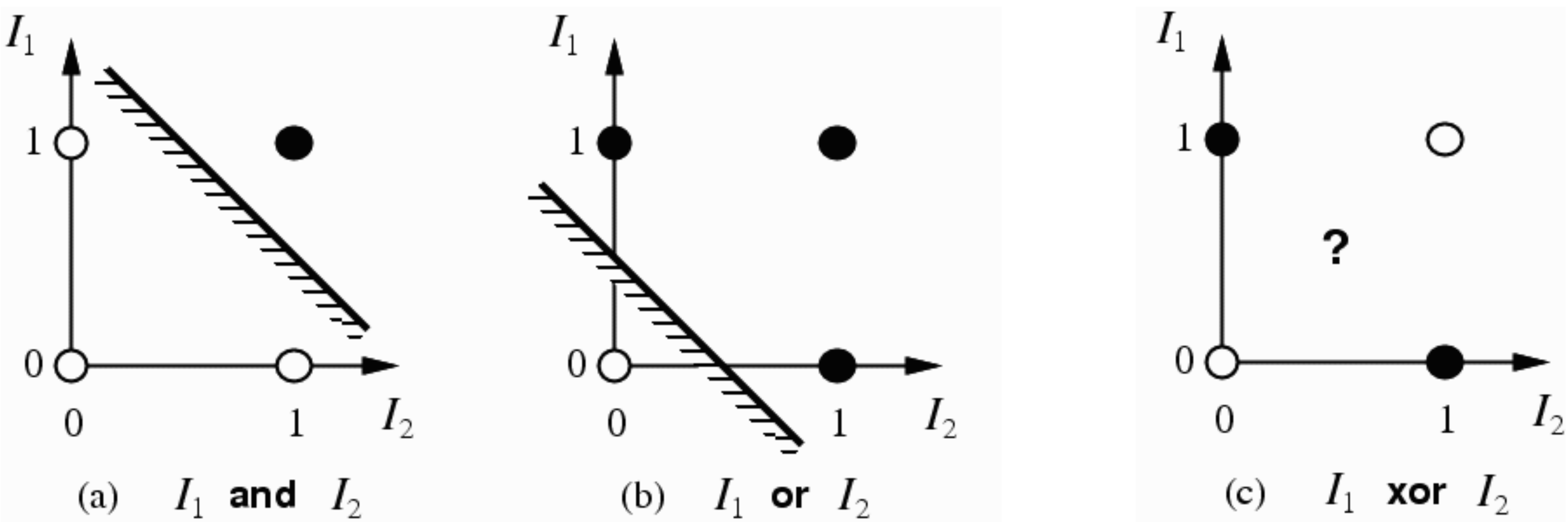
$$\theta = x_1 w_1 + x_2 w_2 \longrightarrow x_2 = \frac{-w_1}{w_2} x_1 + \frac{\theta}{w_2}$$

Logical operations by artificial neurons

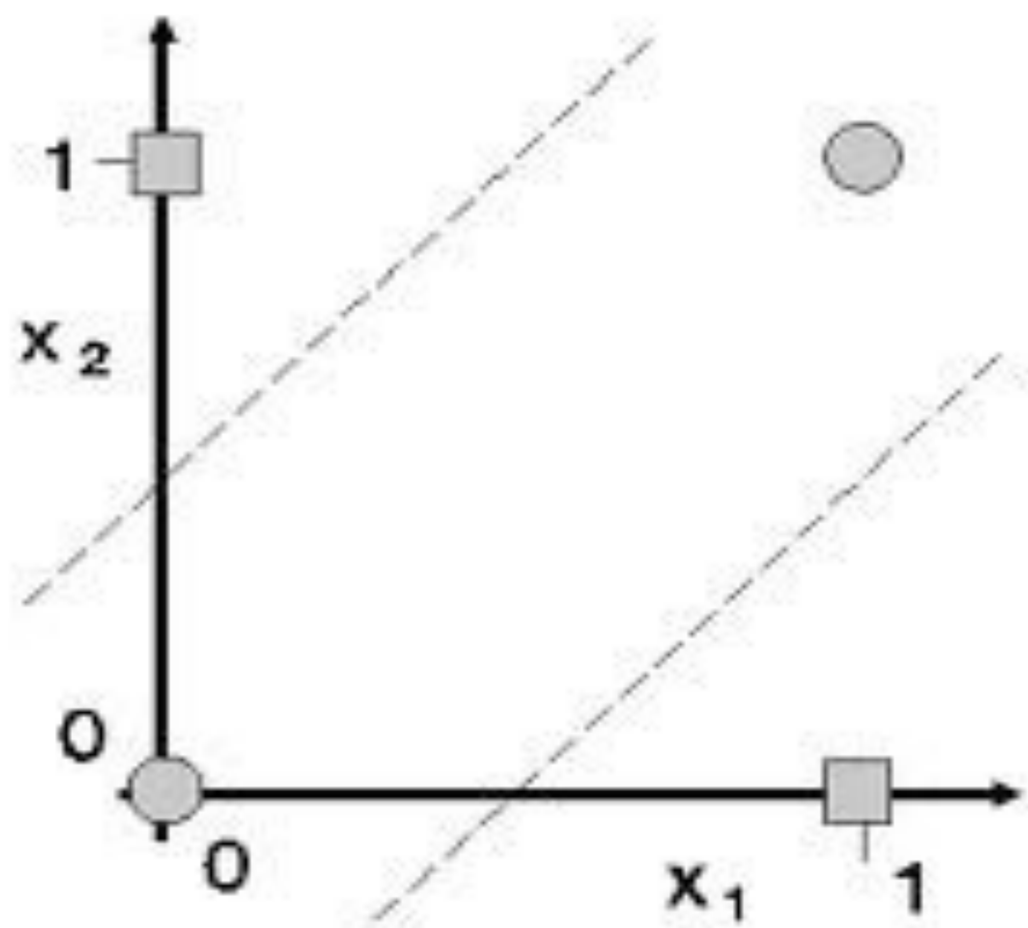


(a) I_1 **and** I_2 (b) I_1 **or** I_2

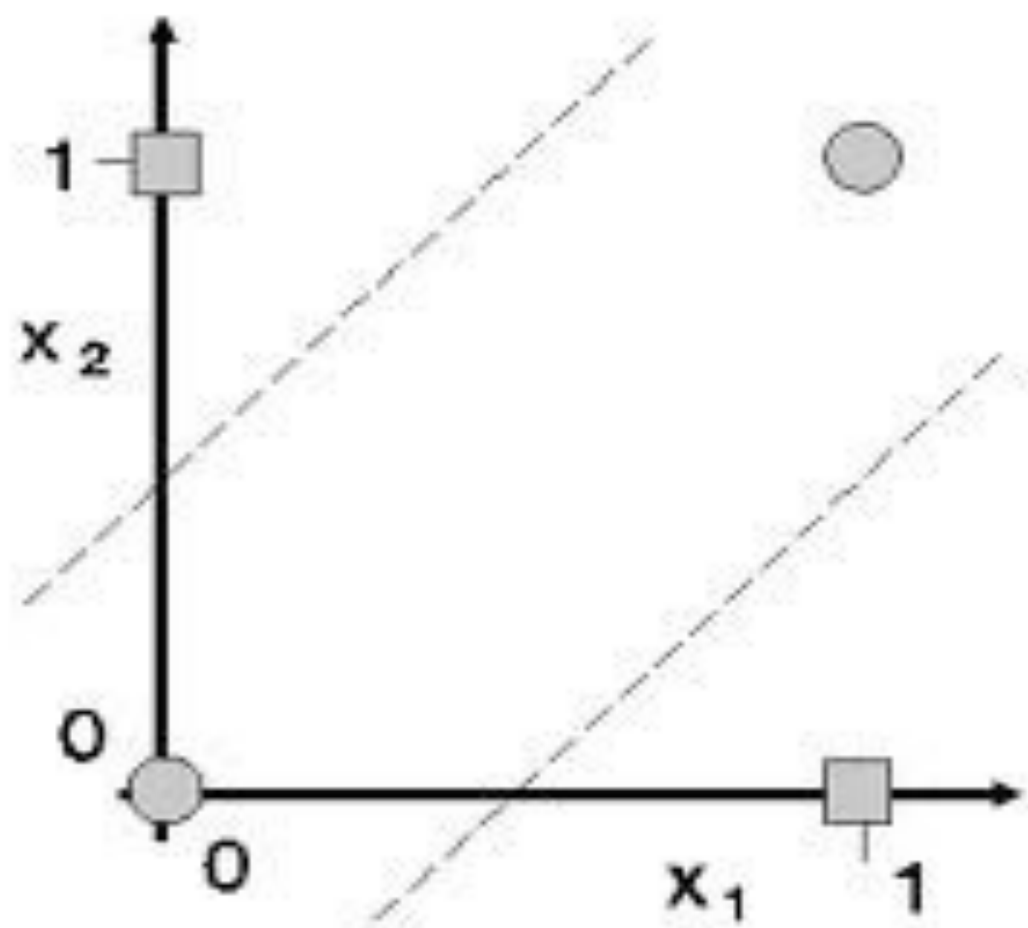
Logical operations by artificial neurons

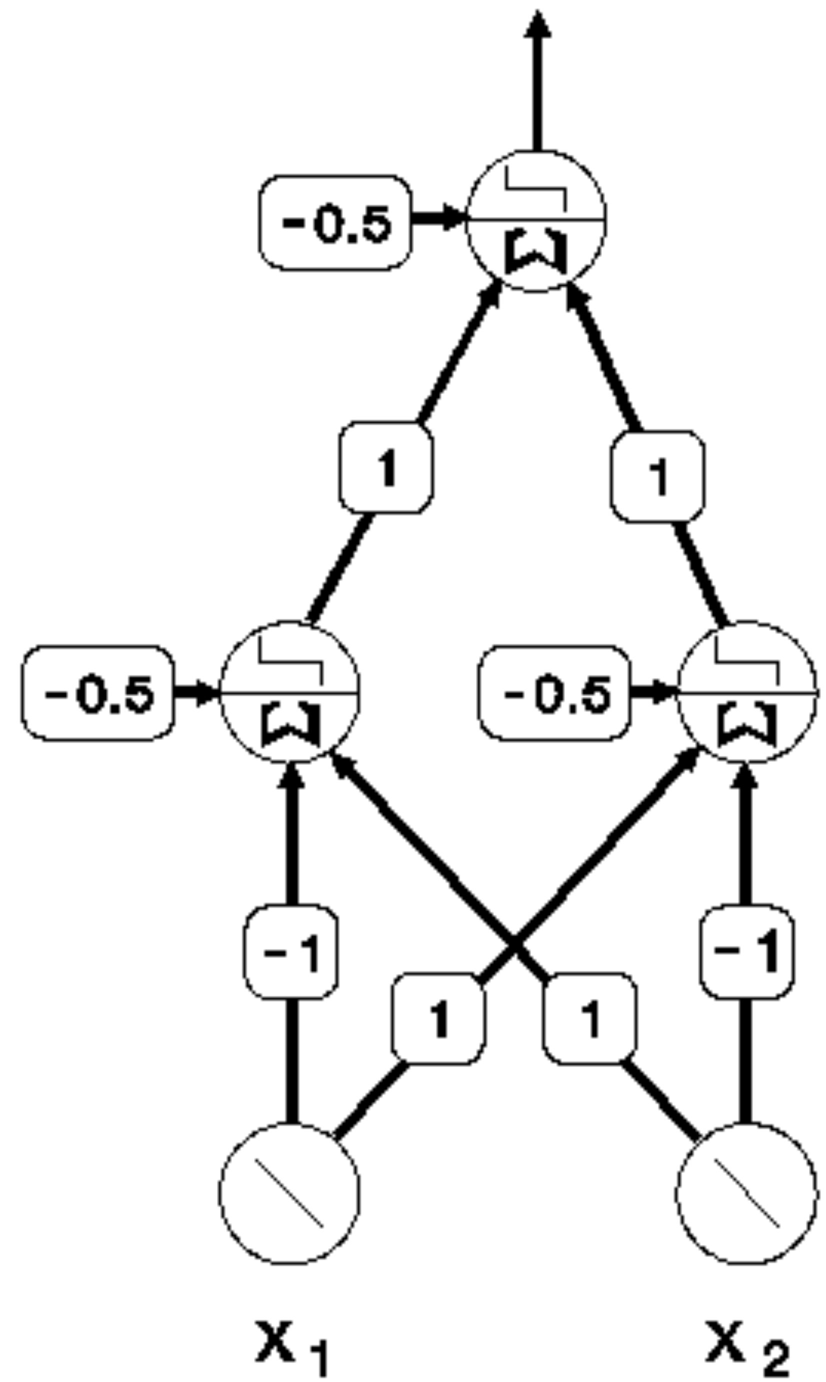


Solution of the XOR problem

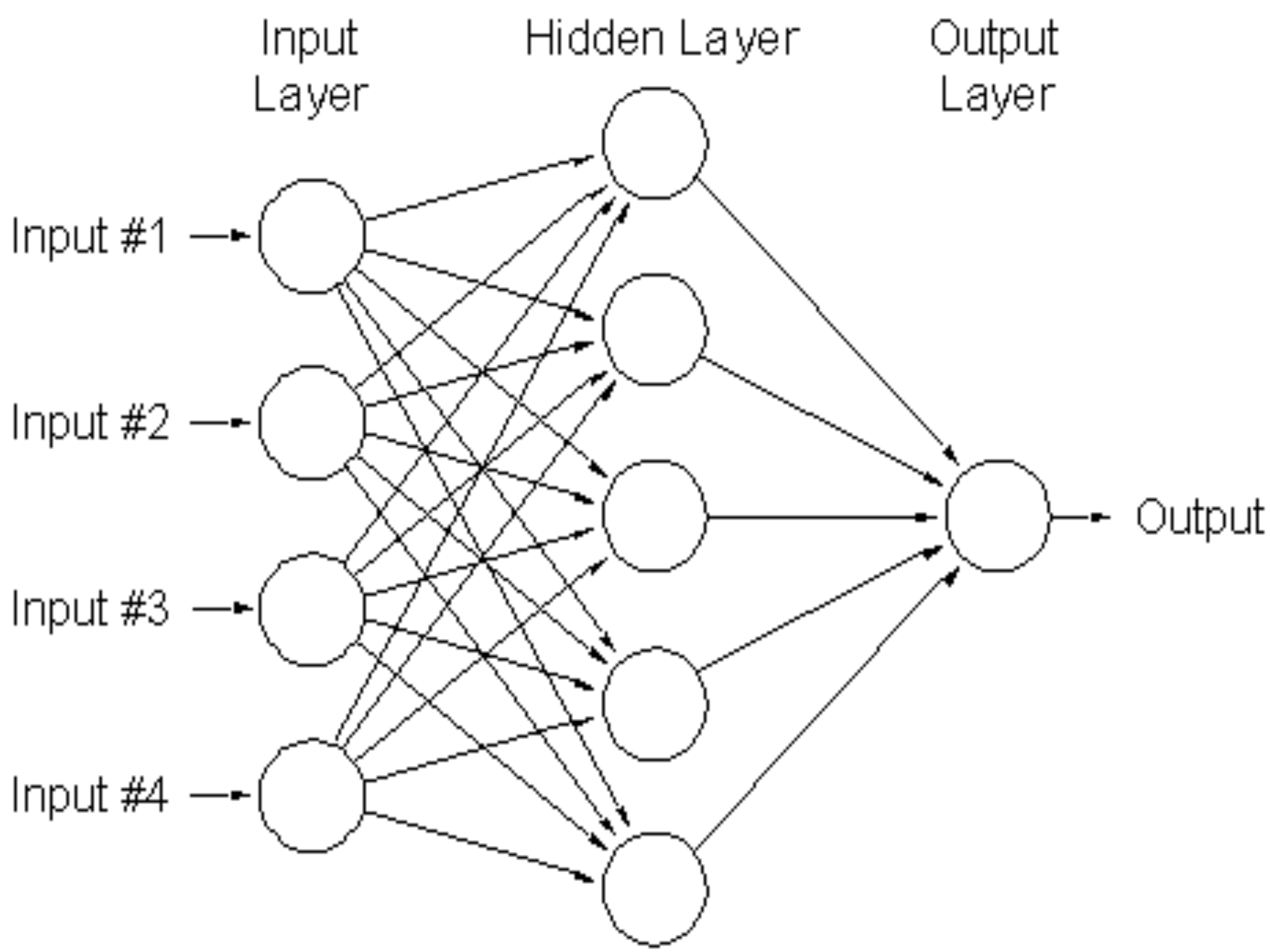


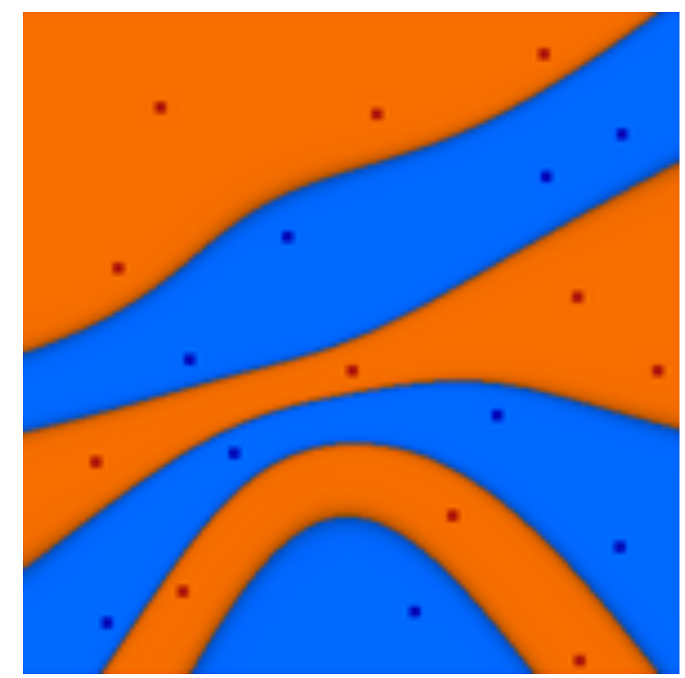
Solution of the XOR problem



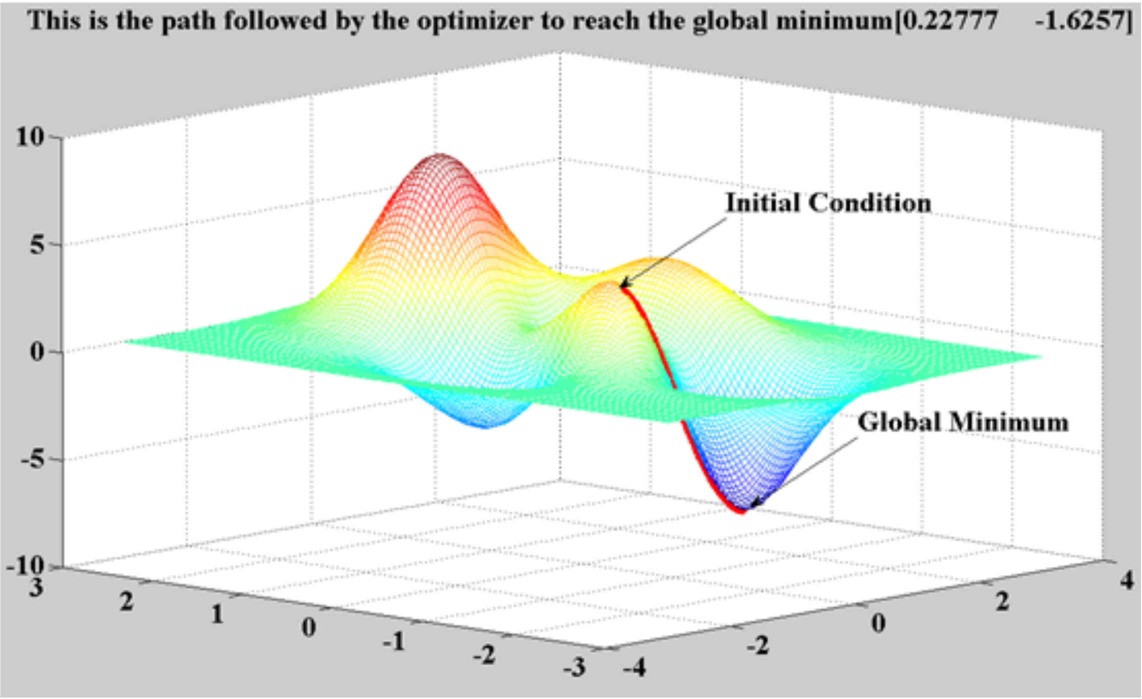


Multi-layer neural network





Gradient descent



- General scheme for parameter optimization
- At each point, we calculate the derivative of the error function with respect to the weights
- We move the weights towards the negative of the gradient
- Finds a local minimum

$$w_b = w_b - \epsilon \frac{\partial E}{\partial w_b}$$

Looking for the best weight values

- given a specific dataset and a set of values for the synaptic weights, the network will misclassify a number of data samples

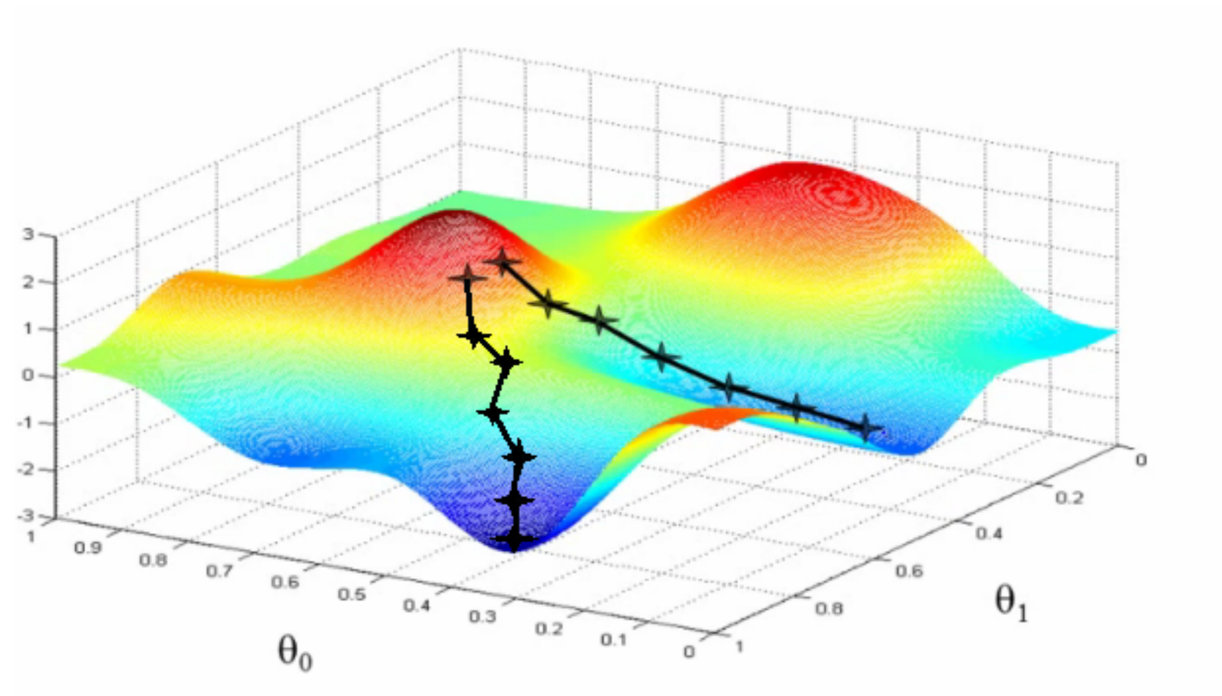
Looking for the best weight values

- given a specific dataset and a set of values for the synaptic weights, the network will misclassify a number of data samples

- we are looking for the weight values that minimize this error

Looking for the best weight values

- given a specific dataset and a set of values for the synaptic weights, the network will misclassify a number of data samples
- we are looking for the weight values that minimize this error
- setting the many weights parallelly is a hard optimization problem



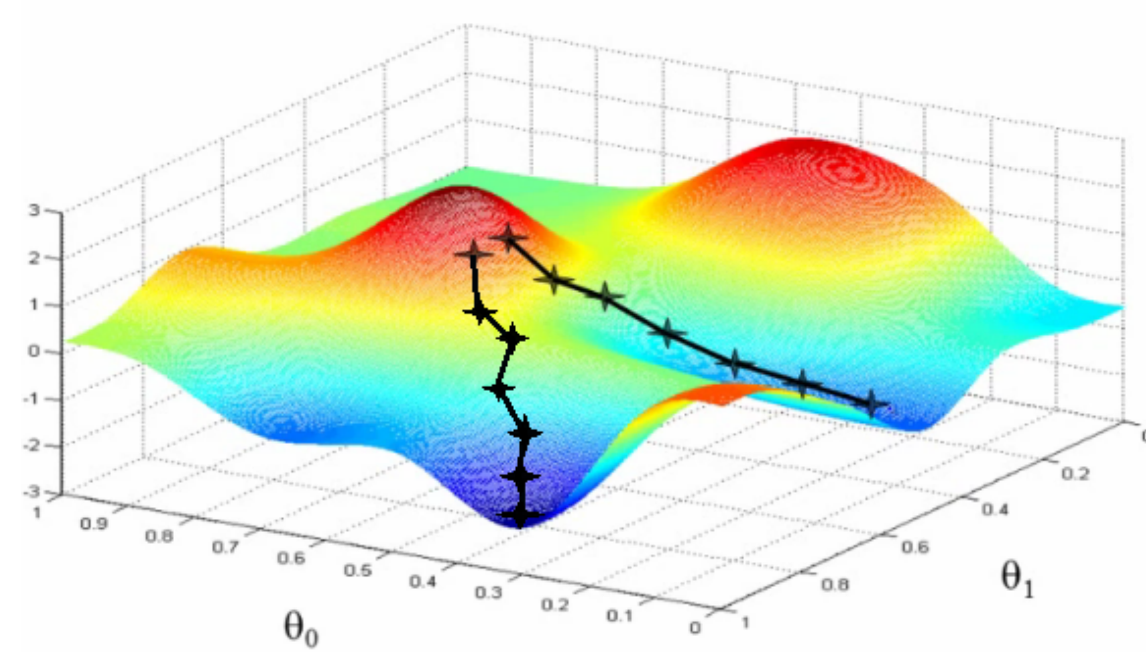
Looking for the best weight values

- given a specific dataset and a set of values for the synaptic weights, the network will misclassify a number of data samples

- we are looking for the weight values that minimize this error

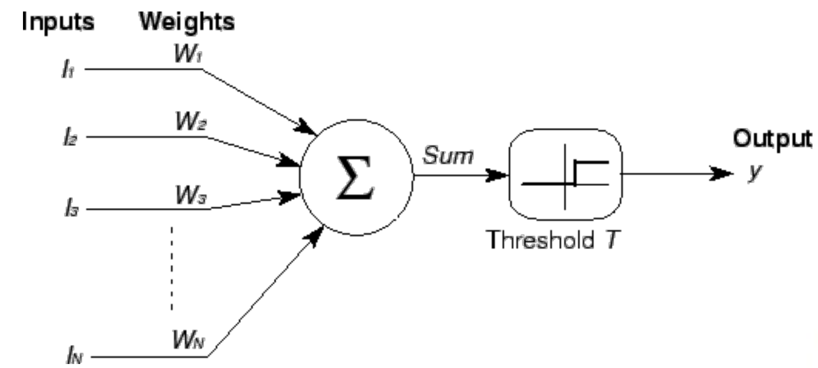
- setting the many weights parallelly is a hard optimization problem

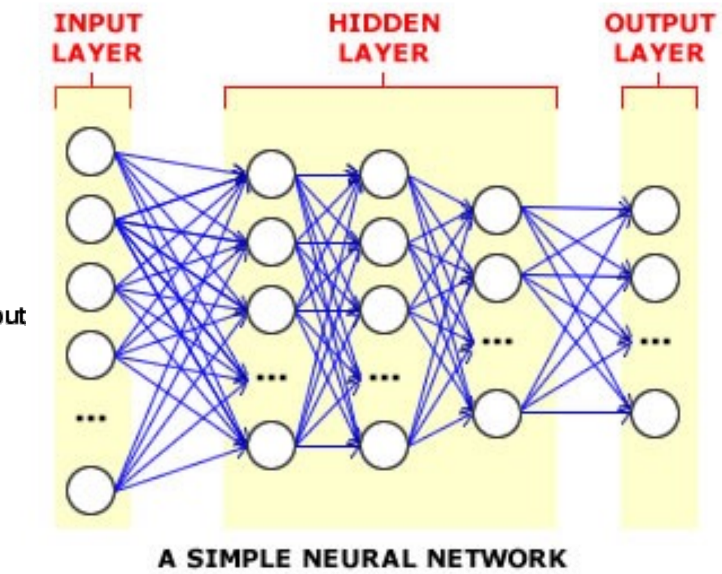
- the back propagation algorithm is a gradient search that finds an approximate optimum

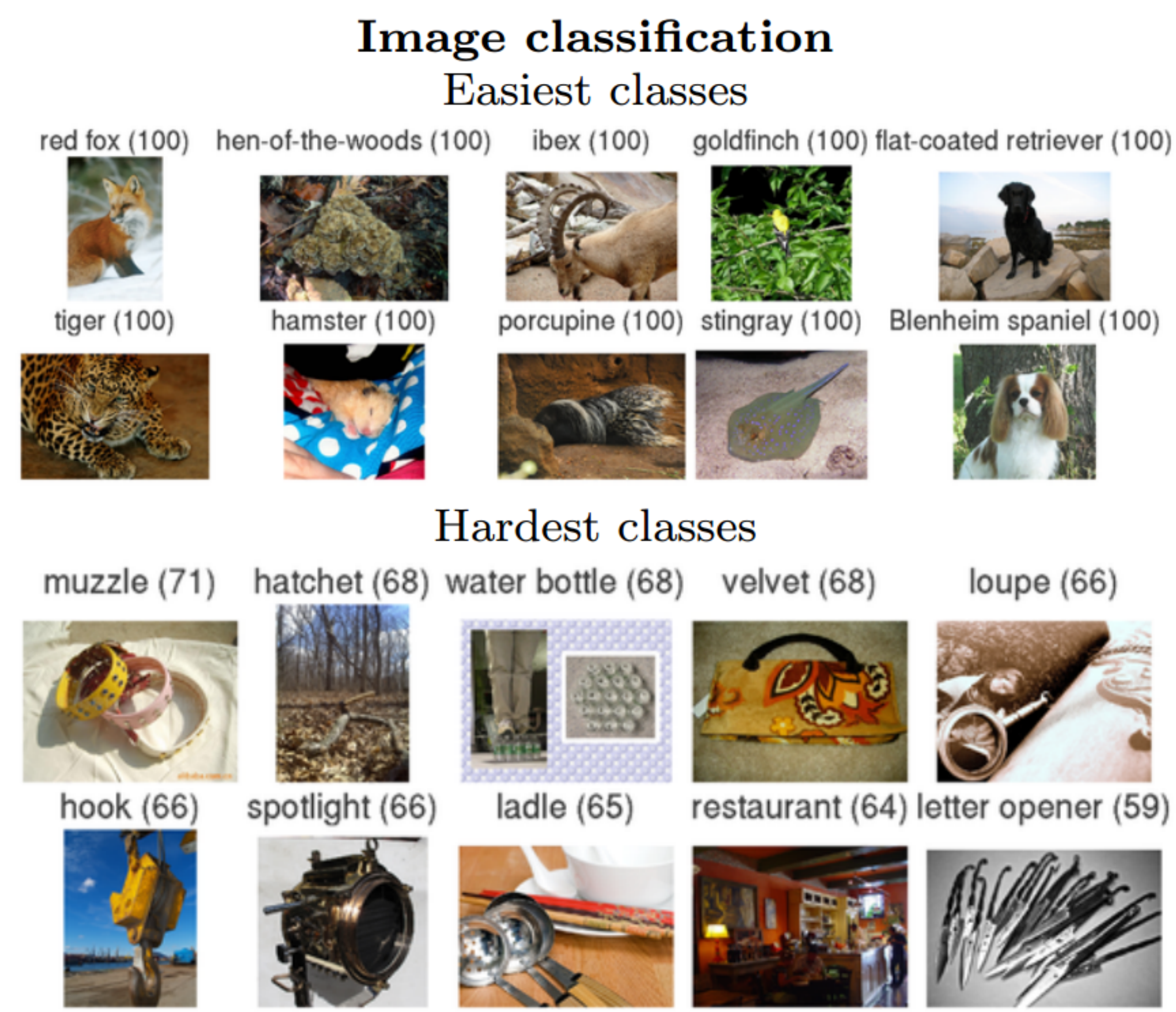


Deep networks

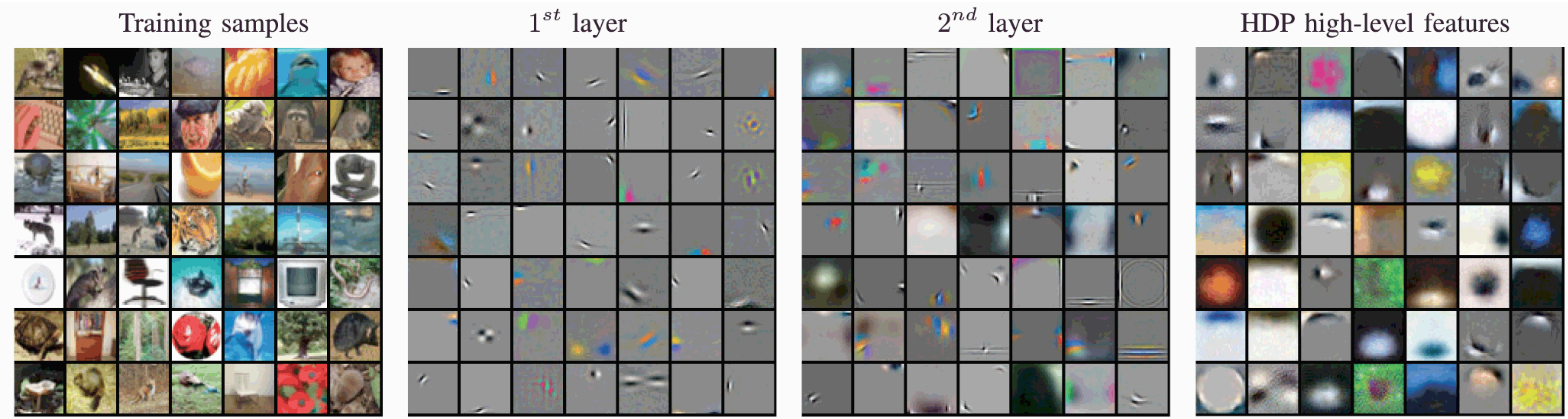
- Deep learning is a highly successful machine learning framework which is employed in a variety of computer tasks, including object recognition
- It employs a computational architecture in which the basic element is a simplified model neuron
- the basic elements are organised into layers, connected by weights
- each layer performs a transformation of an input image
- network weights are tuned to perform classification of the inputs into categories - each category is an object

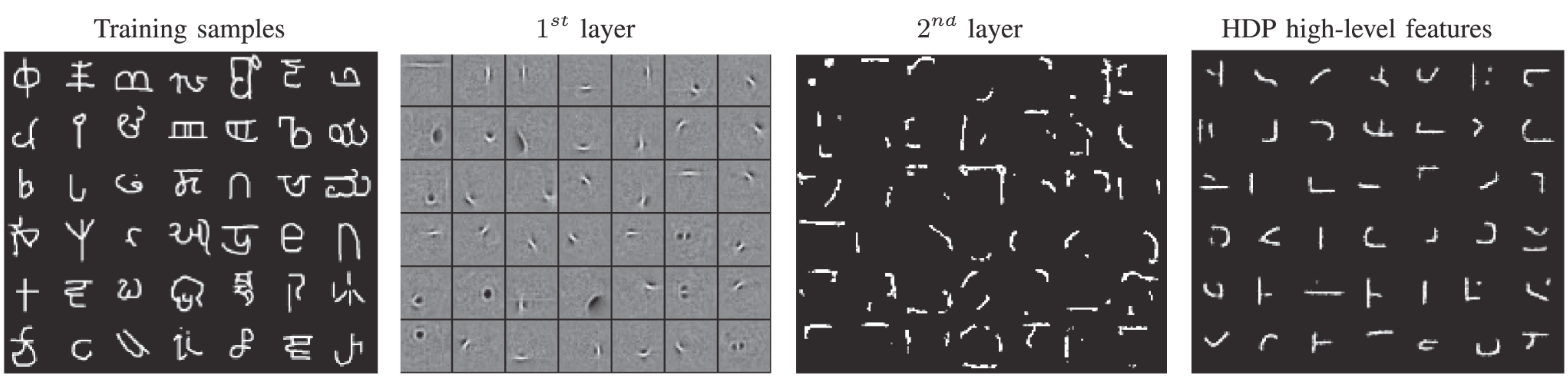




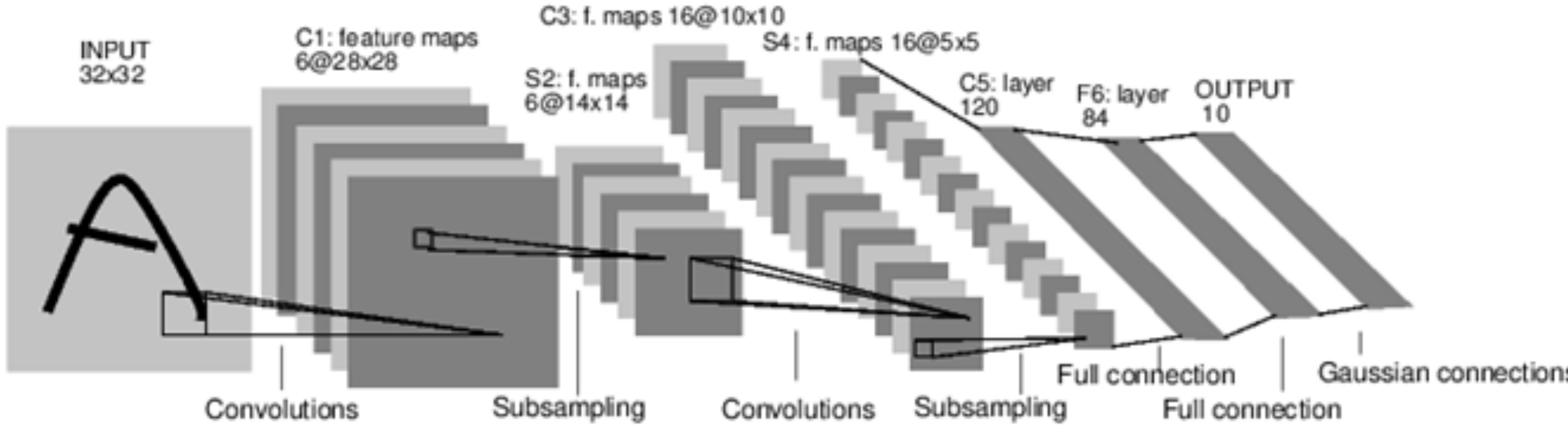


Visual features acquired by deep learning





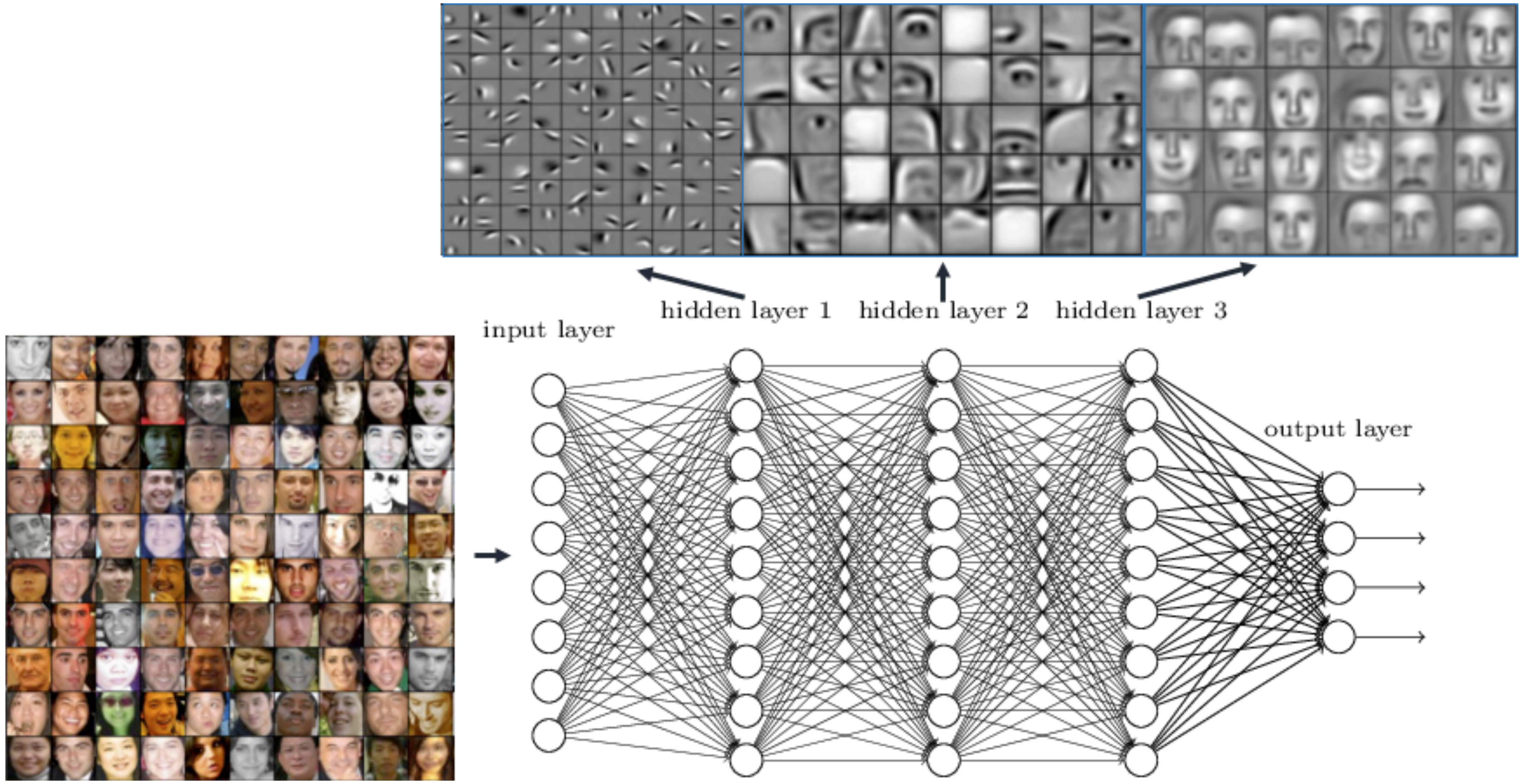
Convolutional networks

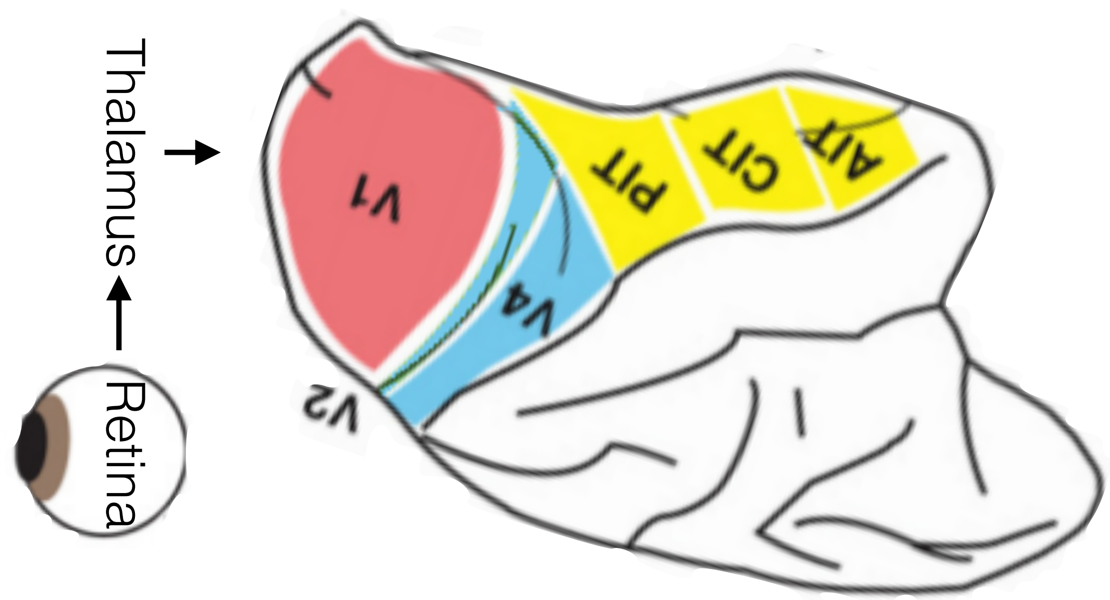


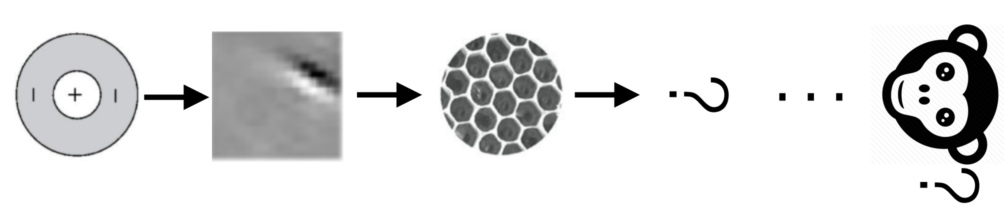
A Full Convolutional Neural Network (LeNet)

- Step1: a set of translation-invariant feature extractions
- Step 2: Subsampling the result to a lower-dimensional space
- repeat

Similarities to the visual cortex







Human against machine

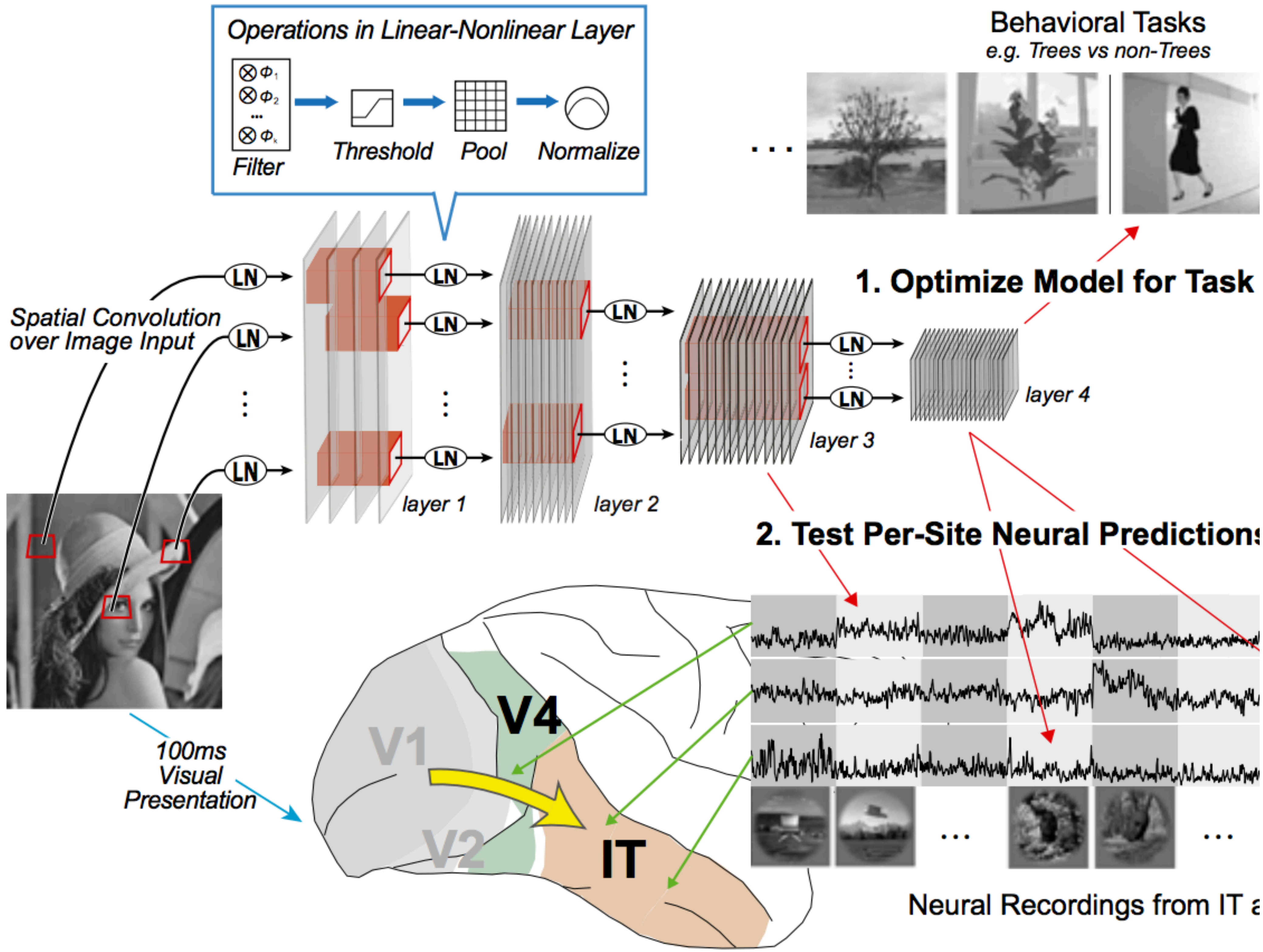
method	top-1 err.	top-5 err.
VGG [40] (ILSVRC'14)	-	8.43†
GoogLeNet [43] (ILSVRC'14)	-	7.89
VGG [40] (v5)	24.4	7.1
PReLU-net [12]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

5.1

Andrej "the human benchmark" Karpathy

Prediction of neural activity with deep networks

Prediction of neural activity with deep networks

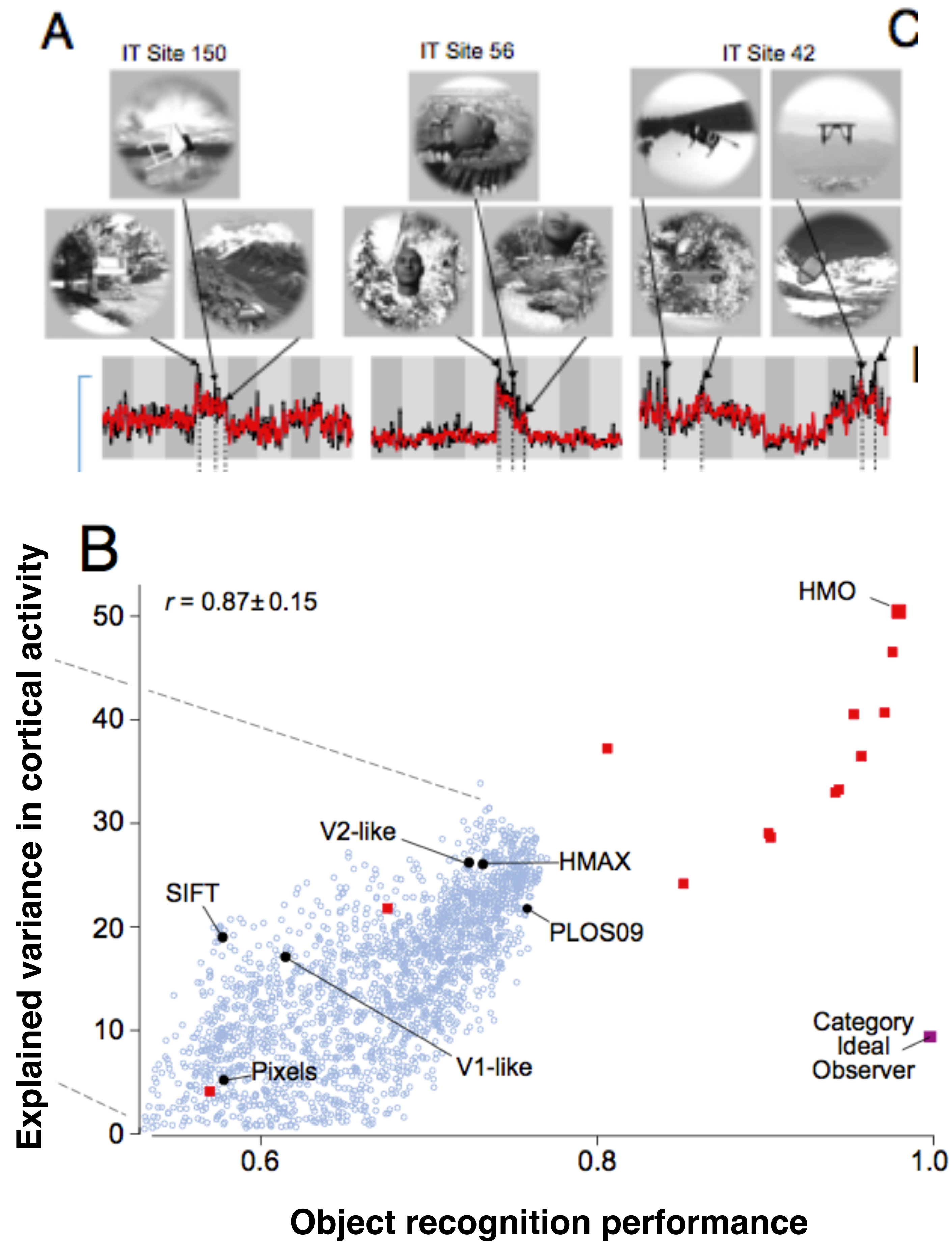


Yamins et al 2014 **for V1: Cadena et al 2019**

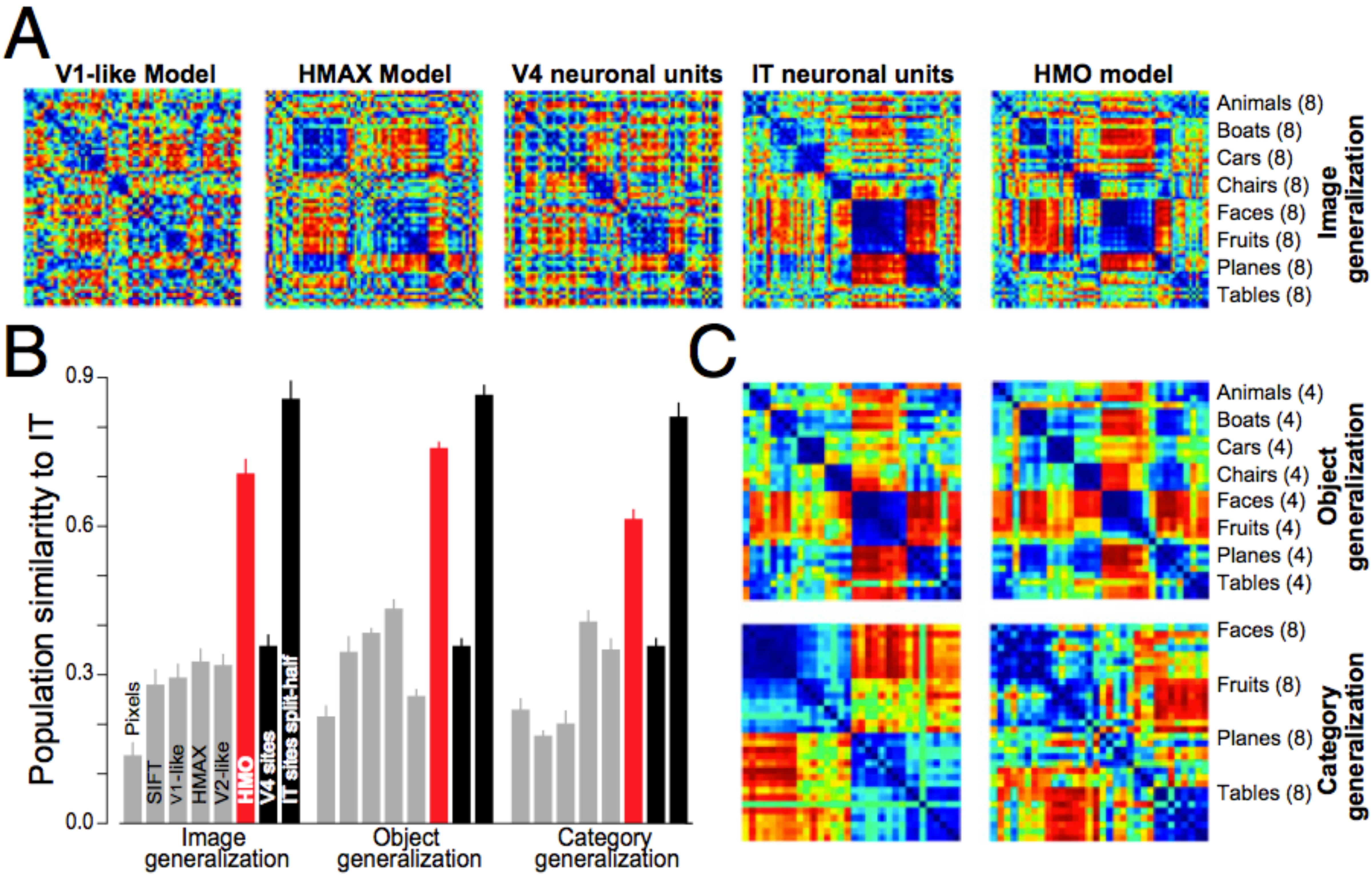
Yamins et al, 2014

Deep convolutional networks' predictions of neural activity

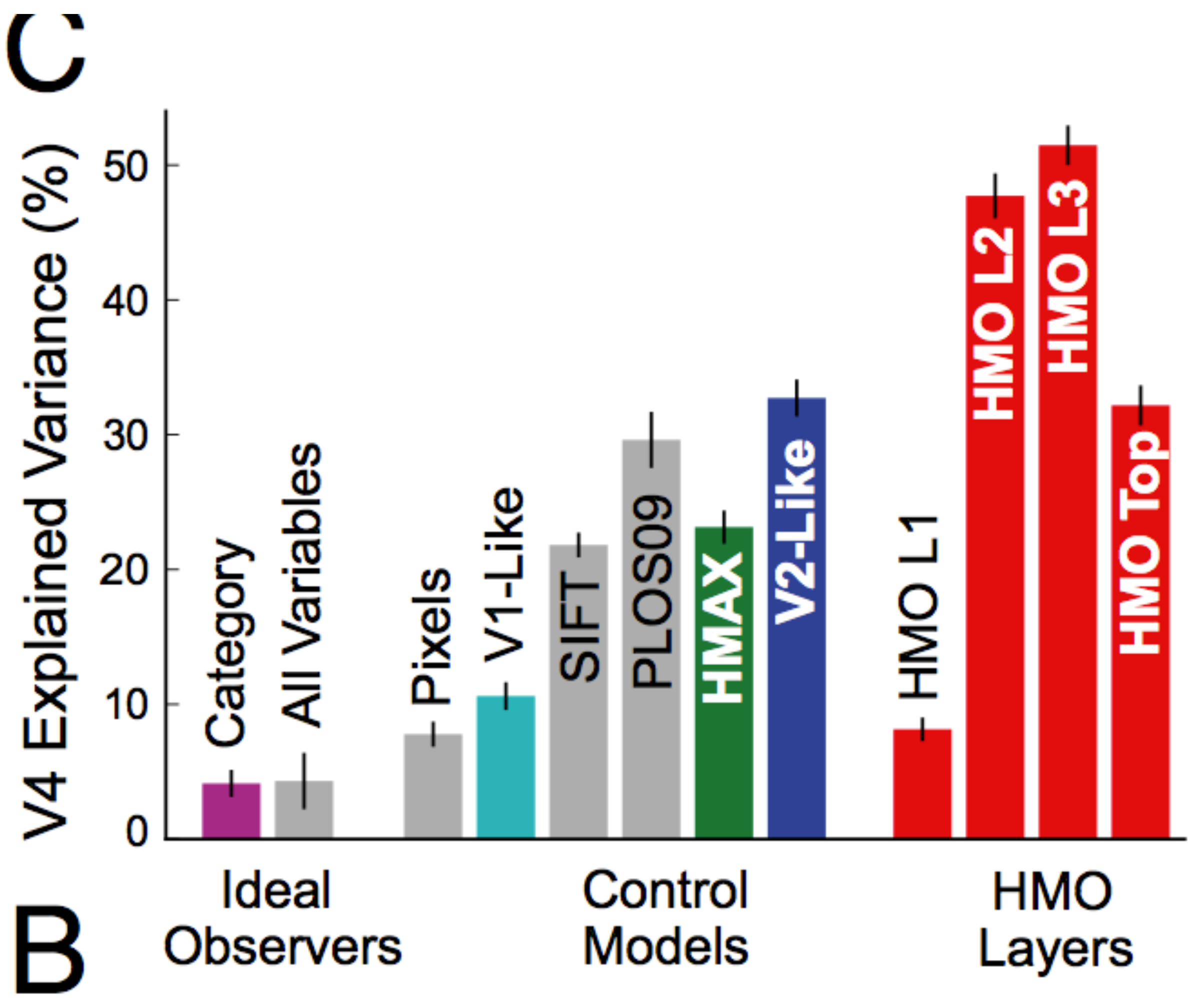
- The inferotemporal cortex (IT) is thought to be involved in object recognition
- A deep network is trained to recognise objects from images
- The top layer of the network is compared to measured activity in IT

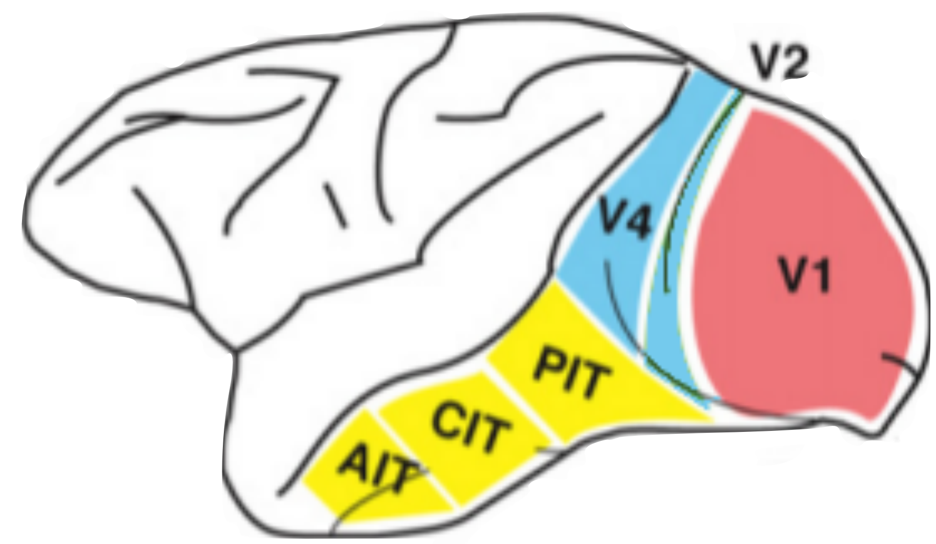


Population-level representational similarity



Predicting V4 activity





Biologically realistic error backpropagation?

Biologically realistic error backpropagation?

- BP has multiple elements that are questionable from a biological point of view

Biologically realistic error backpropagation?

- BP has multiple elements that are questionable from a biological point of view
  - explicit computation of the error term

Biologically realistic error backpropagation?

- BP has multiple elements that are questionable from a biological point of view
  - explicit computation of the error term
  - feed-forward and feed-back weights are tied together

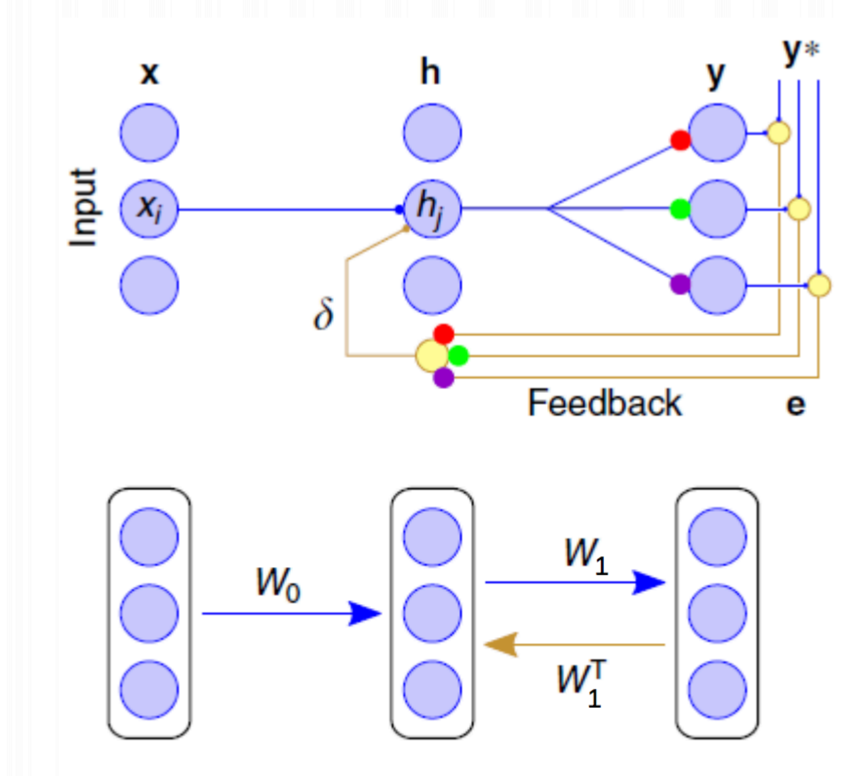
Biologically realistic error backpropagation?

- BP has multiple elements that are questionable from a biological point of view
  - explicit computation of the error term
  - feed-forward and feed-back weights are tied together
  - derivative of the activation function

Biologically realistic error backpropagation?

- BP has multiple elements that are questionable from a biological point of view
  - explicit computation of the error term
  - feed-forward and feed-back weights are tied together
  - derivative of the activation function
  - …

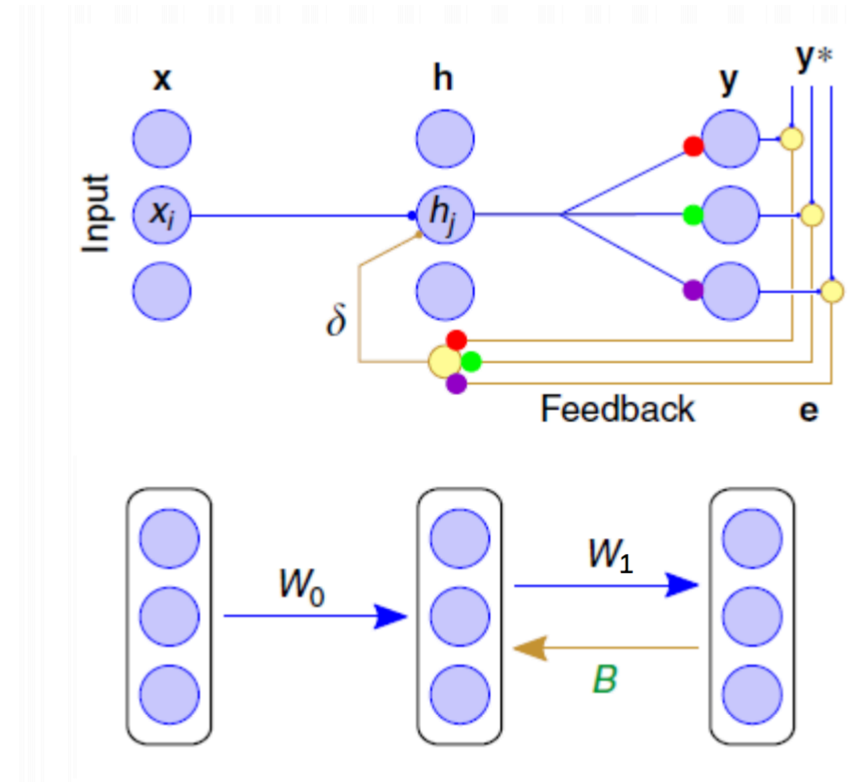
Biologically realistic error backpropagation?



- BP has multiple elements that are questionable from a biological point of view
  - explicit computation of the error term
  - feed-forward and feed-back weights are tied together
  - derivative of the activation function
  - …

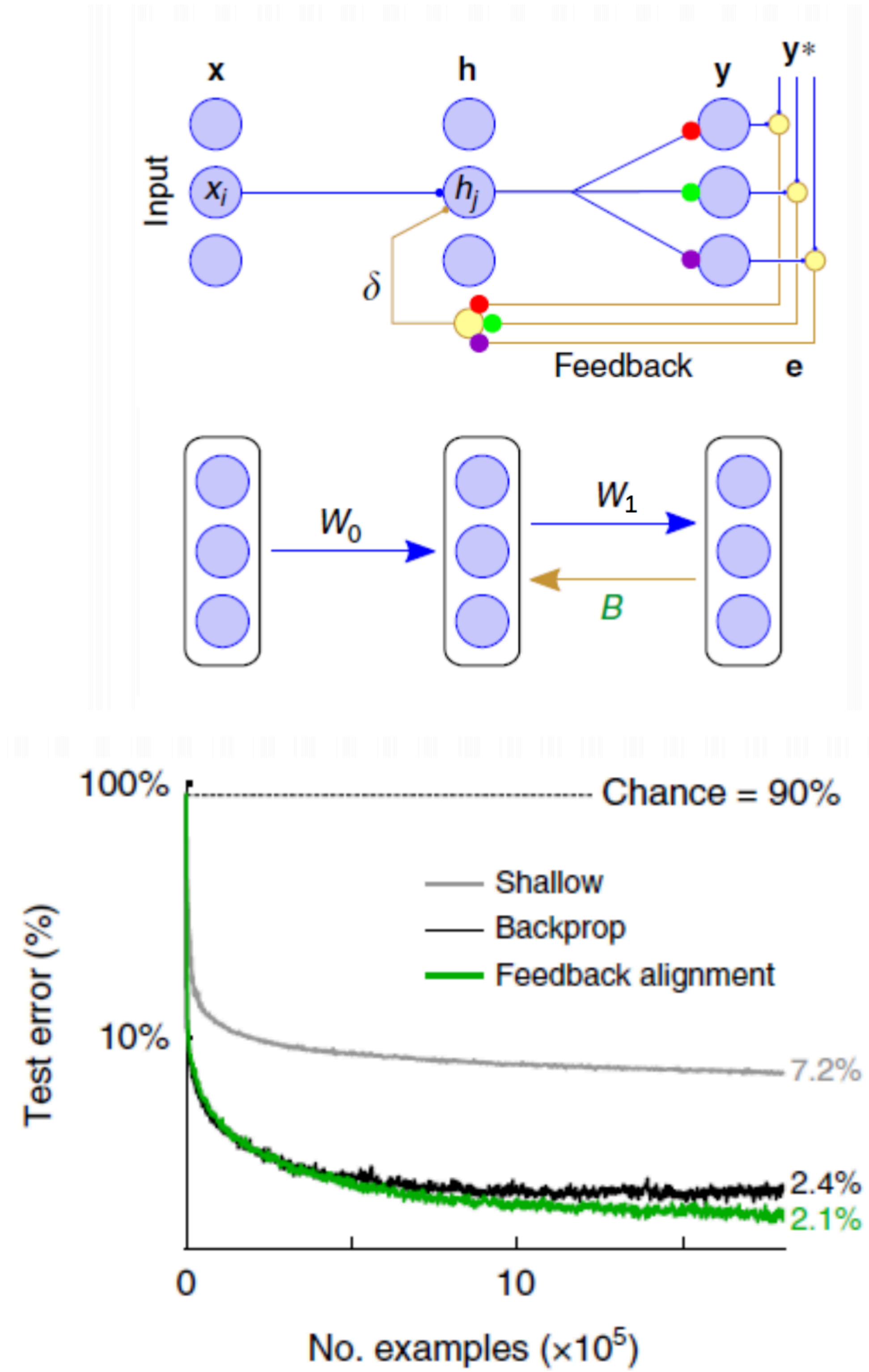
Biologically realistic error backpropagation?

- BP has multiple elements that are questionable from a biological point of view
  - explicit computation of the error term
  - feed-forward and feed-back weights are tied together
  - derivative of the activation function
  - …



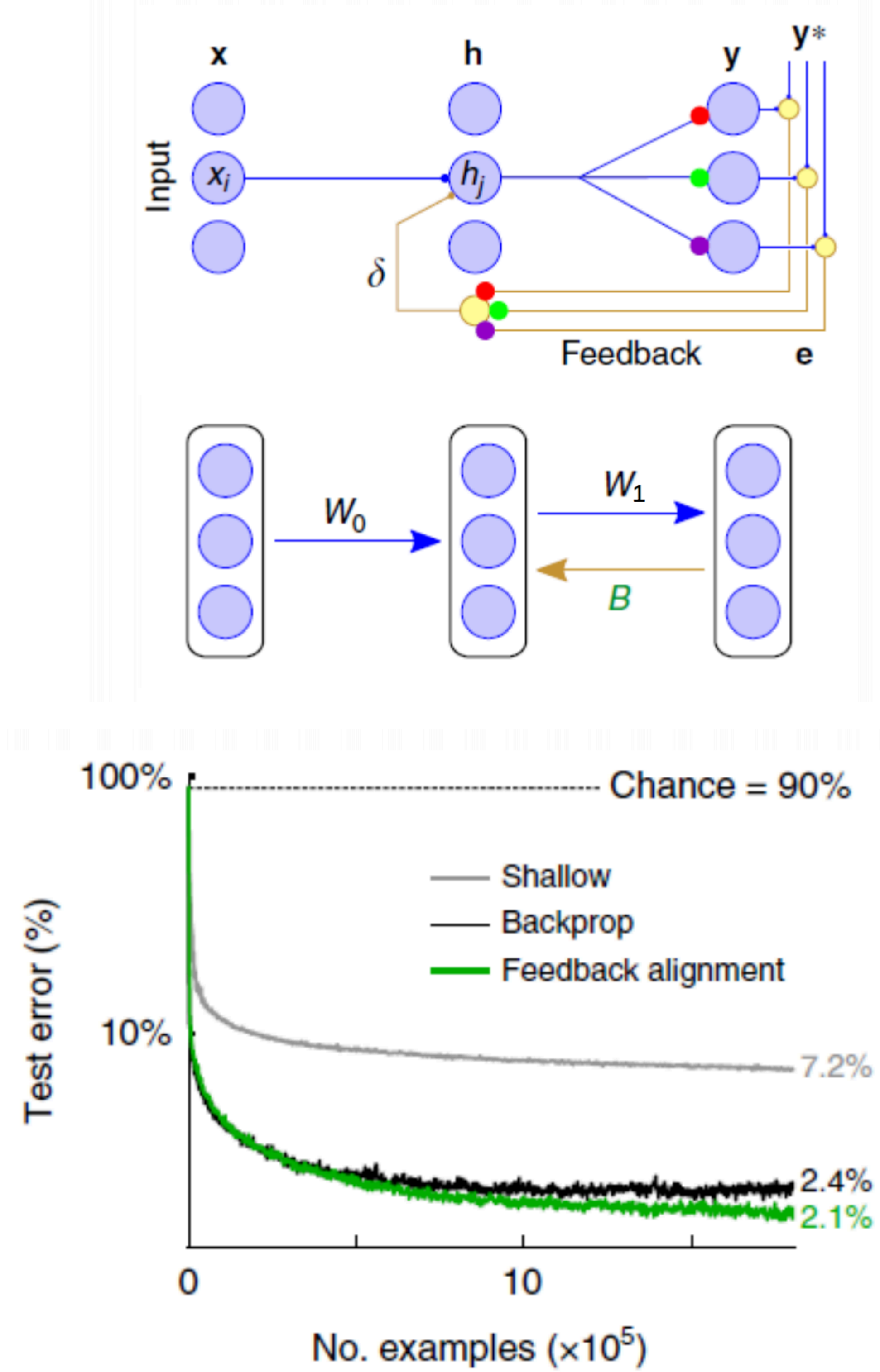
Biologically realistic error backpropagation?

- BP has multiple elements that are questionable from a biological point of view
    - explicit computation of the error term
    - feed-forward and feed-back weights are tied together
    - derivative of the activation function
    - …



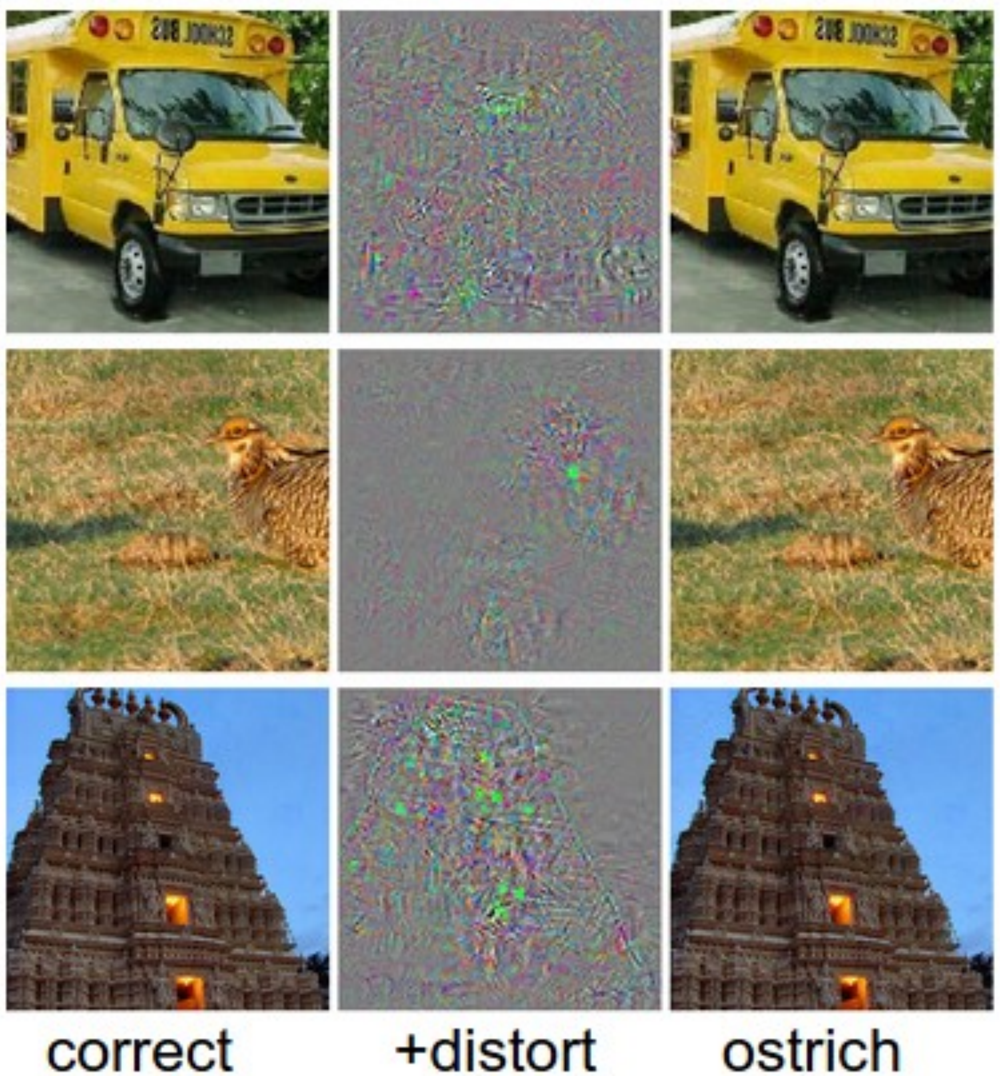
Biologically realistic error backpropagation?

- BP has multiple elements that are questionable from a biological point of view
  - explicit computation of the error term
  - feed-forward and feed-back weights are tied together
  - derivative of the activation function
  - …
- All major issues have solution by now



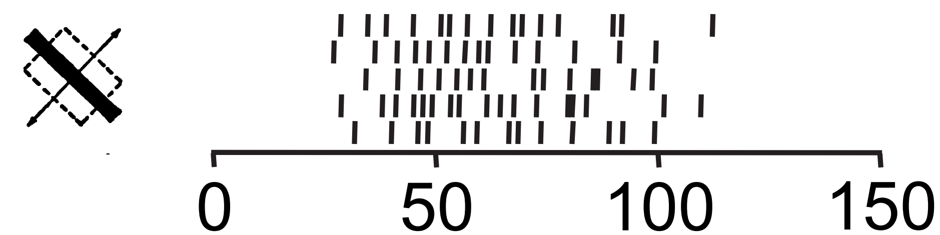
Problems with deep network models

Problems with deep network models

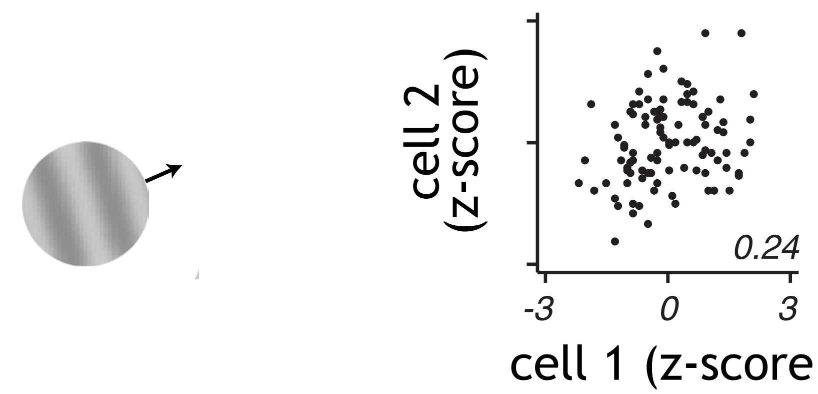


vs.

Problems with deep network models



Gur & Snodderly, Cereb Cortex 2006



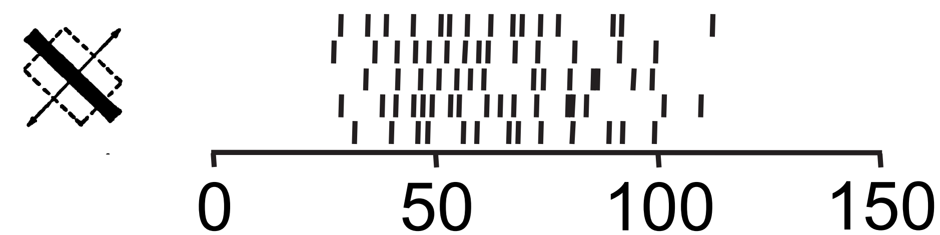
Kohn & Smith, J Neurosci 2005

?

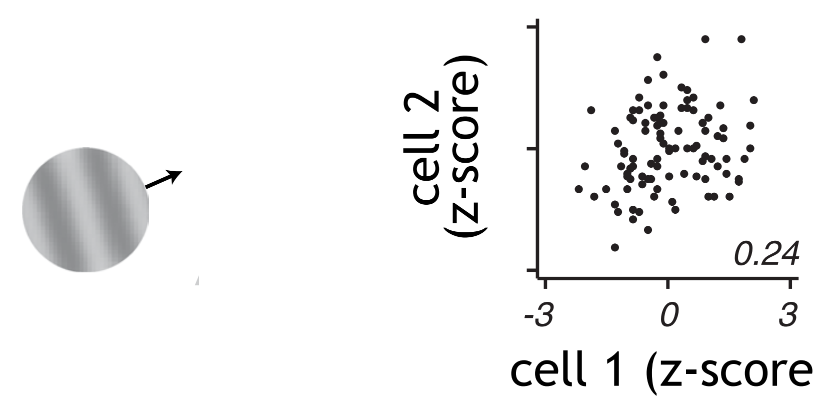
??

Problems with deep network models

?



Gur & Snodderly, Cereb Cortex 2006

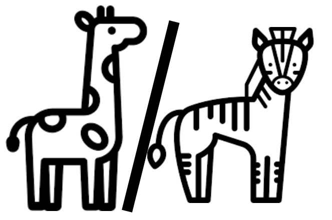


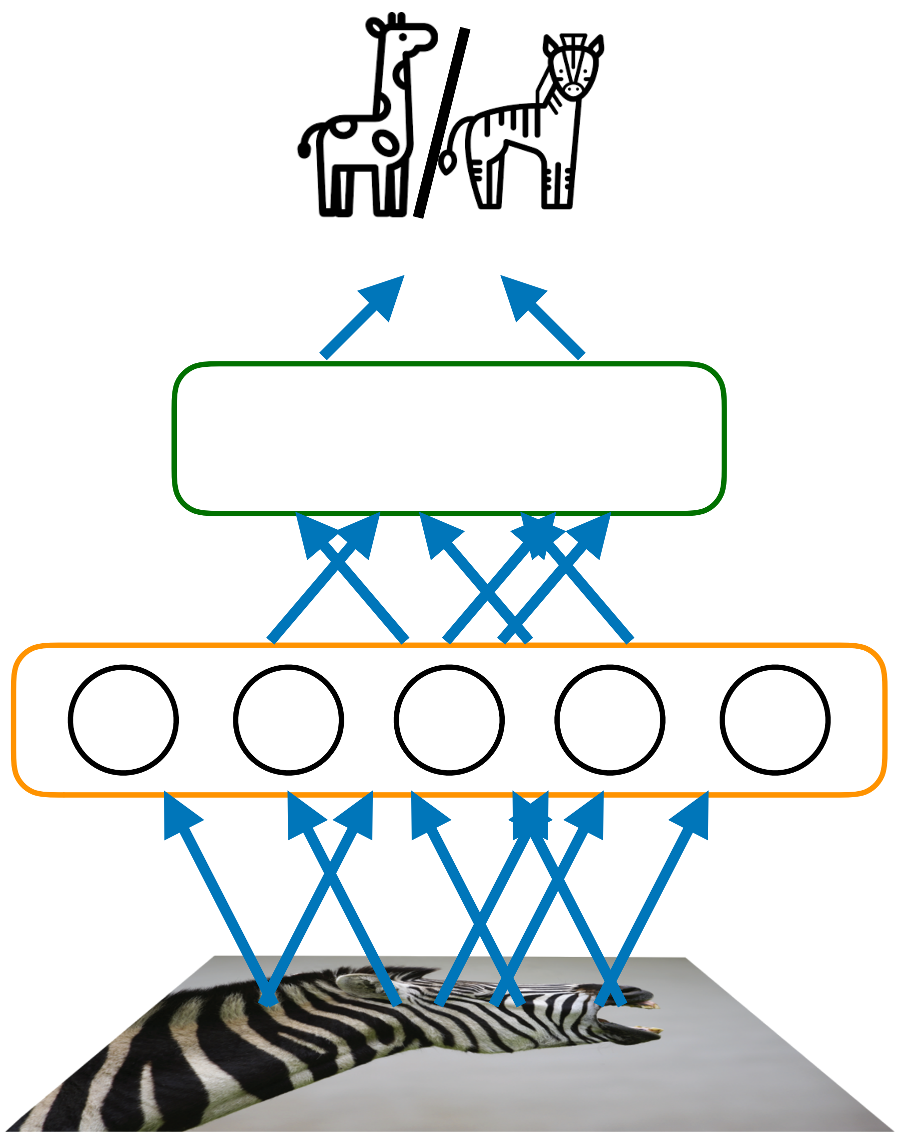
Kohn & Smith, J Neurosci 2005

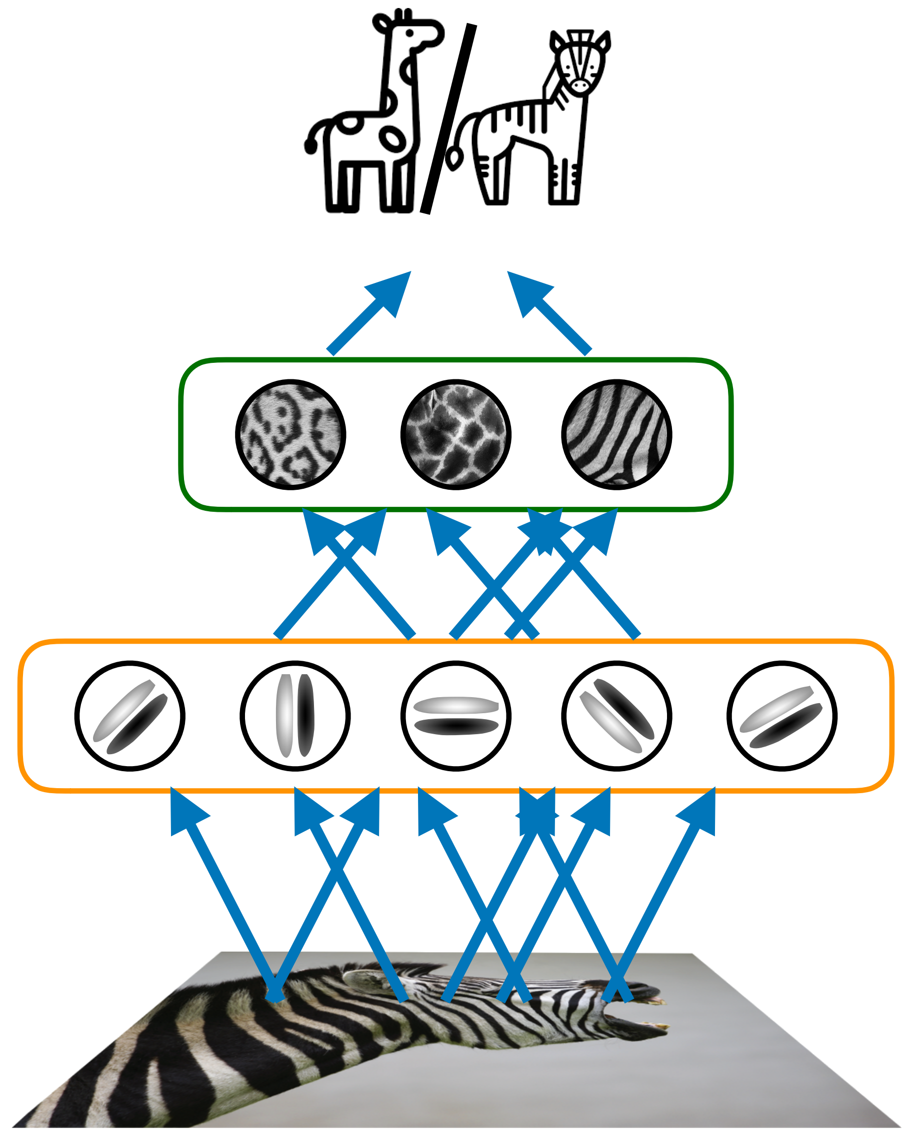
??

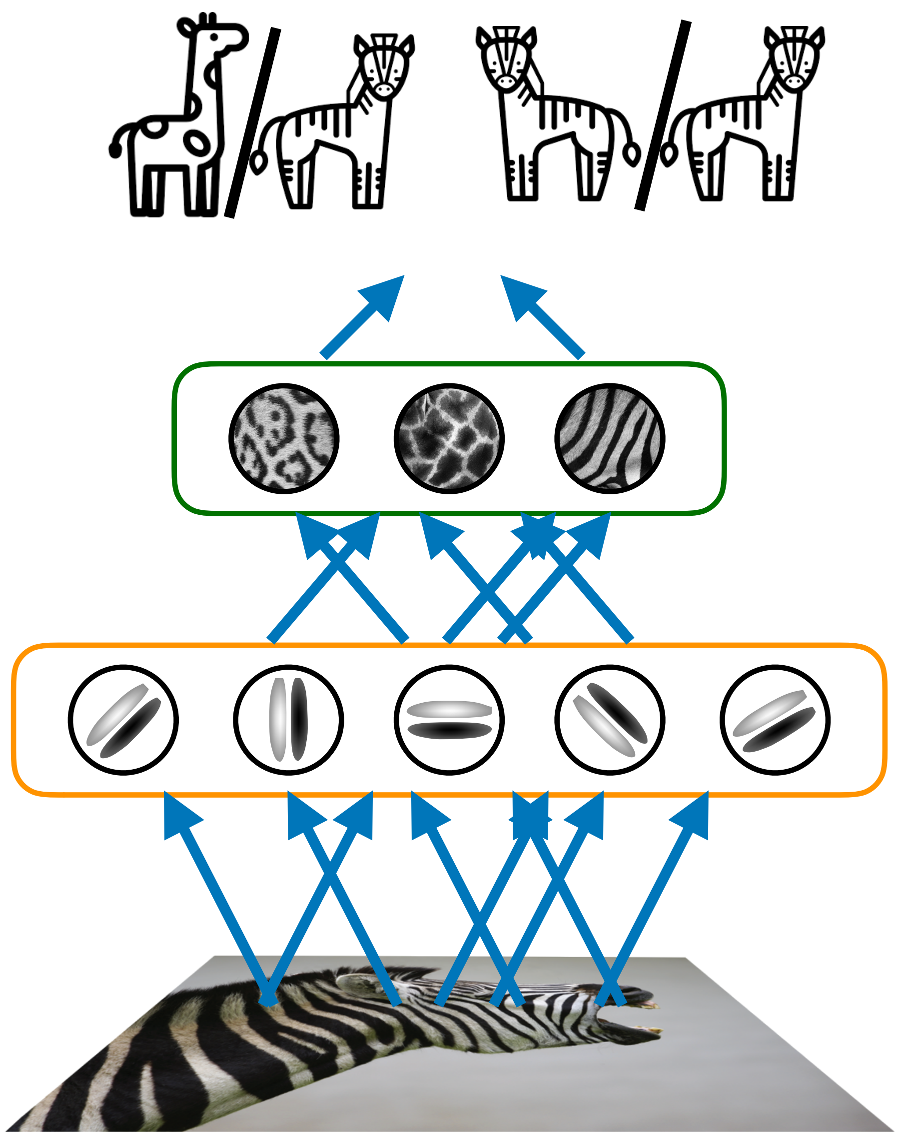
Probabilistic models vs. deep networks

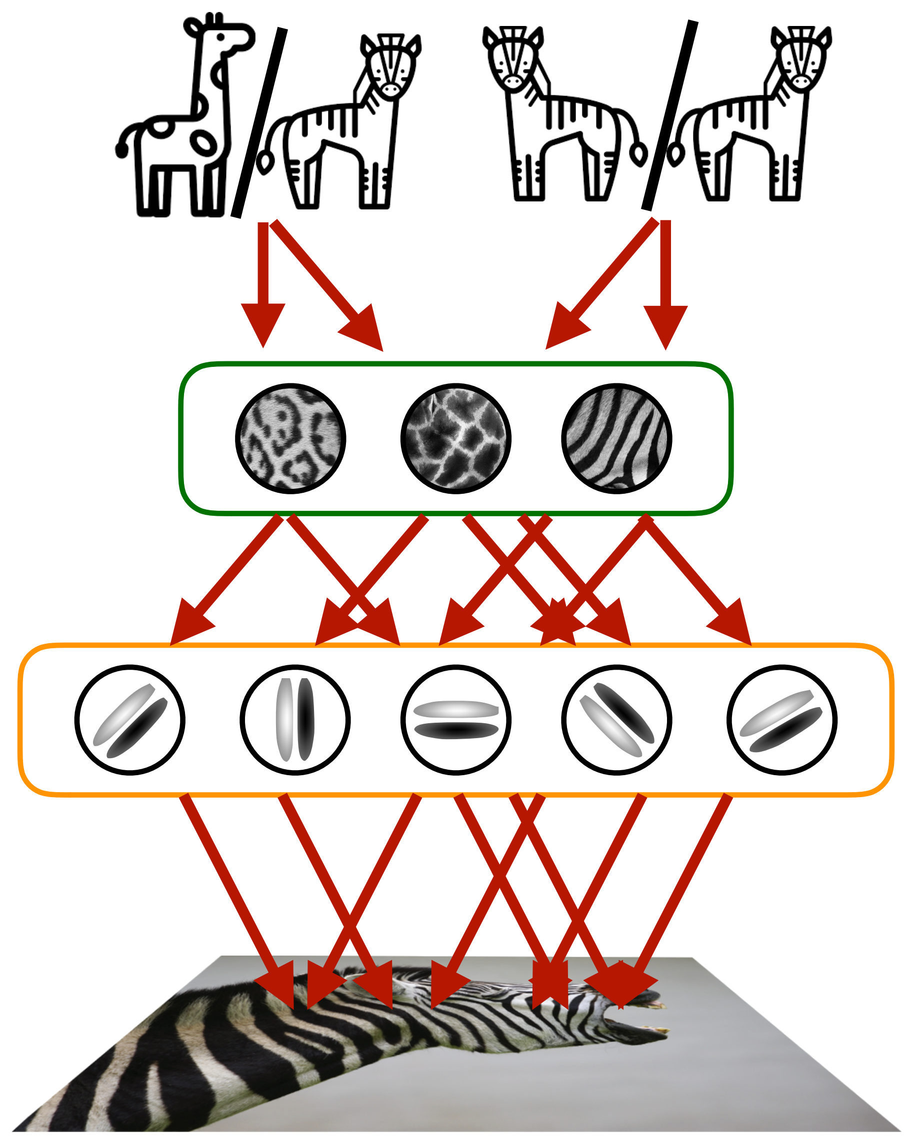
- probabilistic models provide detailed predictions of stimulus statistics
  - most deep learning architectures don't explicitly account for variability
- probabilistic models are explicitly formalised hypotheses about neural computation
  - deep learning models are generic, very flexible computing architectures with no easy interpretation
- in probabilistic models it is often hard to implement inference, and each one is different
  - there are powerful existing methods to train deep learning models
- it is nontrivial to build a generative model if images that performs acceptably in an object recognition task
  - deep learning models do object recognition very well
- they can be combined to yield more powerful predictions





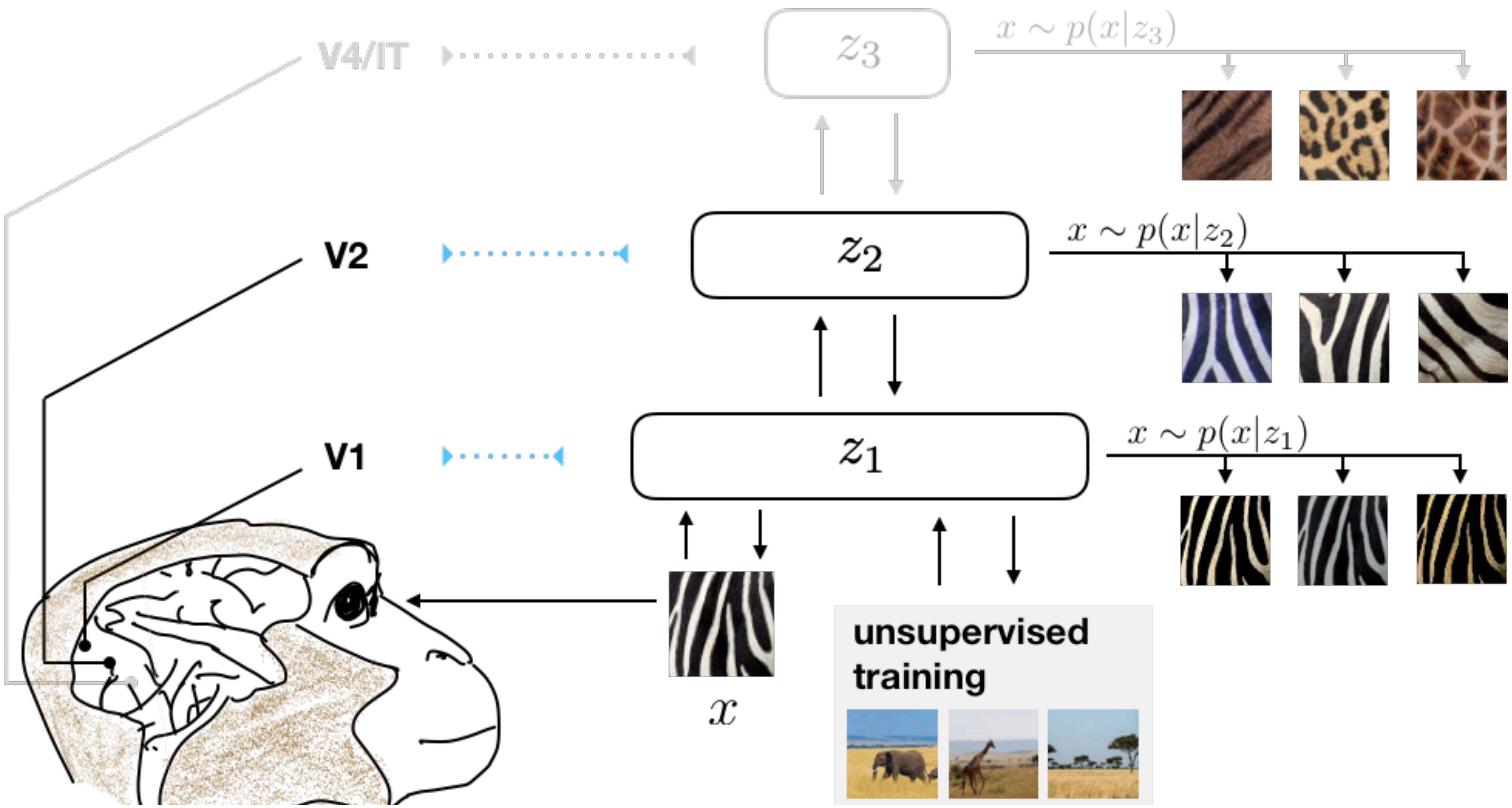




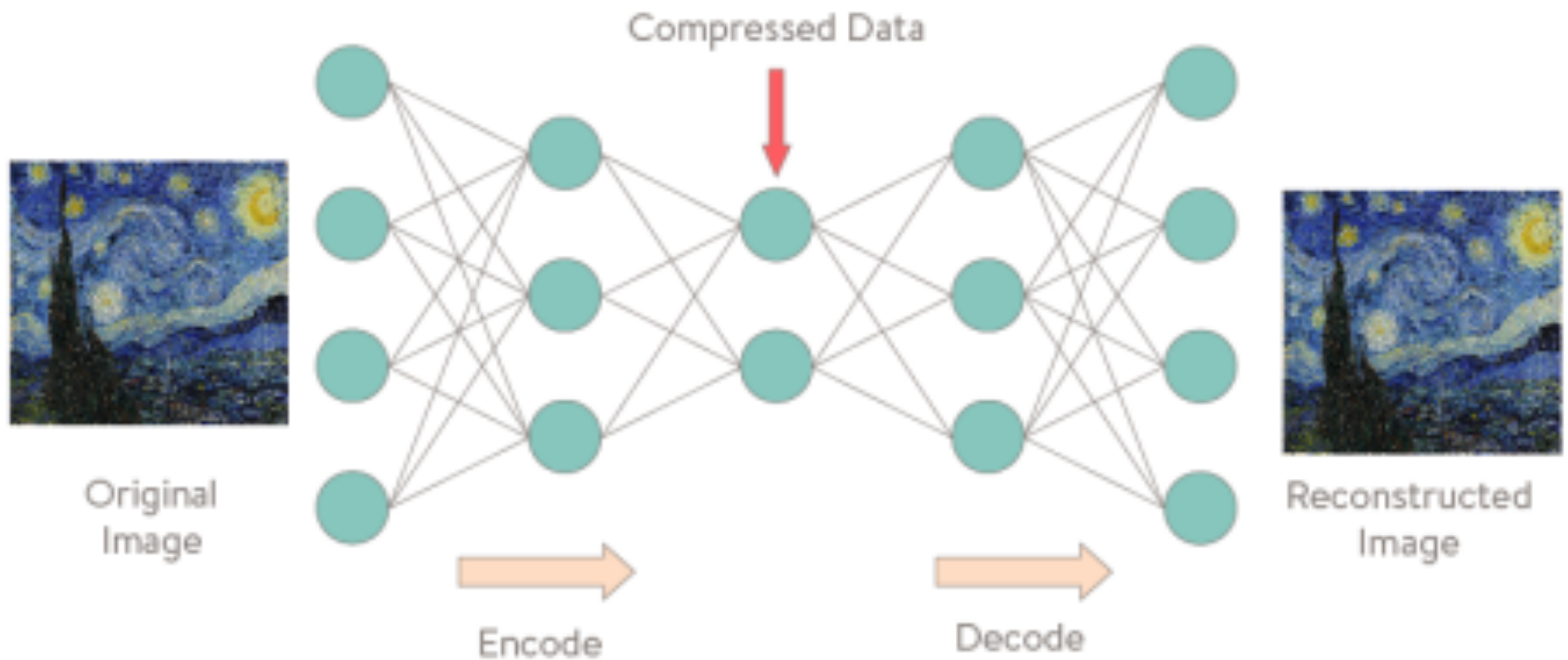


Bányai & Orbán, 2019, Curr Opin Neurobiol

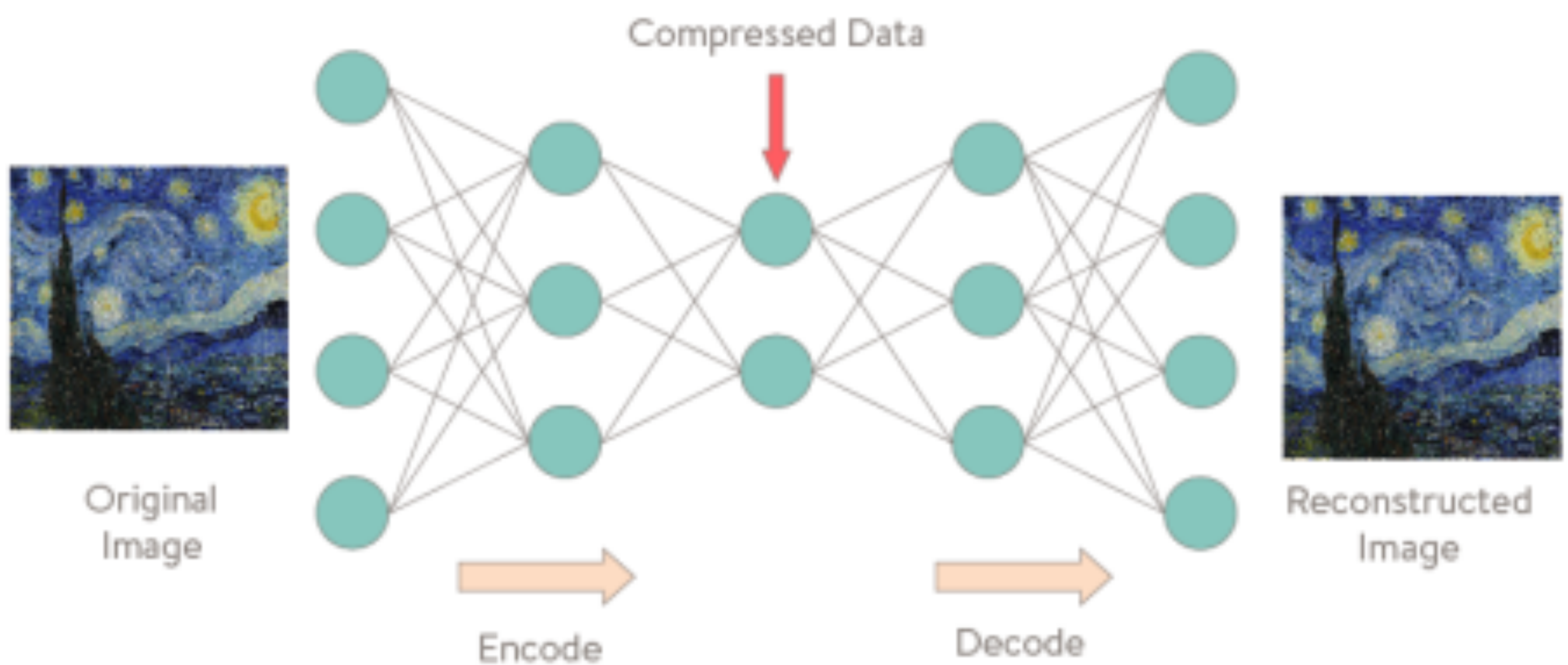
Semantic compression of the visual input

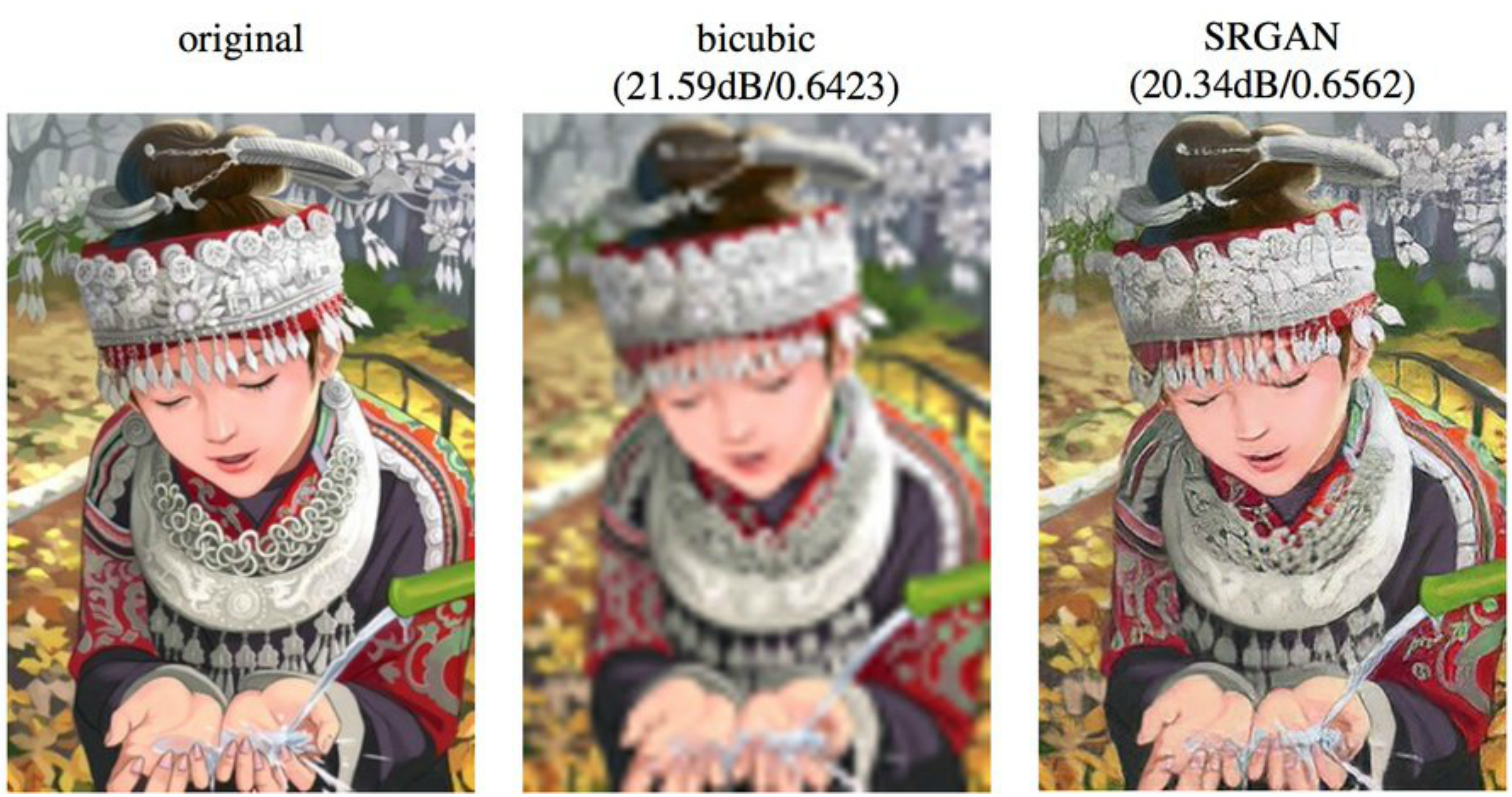


Variational autoencoders and compression

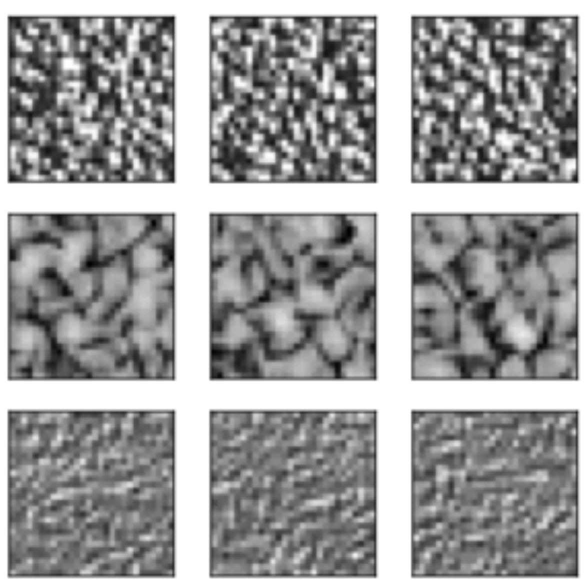


Variational autoencoders and compression

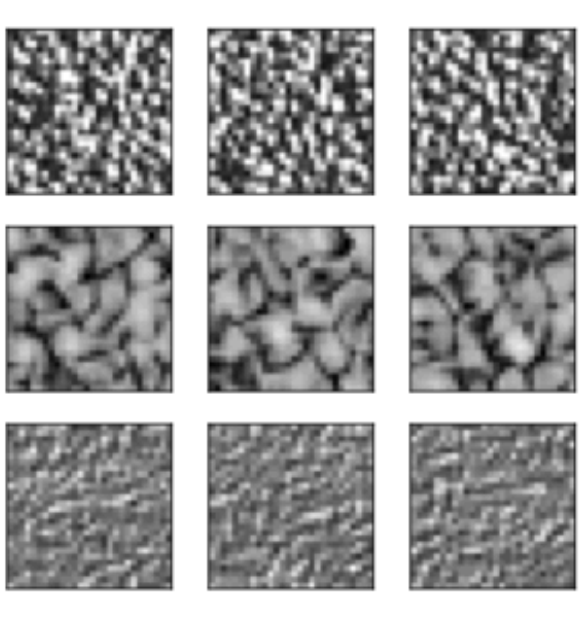


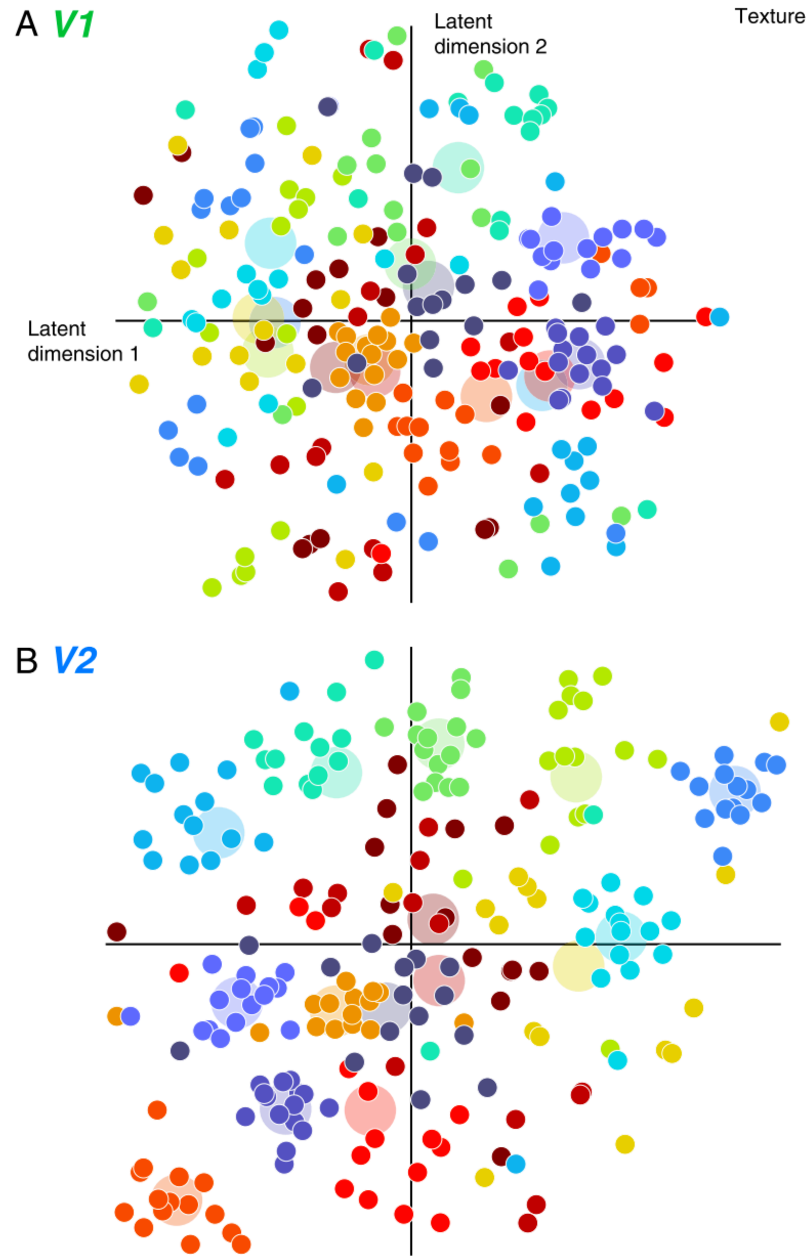


Neural representation of textures



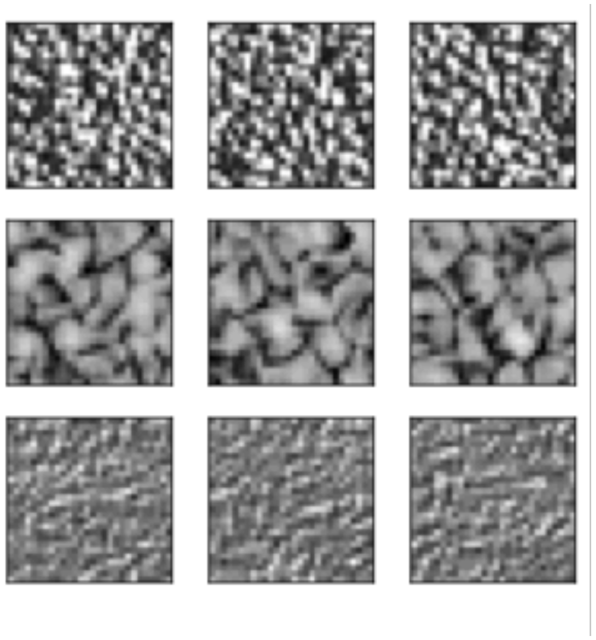
Neural representation of textures

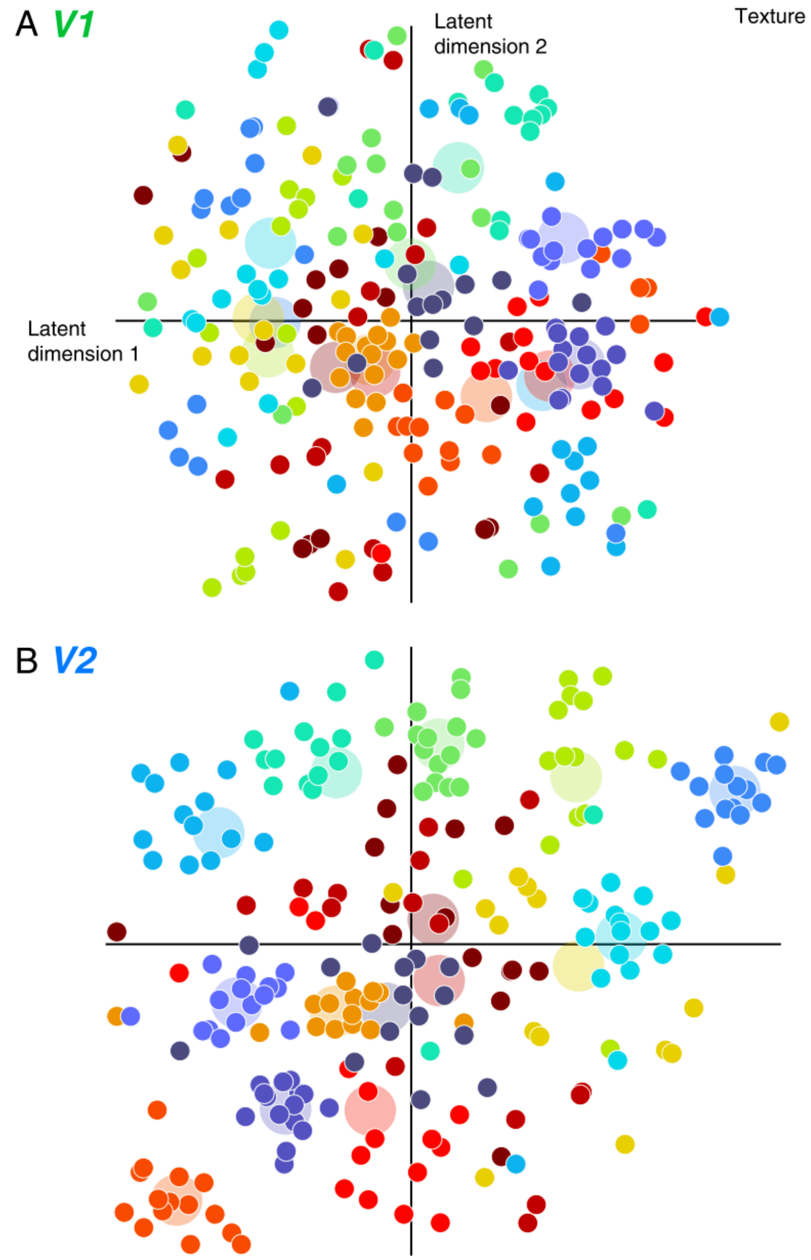


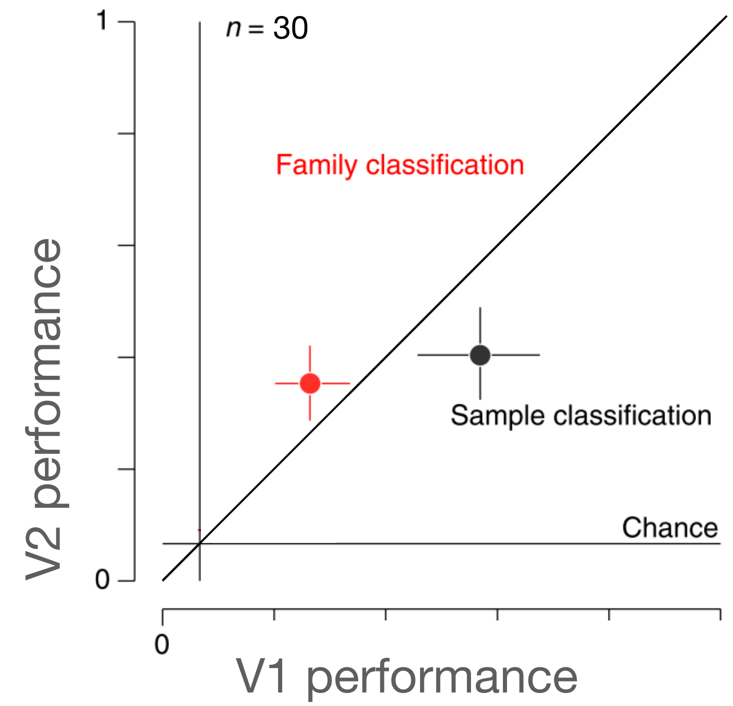


Ziemba et al, 2016, PNAS

Neural representation of textures



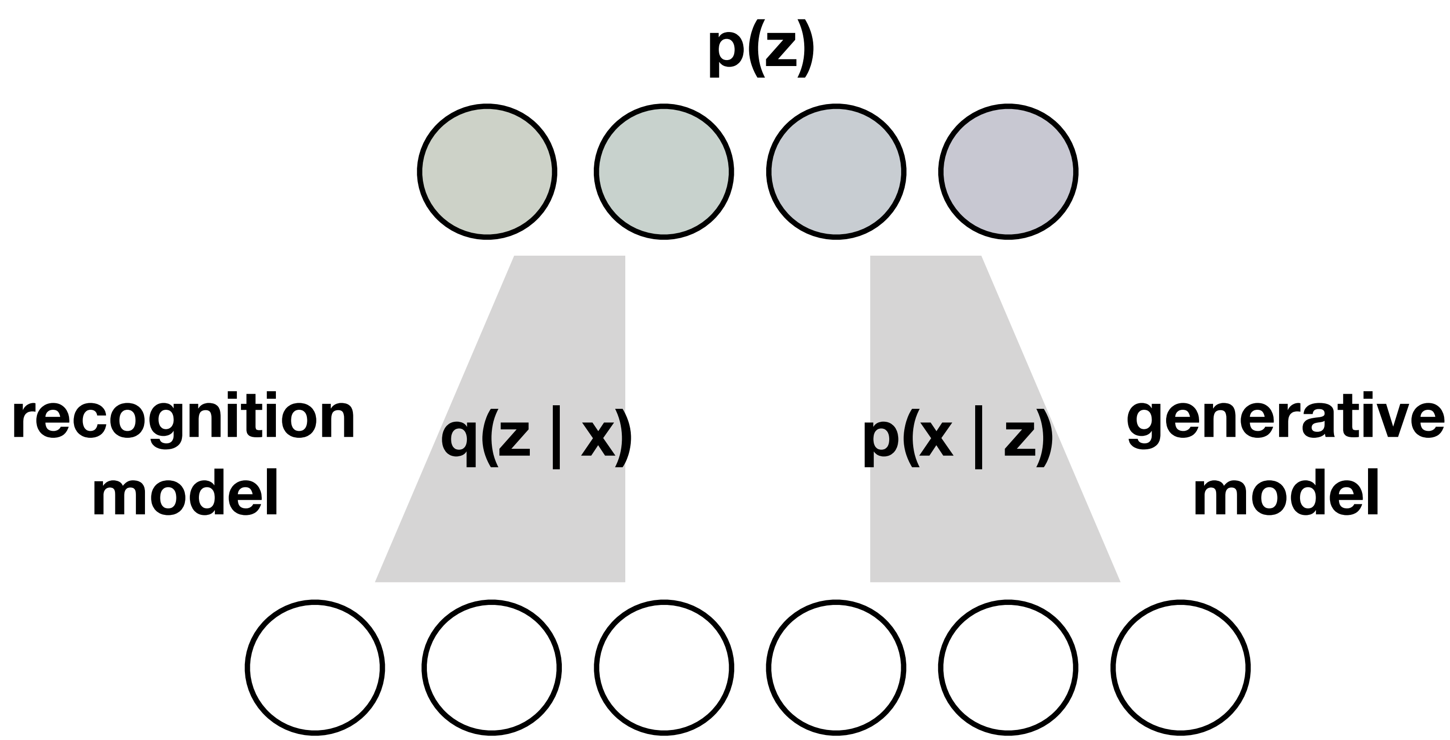




Ziemba et al, 2016, PNAS

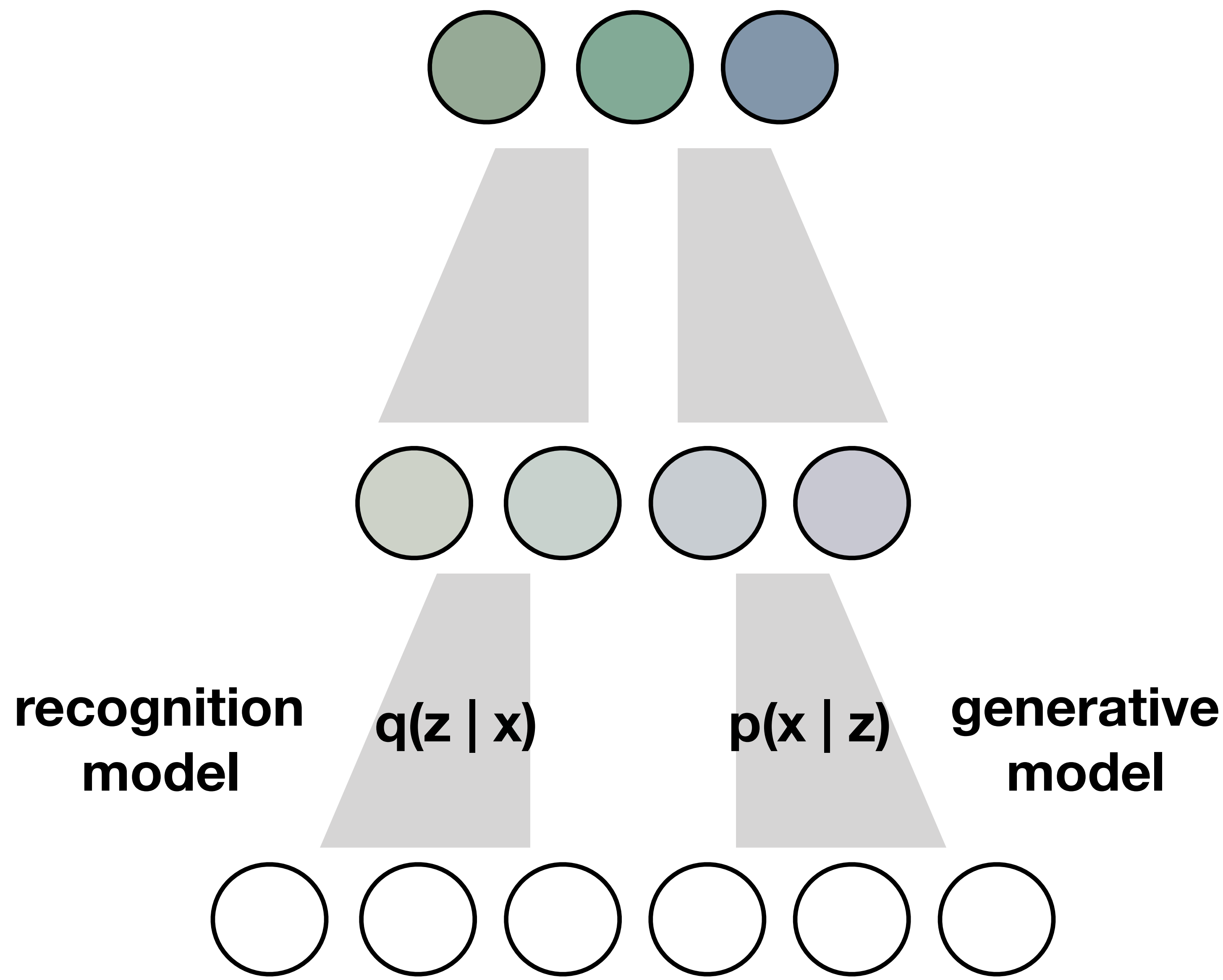
High-dimensional hierarchical generative models

High-dimensional hierarchical generative models



Kingma & Welling, 2013

High-dimensional hierarchical generative models

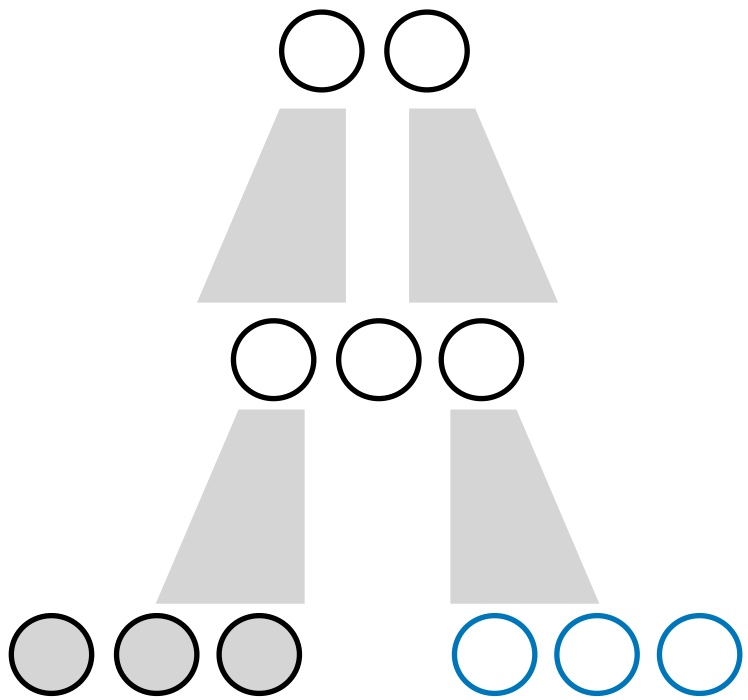


Sønderby et al, 2016

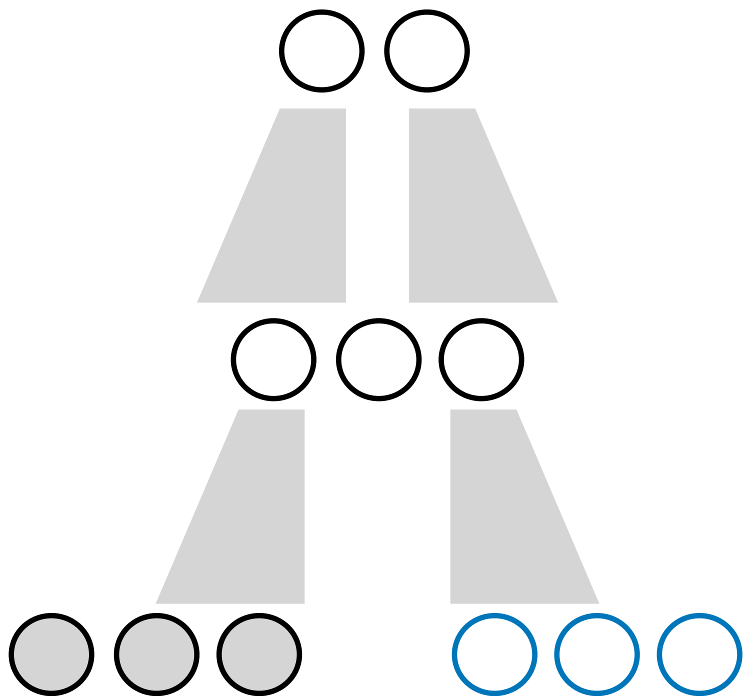
Kingma & Welling, 2013

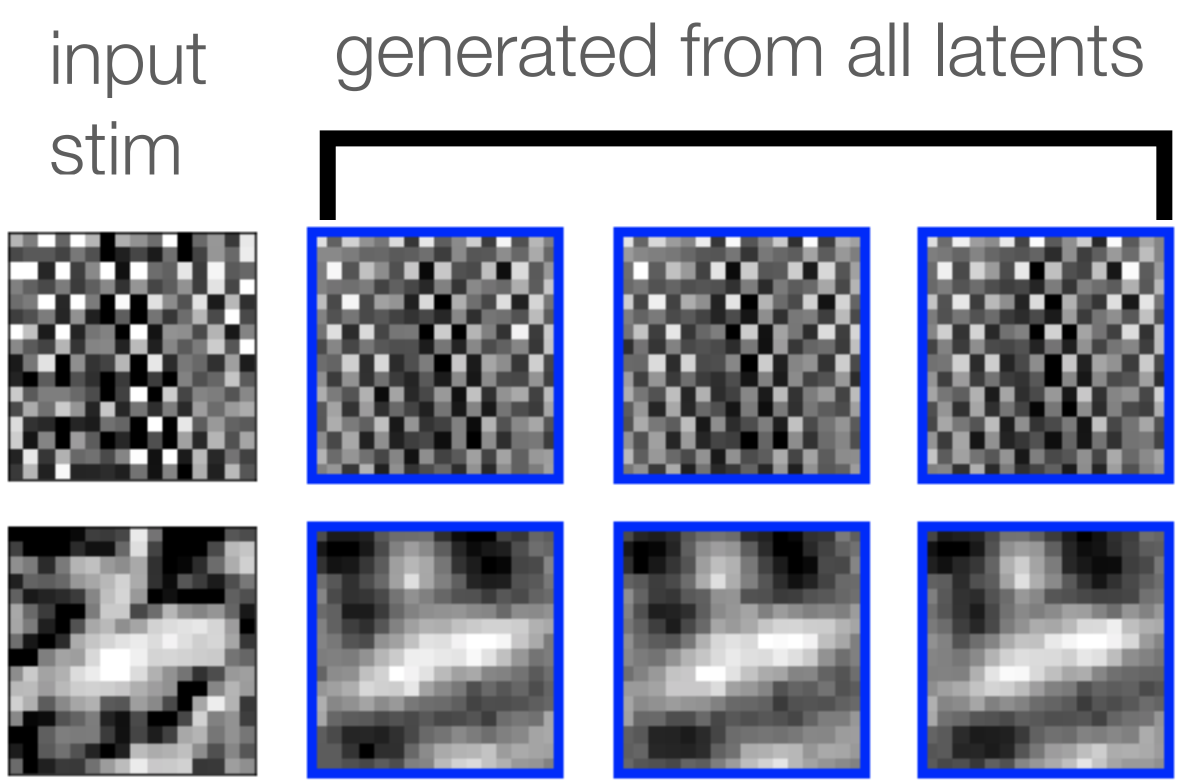
Semantic compression of textures

Semantic compression of textures

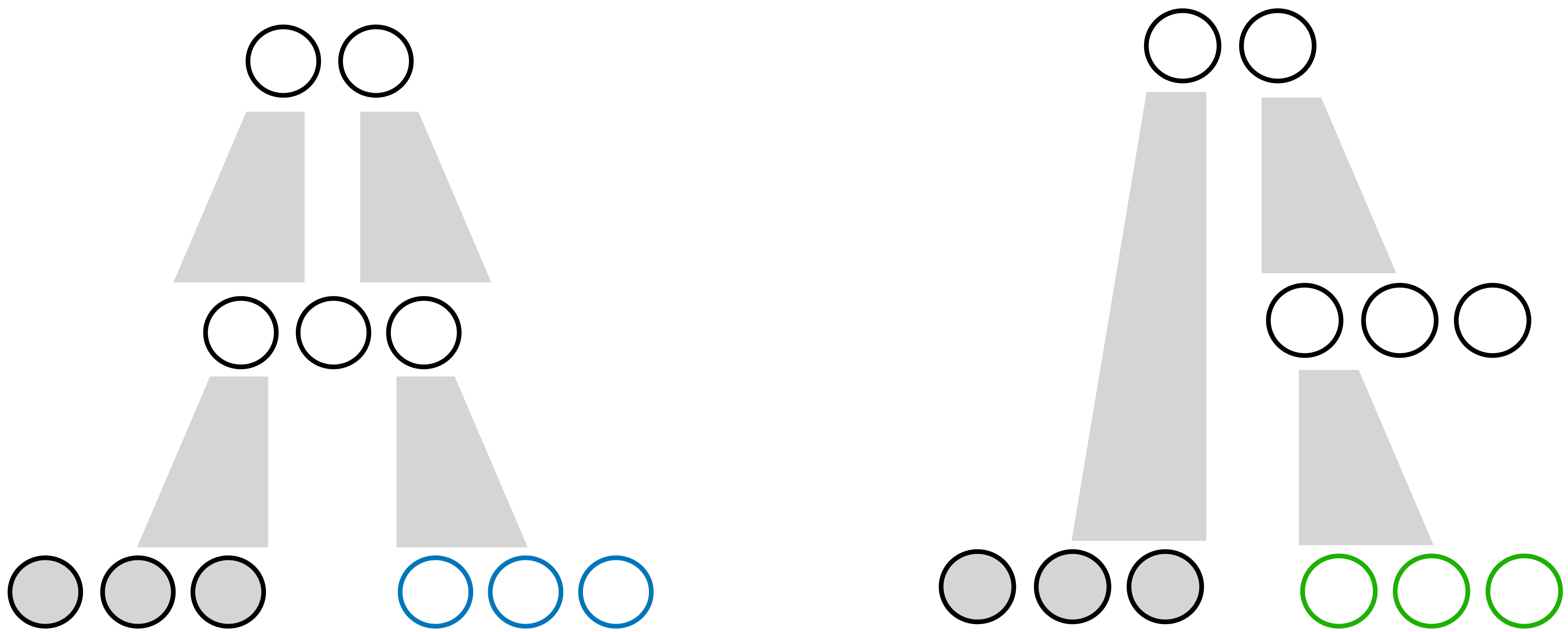


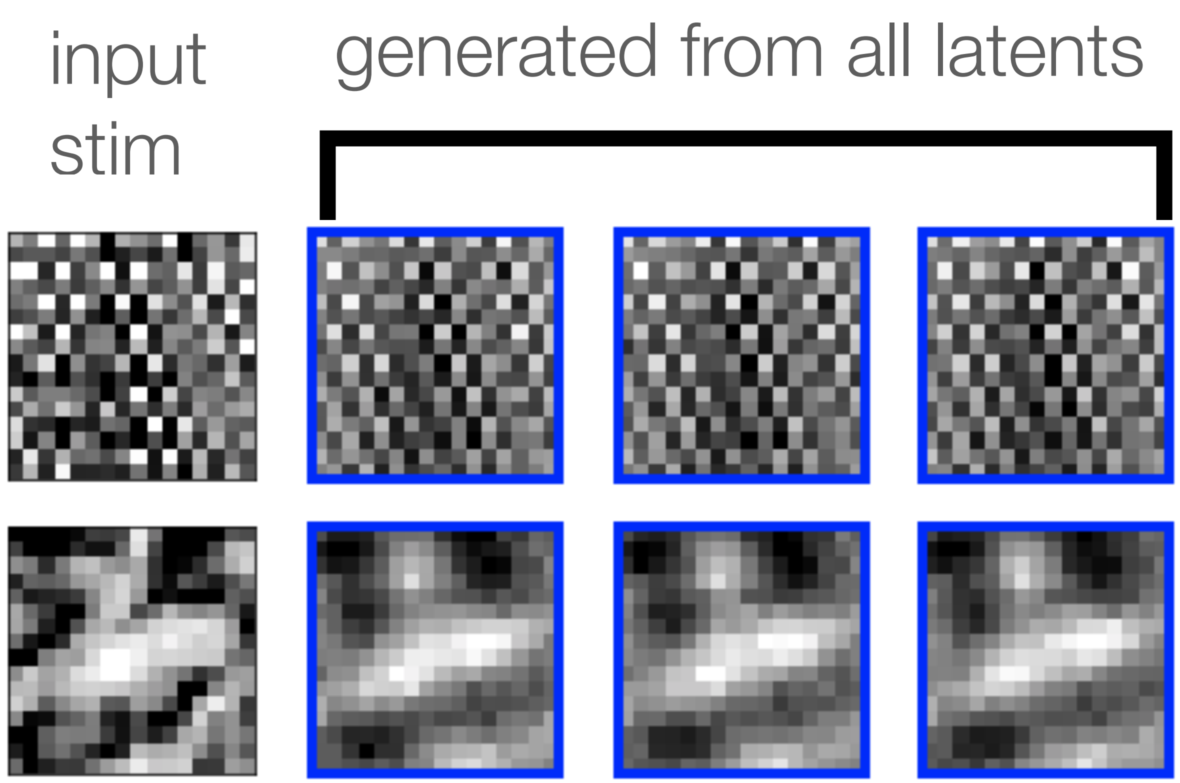
Semantic compression of textures



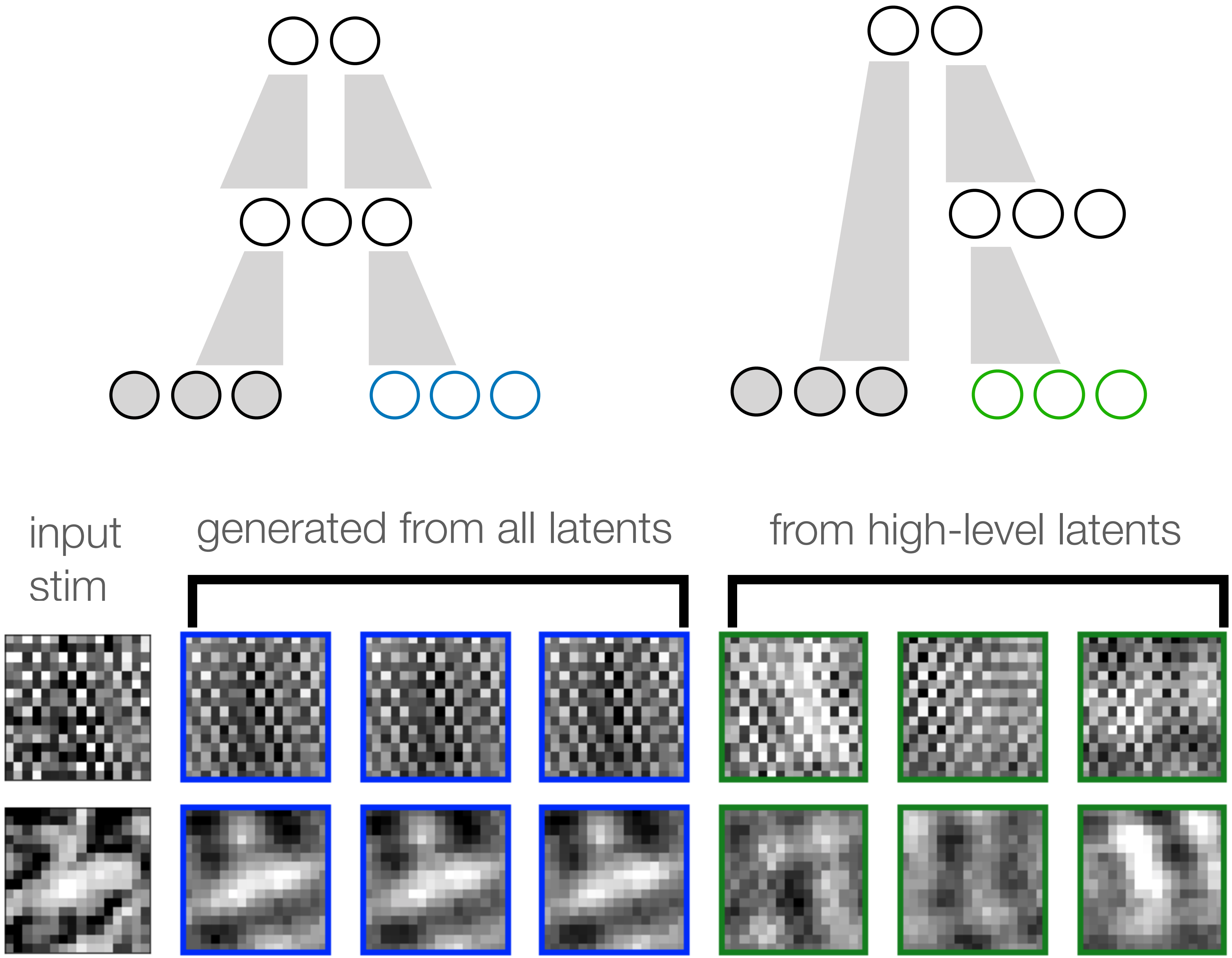


Semantic compression of textures

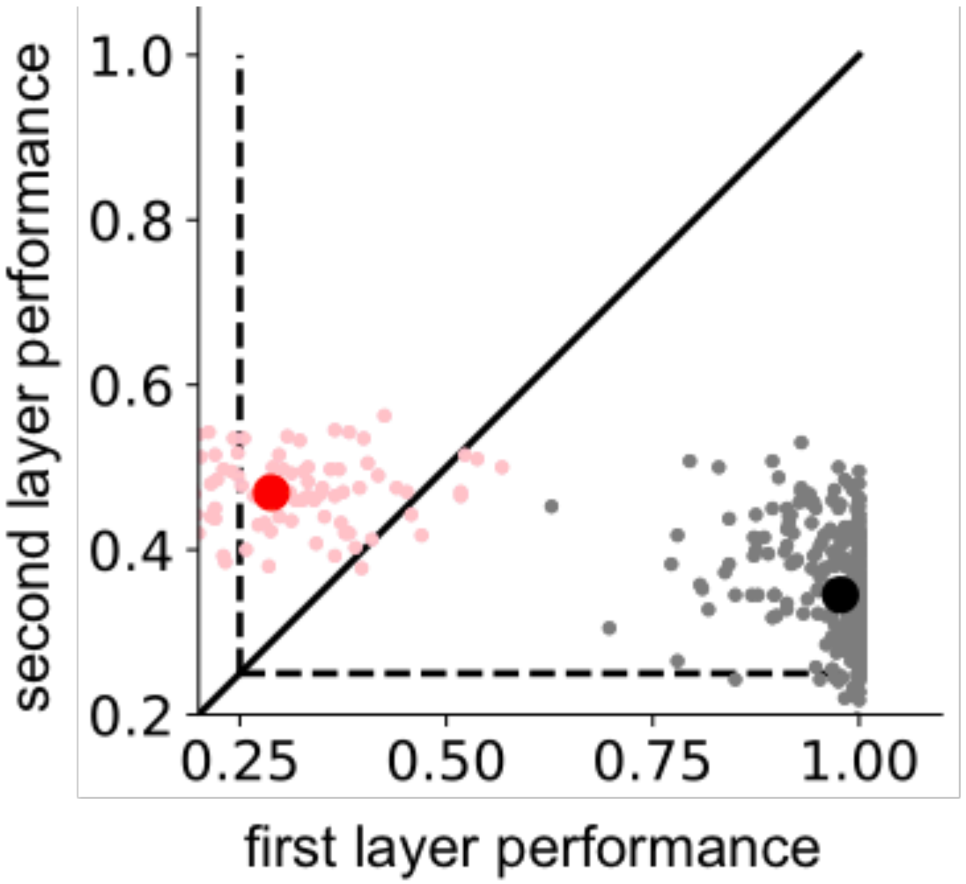




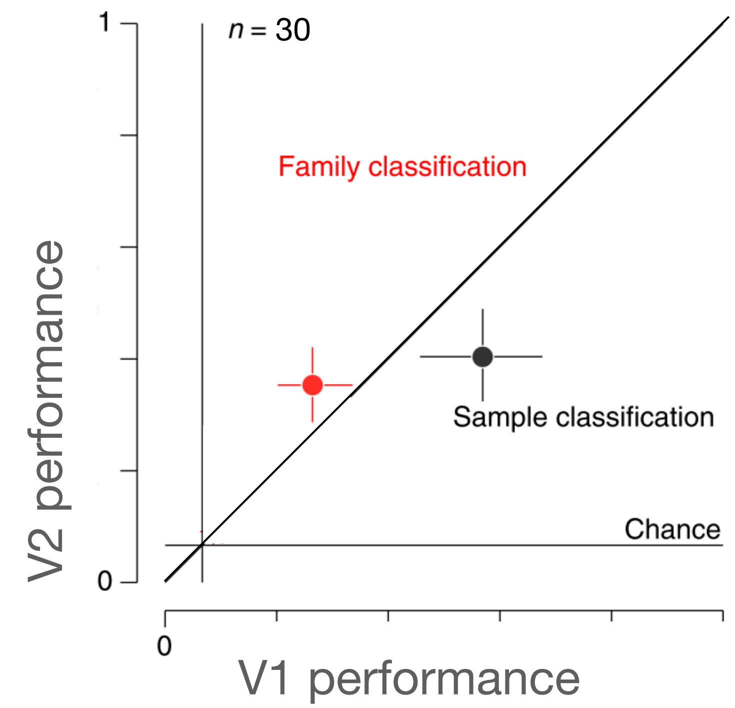
Semantic compression of textures



Representation of texture families in the model



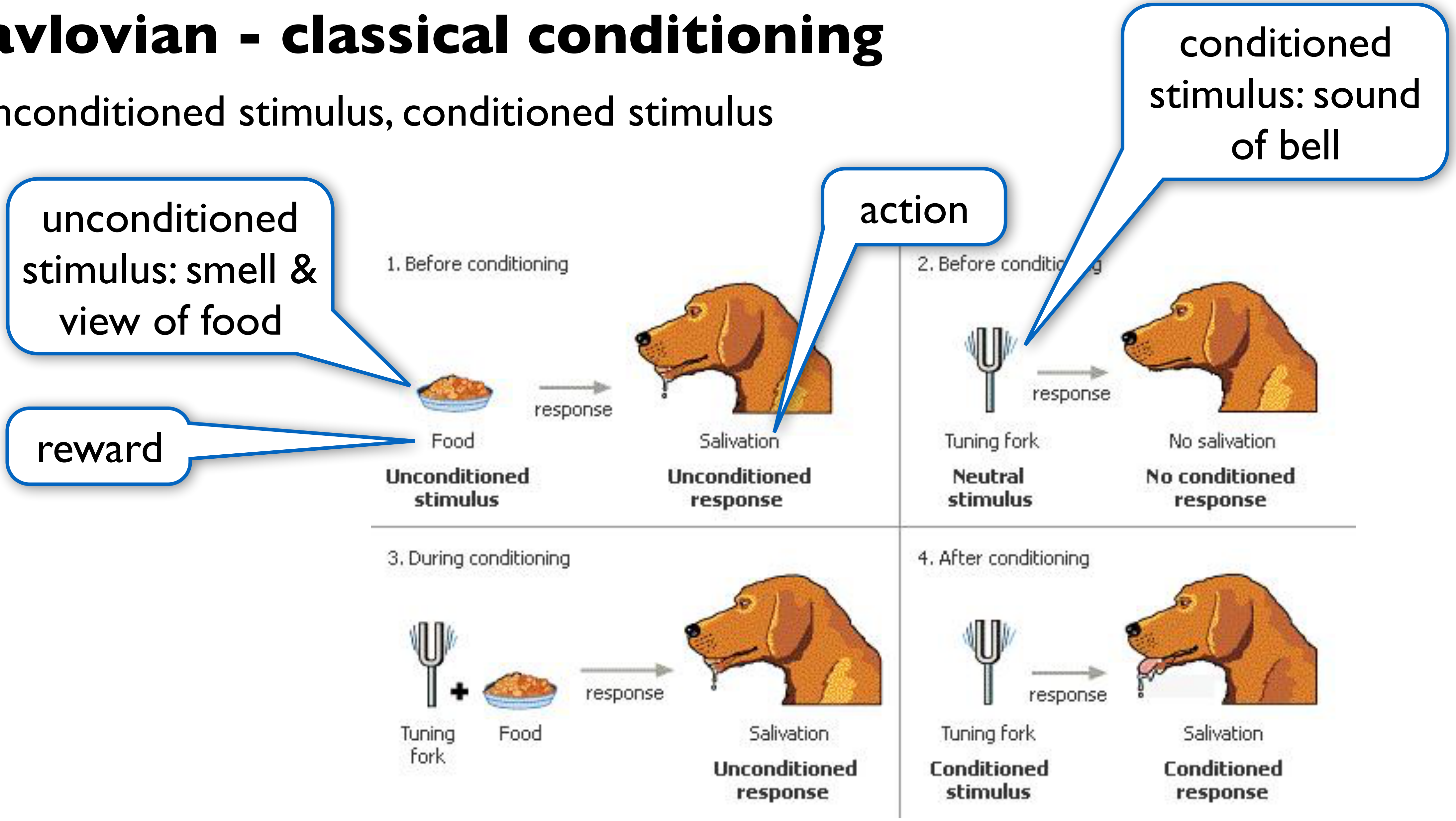
Bányai, Nagy & Orbán, 2019, CCN



Predicting reward

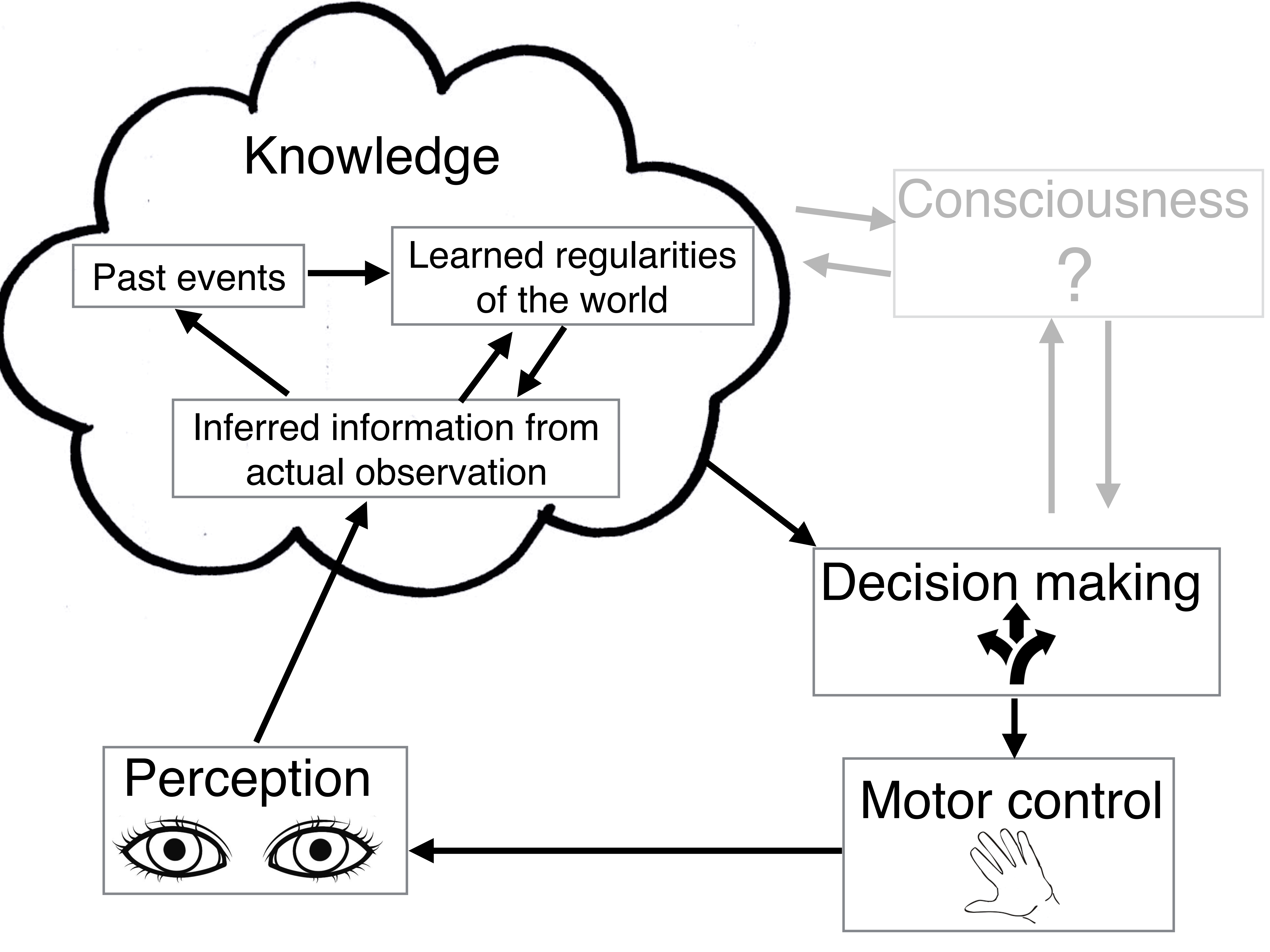
Pavlovian - classical conditioning

• unconditioned stimulus, conditioned stimulus



instrumental - operant conditioning

• the actions of the animal determine the reward

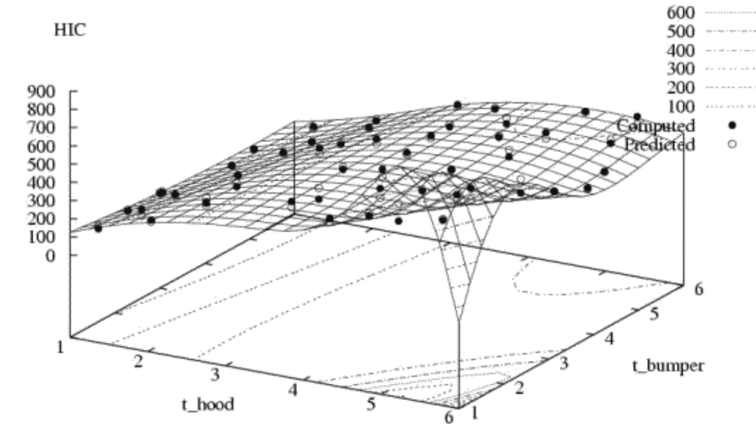

Knowledge
Past events
Learned regularities of the world
Inferred information from actual observation
Consciousness
?
Decision making
Motor control
Perception

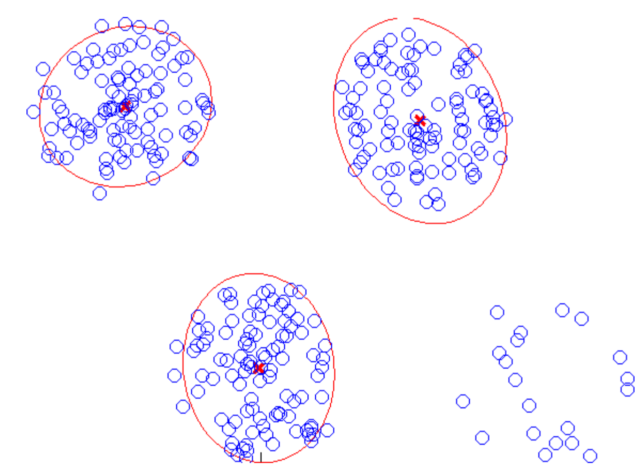
How to build a decision making model on top of the perceptual one?

- We need to choose a target variable from the model that we will try to optimise - this will be called the reward
- Motor output will be modelled a fixed set of possible actions that make modifications in the environment
- A combination of inferred values for the latent variables is called a (perceived) state of the environment
- We have to figure out which action to choose in every state

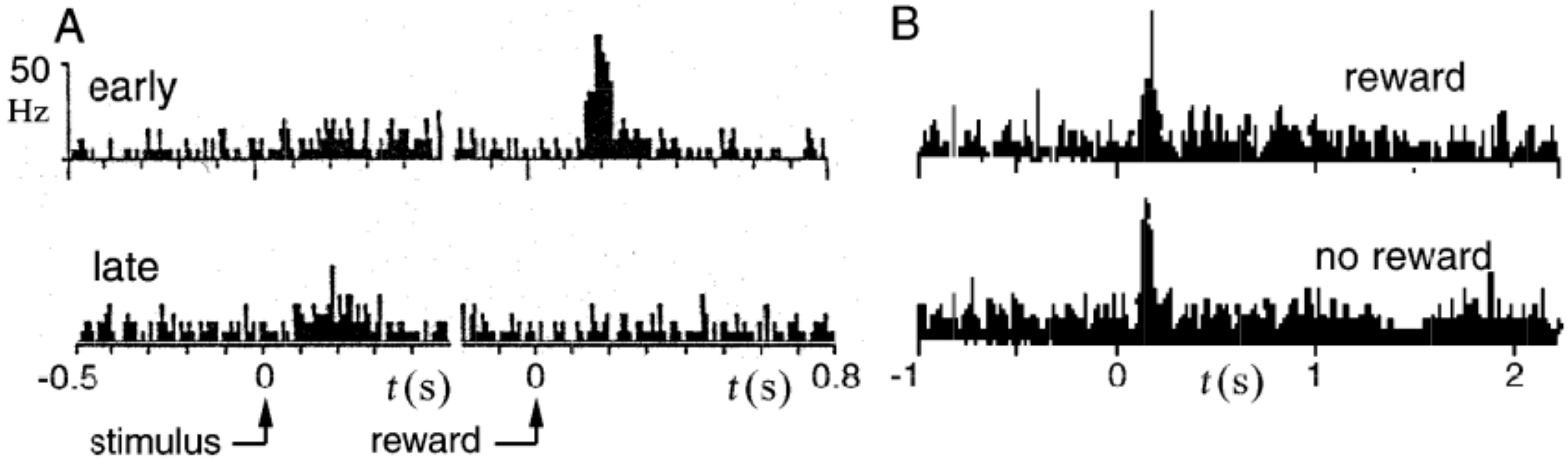
Basic types of learning problems

- Supervised
  - data: input-output pairs
  - approximate the mapping between them
  - discrete output: classification
  - continuous: regression
- Unsupervised
  - data: set of values
  - fit a predefined structure on it
  - find the optimal representation: clustering, filtering
- Reinforcement
  - data: state information and sparse reward
  - learn optimal strategies
  - active learning



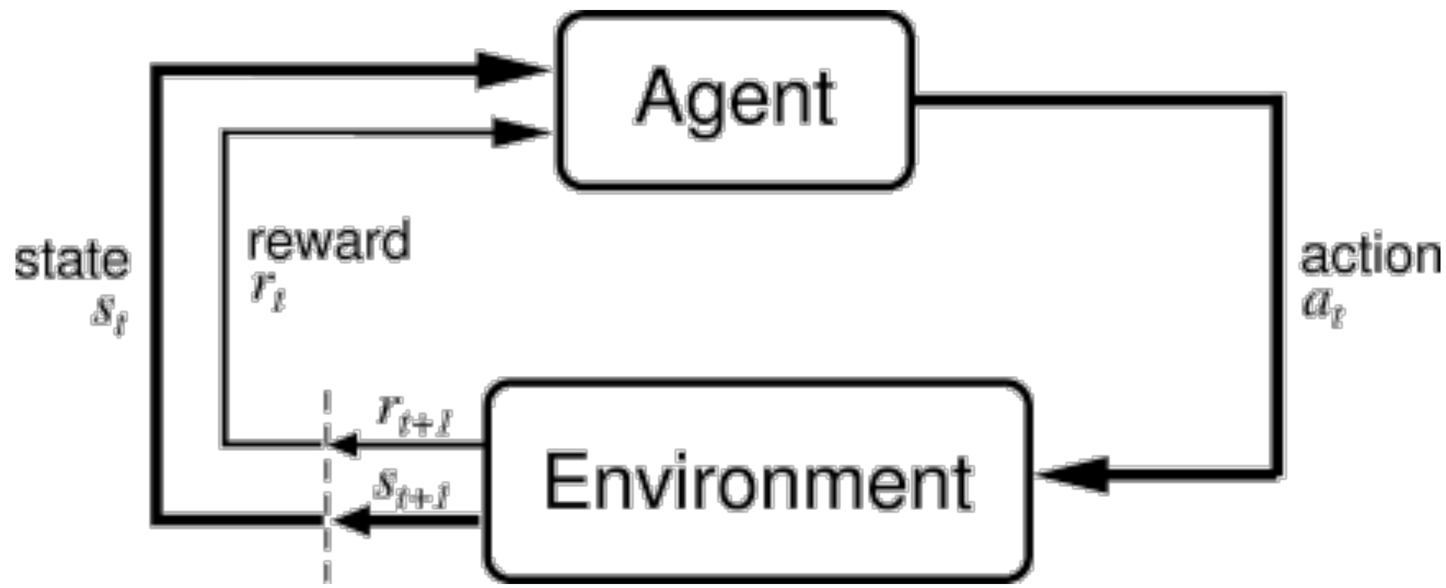


Reward encoding in the cortex



- Dopamin neurons in the monkey's cortex respond according to the learned association between indicator variables and reward
- Activity is proportional to surprise

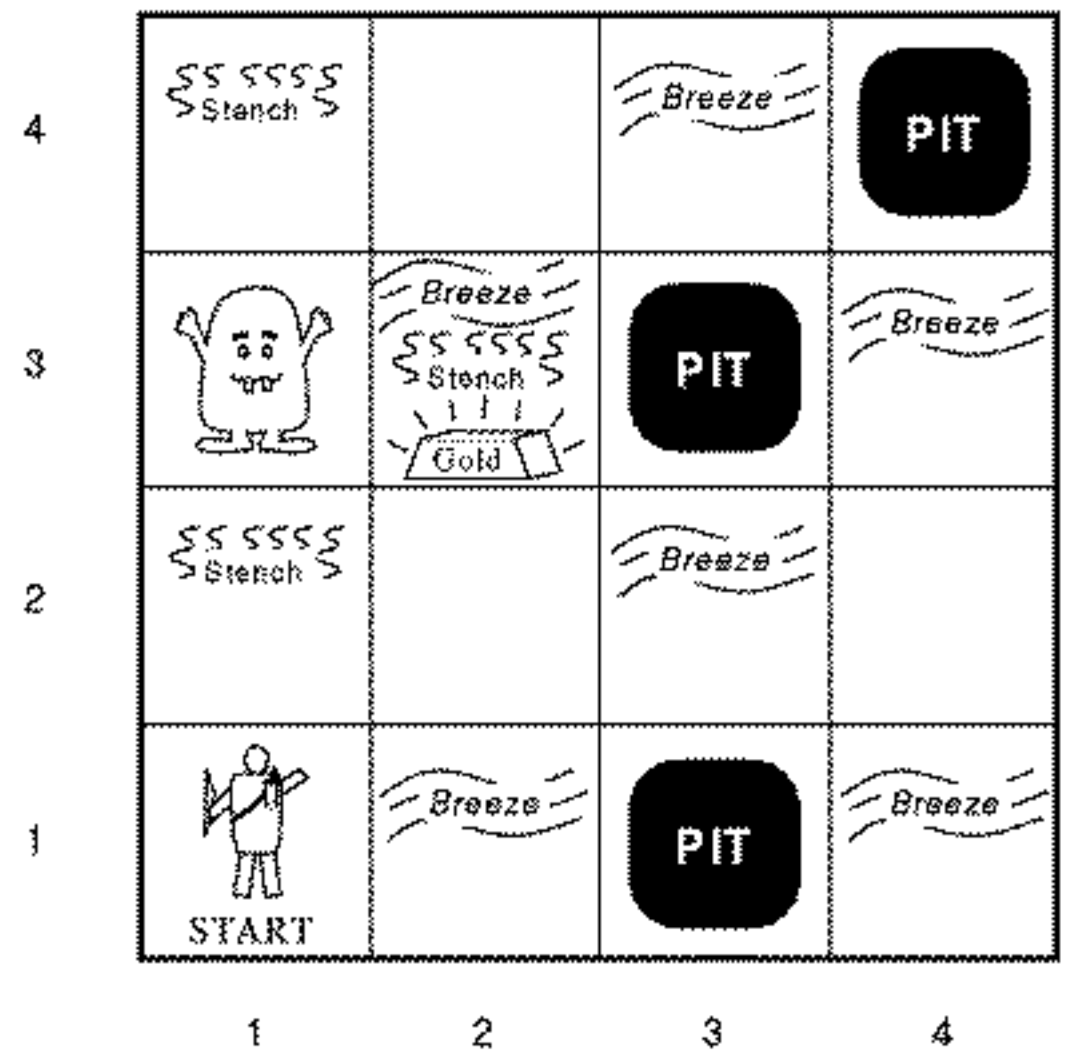
Agent-environment framework



- The environment communicates towards the agent which state it's in
- Reward is given in some states to the agent
- The agent pushes the environment to new states by its actions

Learning the value of states

- Simplest case: there is a finite number of states of the environment
- each state *s* is a combination of inferred values for the latent variables of the mental model (e.g. we take the *a posteriori* most probable values)
- for each state we assign a value, *V(s)*, that encodes the desirability of that state
- after each decision, at time *t*, we update the estimation of state values using previous estimations and the reward
- intuition: a value of a state is determined by the reward, and the value of other states that are accessible from it through a few actions



Russell & Norvig, 1995

https://cs.stanford.edu/people/karpathy/reinforcejs/gridworld_td.html

Reinforcement learning

- The goal of RL is to maximize reward in the long run
- We have to learn how useful certain states and actions are to do that
- Trial and error
- Set values based on reward
- Propagate value to states without reward

Temporal Difference learning

- We learn from prediction error
- The value of the state we stepped on makes the previously visited state more similar to itself
- We can propagate to earlier states too

$$V(S_t) \leftarrow \underbrace{V(S_t)}_{\text{Previous estimate}} + \alpha[\underbrace{\underbrace{R_{t+1}}_{\text{Reward t+1}} + \underbrace{\gamma V(S_{t+1})}_{\text{Discounted value on the next step}}}_{\text{TD Target}} - V(S_t)]$$

Action selection

Action selection

- Now we have the state value function

Action selection

- Now we have the state value function
  - V: s -> R

Action selection

- Now we have the state value function
    - V: s -> R
- In order to choose an action, we need to know that from the current state, which action leads to which other state

Action selection

- Now we have the state value function
  - V: s -> R
- In order to choose an action, we need to know that from the current state, which action leads to which other state
  - M: (s,a) -> s

Action selection

- Now we have the state value function
  - V: s -> R
- In order to choose an action, we need to know that from the current state, which action leads to which other state
  - M: (s,a) -> s
  - or more generally, M: (s1,s2,a) -> P

Action selection

- Now we have the state value function
    - V: s -> R
- In order to choose an action, we need to know that from the current state, which action leads to which other state
    - M: (s,a) -> s
    - or more generally, M: (s1,s2,a) -> P
    - this function, M, is called the **model** of the environment

Action selection

- Now we have the state value function
    - V: s -> R
- In order to choose an action, we need to know that from the current state, which action leads to which other state
    - M: (s,a) -> s
    - or more generally, M: (s1,s2,a) -> P
    - this function, M, is called the **model** of the environment
- In each RL setting in which we learn a state-value function, we have to learn a model function as well

Action selection

- Now we have the state value function
  - V: s -> R
- In order to choose an action, we need to know that from the current state, which action leads to which other state
  - M: (s,a) -> s
  - or more generally, M: (s1,s2,a) -> P
  - this function, M, is called the **model** of the environment
- In each RL setting in which we learn a state-value function, we have to learn a model function as well
  - these are called **model-based** solutions

Model-free RL

Model-free RL

- Instead of learning the model of the environment, we can learn the value of state-action pairs directly

Model-free RL

- Instead of learning the model of the environment, we can learn the value of state-action pairs directly
  - the function Q: (s,a) -> R

Model-free RL

- Instead of learning the model of the environment, we can learn the value of state-action pairs directly
  - the function Q: (s,a) -> R
  - the TD rule applies nicely here as well

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \epsilon \left[r_{t+1} + \gamma \max_a Q_t(s_t + 1, a) - Q_t(s_t, a_t) \right]$$

Model-free RL

- Instead of learning the model of the environment, we can learn the value of state-action pairs directly
  - the function Q: (s,a) -> R
  - the TD rule applies nicely here as well
  - action selection only requires the Q function, not the model

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \epsilon \left[r_{t+1} + \gamma \max_a Q_t(s_t + 1, a) - Q_t(s_t, a_t) \right]$$

Deliberative and reactive systems

- The Q function has (#states x #actions) parameters, while the V only has (#states)

Deliberative and reactive systems

- The Q function has (#states x #actions) parameters, while the V only has (#states)
  - it requires much more observations to estimate Q reliably than it does to estimate V

Deliberative and reactive systems

- The Q function has (#states x #actions) parameters, while the V only has (#states)
  - it requires much more observations to estimate Q reliably than it does to estimate V
- The learned model can be reused in a different task, a model-free agent has to start from scratch

Deliberative and reactive systems

- The Q function has (#states x #actions) parameters, while the V only has (#states)
    - it requires much more observations to estimate Q reliably than it does to estimate V
- The learned model can be reused in a different task, a model-free agent has to start from scratch
- In model-based solutions, action-selection takes a lot more computation

Deliberative and reactive systems

- The Q function has (#states x #actions) parameters, while the V only has (#states)
    - it requires much more observations to estimate Q reliably than it does to estimate V
- The learned model can be reused in a different task, a model-free agent has to start from scratch
- In model-based solutions, action-selection takes a lot more computation
    - but you can make long-term plans using the model

Deliberative and reactive systems

- The Q function has (#states x #actions) parameters, while the V only has (#states)
  - it requires much more observations to estimate Q reliably than it does to estimate V
- The learned model can be reused in a different task, a model-free agent has to start from scratch
- In model-based solutions, action-selection takes a lot more computation
  - but you can make long-term plans using the model
  - model-free solutions can be regarded as a bunch reflexes

Deliberative and reactive systems

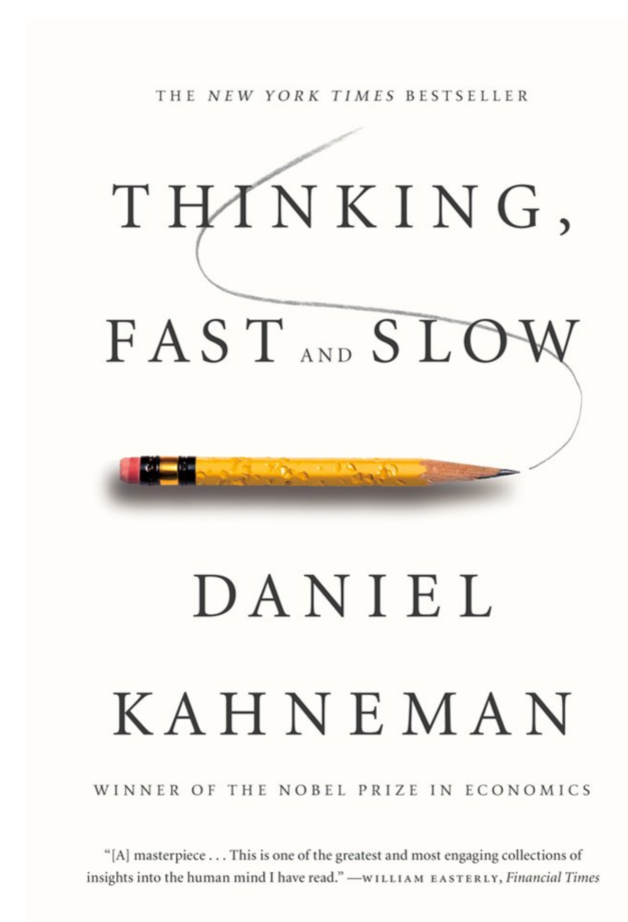
- The Q function has (#states x #actions) parameters, while the V only has (#states)
  - it requires much more observations to estimate Q reliably than it does to estimate V
- The learned model can be reused in a different task, a model-free agent has to start from scratch
- In model-based solutions, action-selection takes a lot more computation
  - but you can make long-term plans using the model
  - model-free solutions can be regarded as a bunch reflexes
  - the two systems can complement each other at different stages of proficiency

Deliberative and reactive systems

- The Q function has (#states x #actions) parameters, while the V only has (#states)
    - it requires much more observations to estimate Q reliably than it does to estimate V
- The learned model can be reused in a different task, a model-free agent has to start from scratch
- In model-based solutions, action-selection takes a lot more computation
    - but you can make long-term plans using the model
    - model-free solutions can be regarded as a bunch reflexes
    - the two systems can complement each other at different stages of proficiency

Deliberative and reactive systems

- The Q function has (#states x #actions) parameters, while the V only has (#states)
    - it requires much more observations to estimate Q reliably than it does to estimate V
- The learned model can be reused in a different task, a model-free agent has to start from scratch
- In model-based solutions, action-selection takes a lot more computation
    - but you can make long-term plans using the model
    - model-free solutions can be regarded as a bunch reflexes
    - the two systems can complement each other at different stages of proficiency



Exploration vs. exploitation

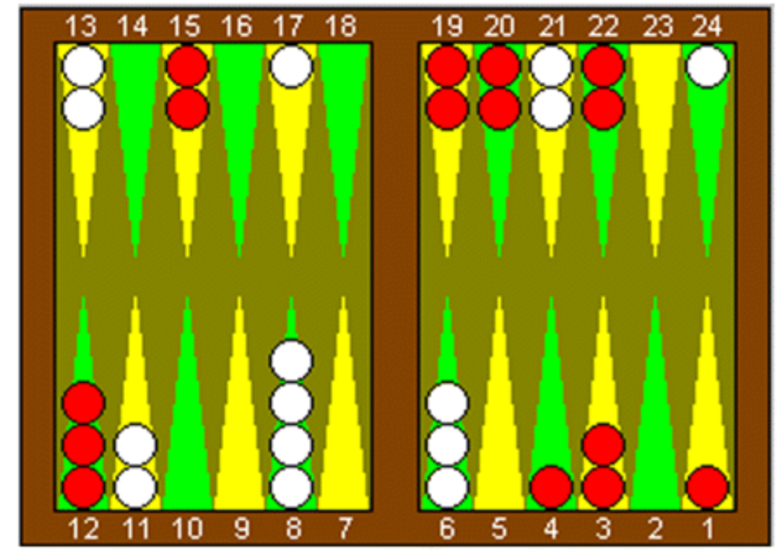
- When we don’t know anything about the environment yet, it doesn’t make sense to repeat the first series of actions that led to some reward
- When we know more, we can just use the best strategy we found
- Usual way: act randomly in the beginning, gradually increase of probability of choosing the action we think is best

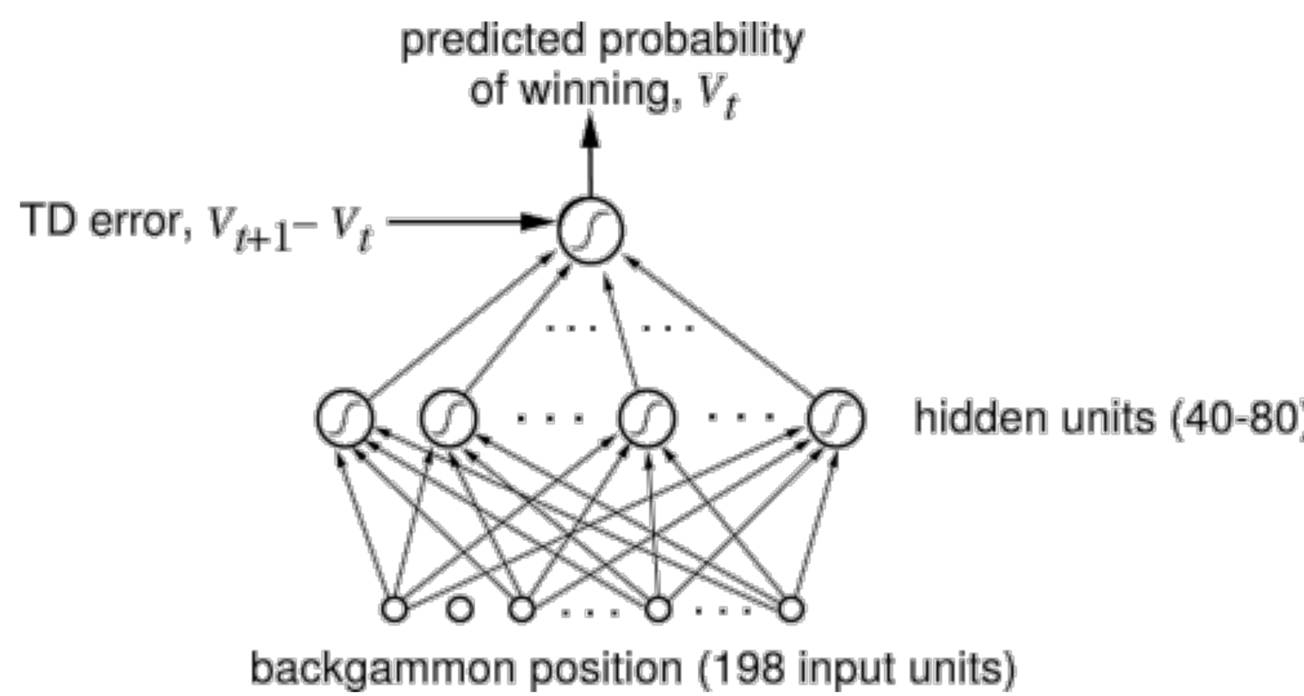
Representation of the value function

- If there are not so many, we can use a table
- With large and continuous spaces
  - we can only represent state variables
  - we need to generalise to states never visited
  - feedforward neural networks are a good choice
  - we have to construct a desired output for backpropagation at each step from the prediction error

TD learning with a neural network

- Gerald Tesauro: TD-Gammon
  - feed-forward neural network
  - Input: states attainable by possible actions
  - Output: state value (winning probability)
- At each step, we have to calculate an output error for the network
  - based on the reward signal
- Result: comparable to best human players
- Total training time today: 5s





TD using a neural representation

- Using the prediction error for learning
- Update of the state value in the neural representation:

$$w(\tau) \leftarrow w(\tau) + \varepsilon\delta(t)u(t-\tau) \qquad \delta(t) = \sum_t r(t+\tau) - v(t)$$

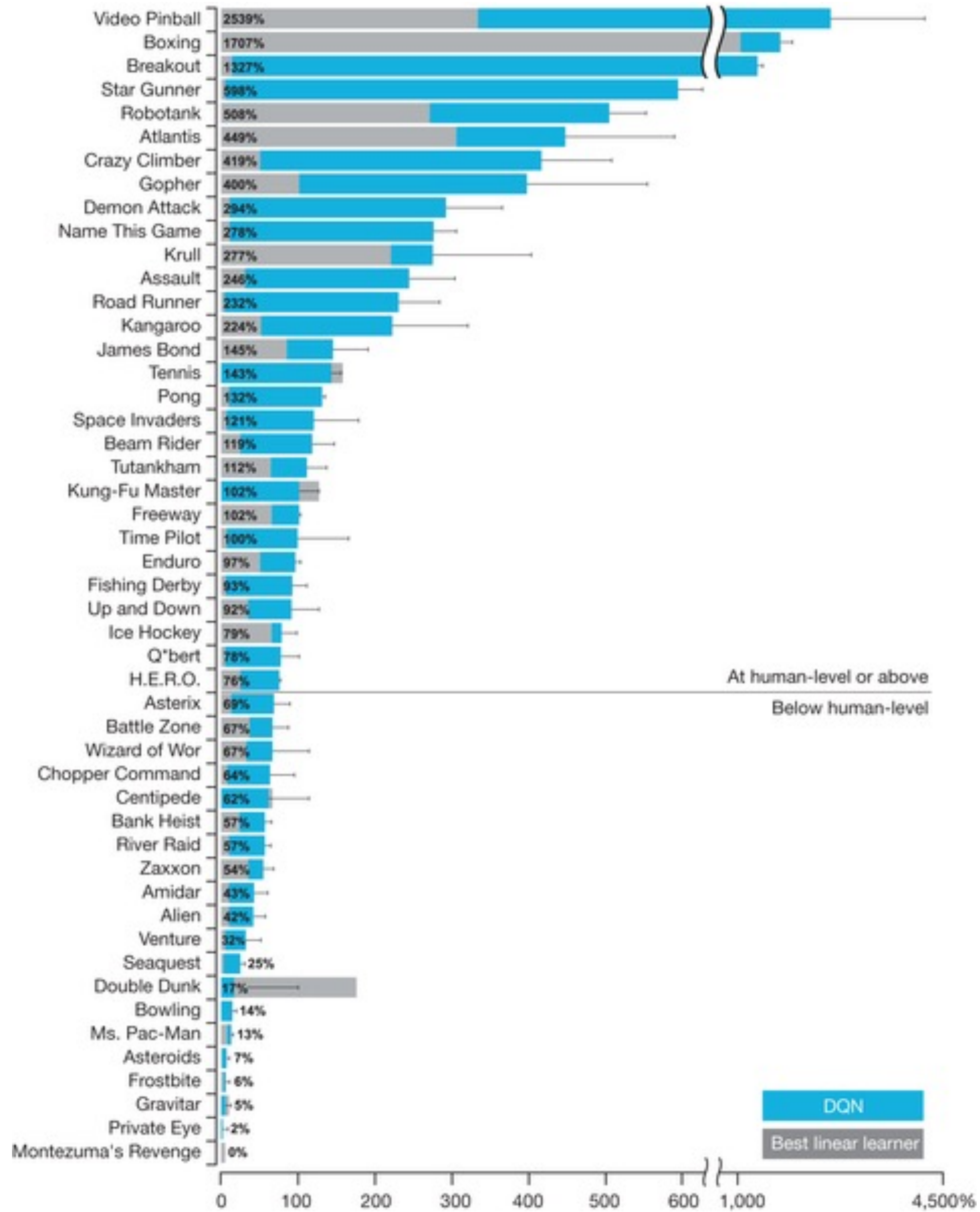
- Calculating the prediction error
  - Ideally we'd need the sum of future rewards
  - We use a single-step local approximation

$$\sum_t r(t+\tau) - v(t) \approx r(t) + v(t+1)$$

  - If the environment is observable, this converges to an optimal strategy
- We can propagate the error back to previous states too

Learning to play computer games

- reinforcement learning combined with deep networks
- observation: computer screen & score

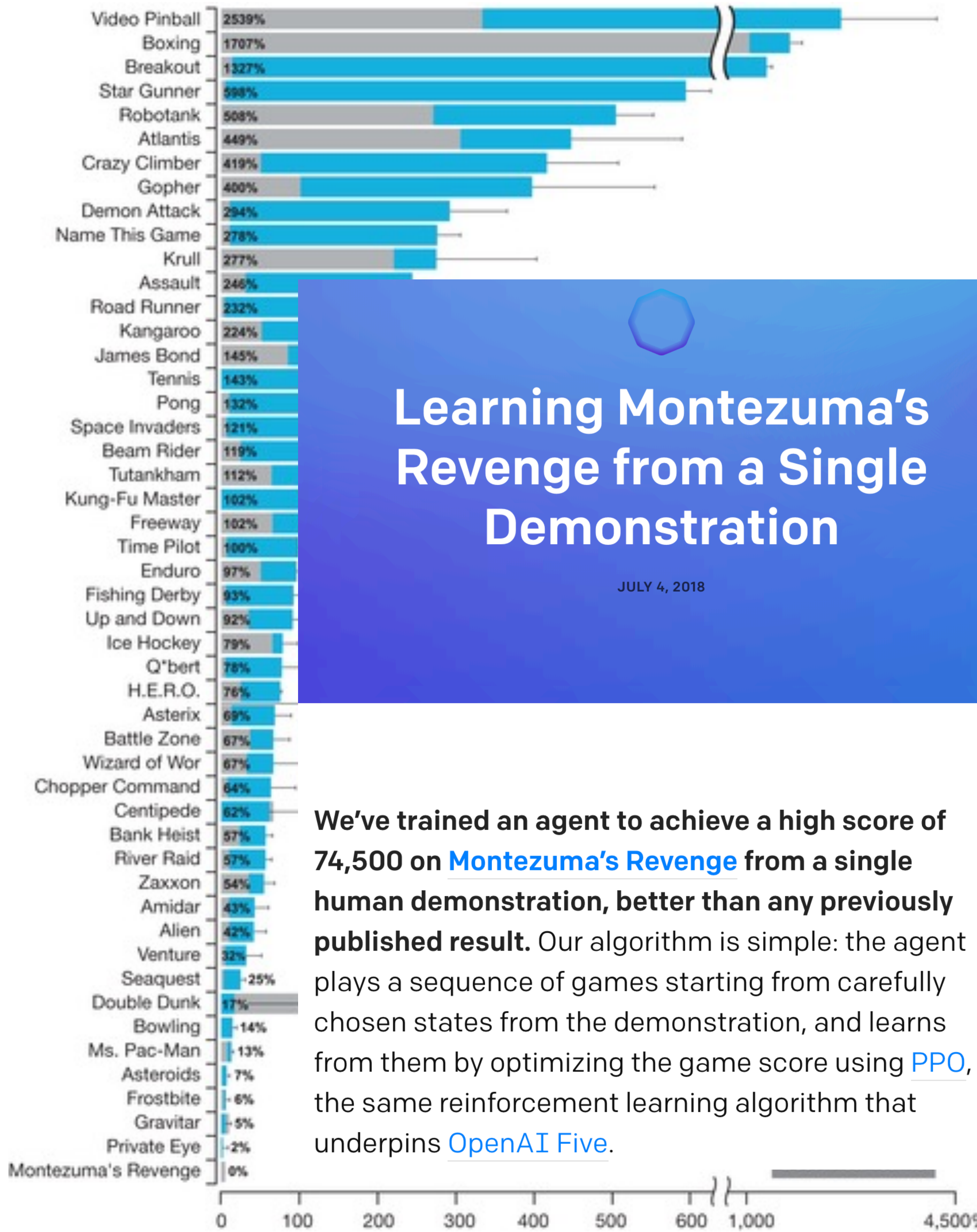


Mnih et al, 2015

Learning to play computer games

- reinforcement learning combined with deep networks
- observation: computer screen & score

Mnih et al, 2015

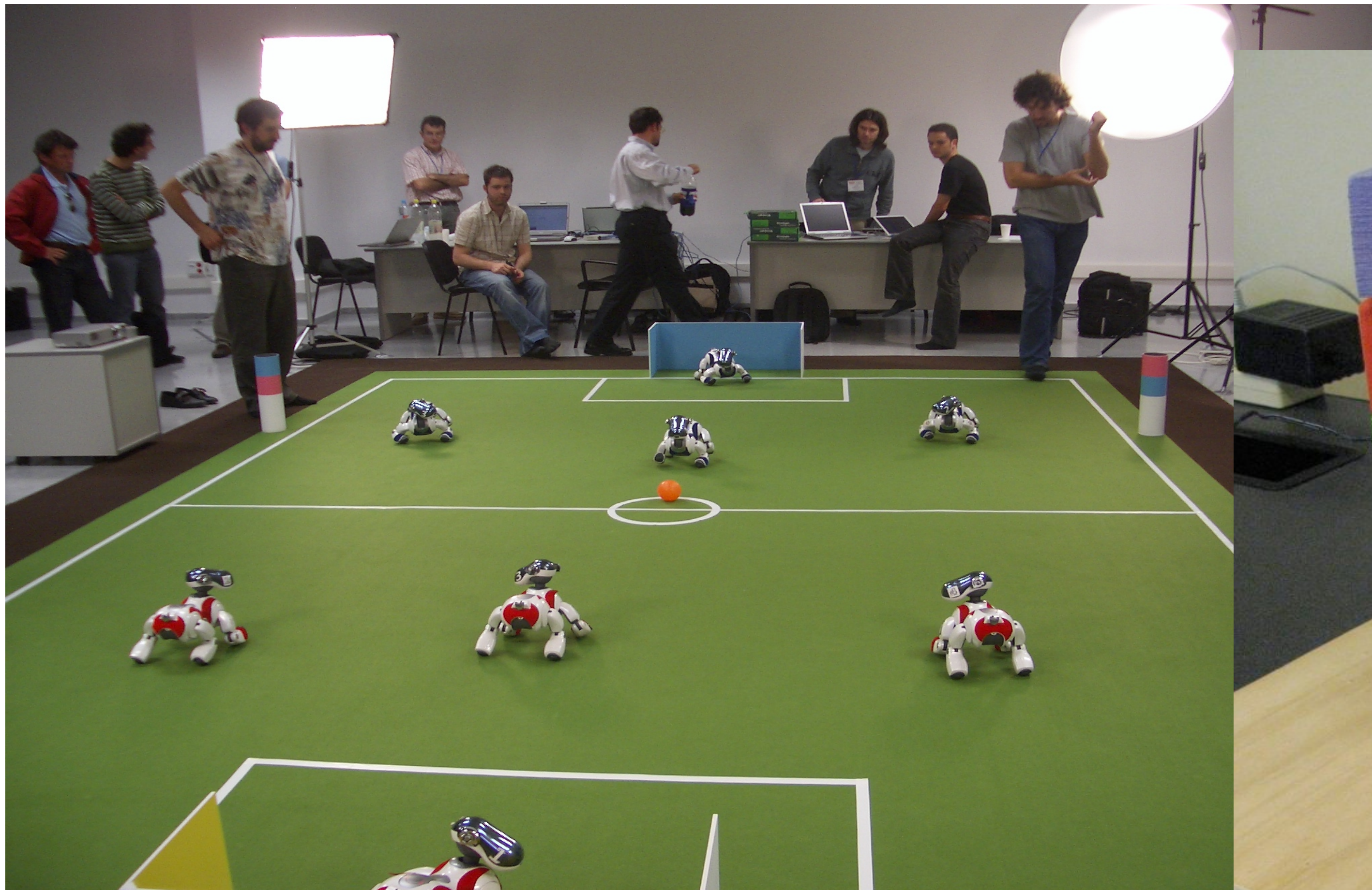


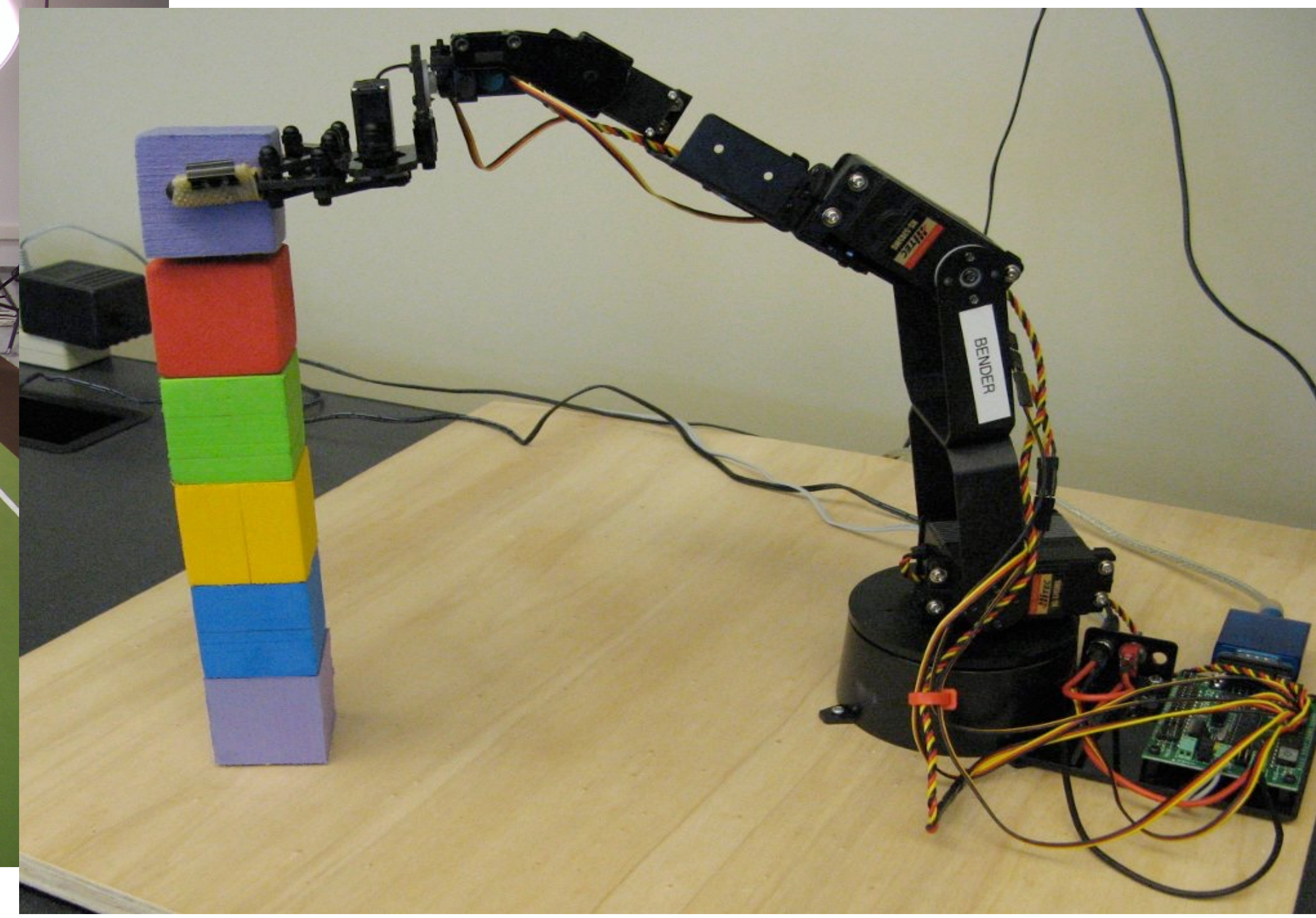
Learning Montezuma's Revenge from a Single Demonstration

JULY 4, 2018

We've trained an agent to achieve a high score of 74,500 on Montezuma's Revenge from a single human demonstration, better than any previously published result. Our algorithm is simple: the agent plays a sequence of games starting from carefully chosen states from the demonstration, and learns from them by optimizing the game score using PPO, the same reinforcement learning algorithm that underpins OpenAI Five.

Learning physical movement with RL

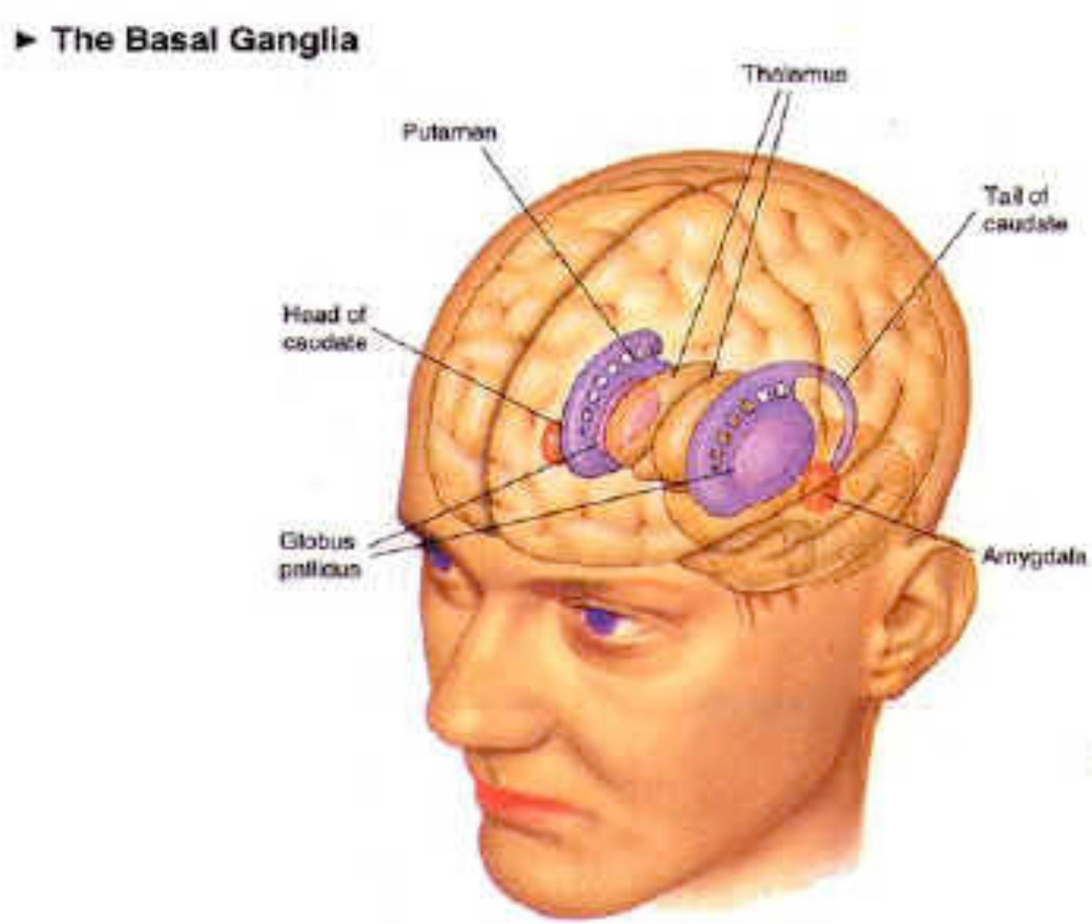


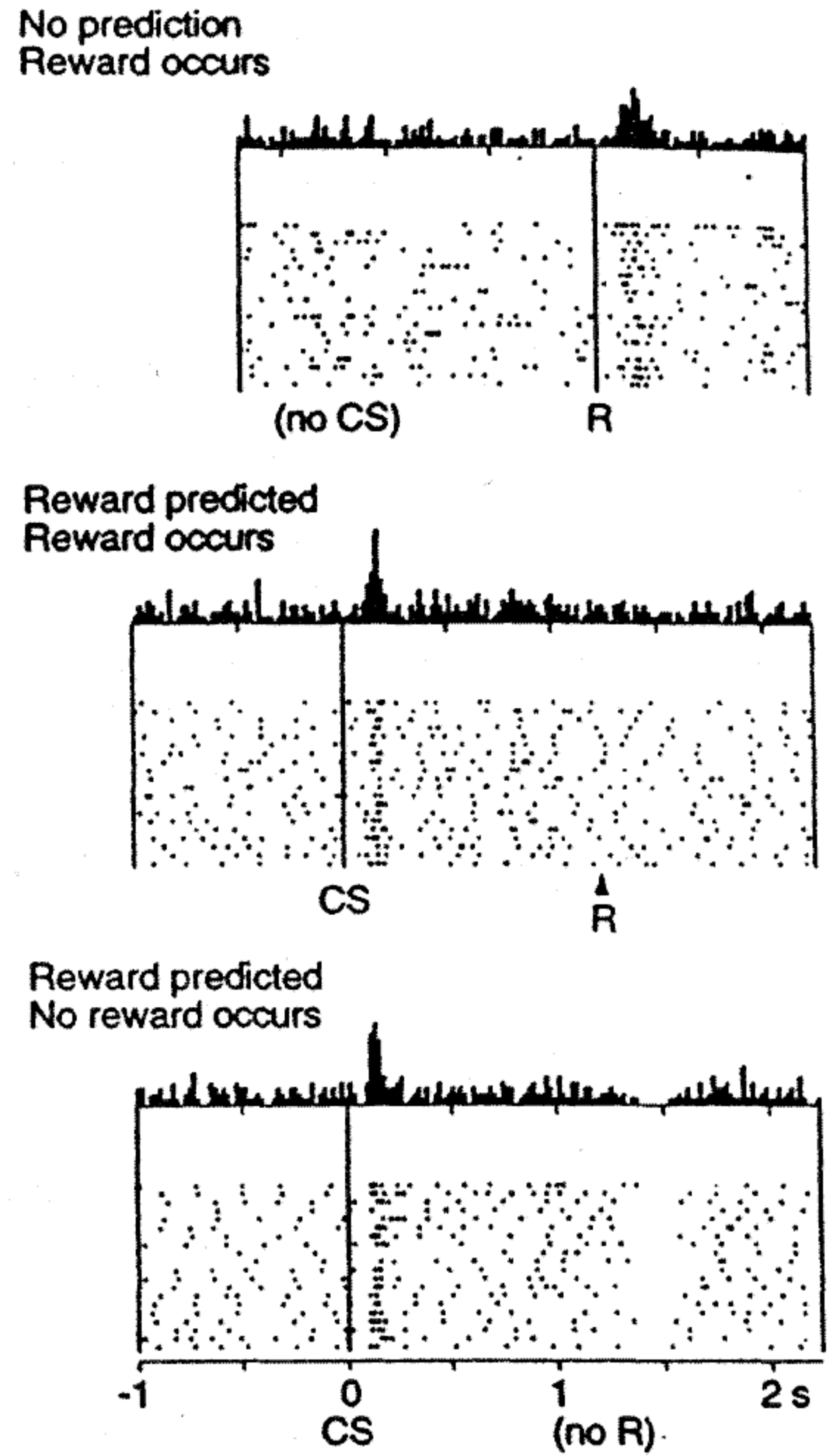


The effect of reward in dopaminerg cell of basal ganglia

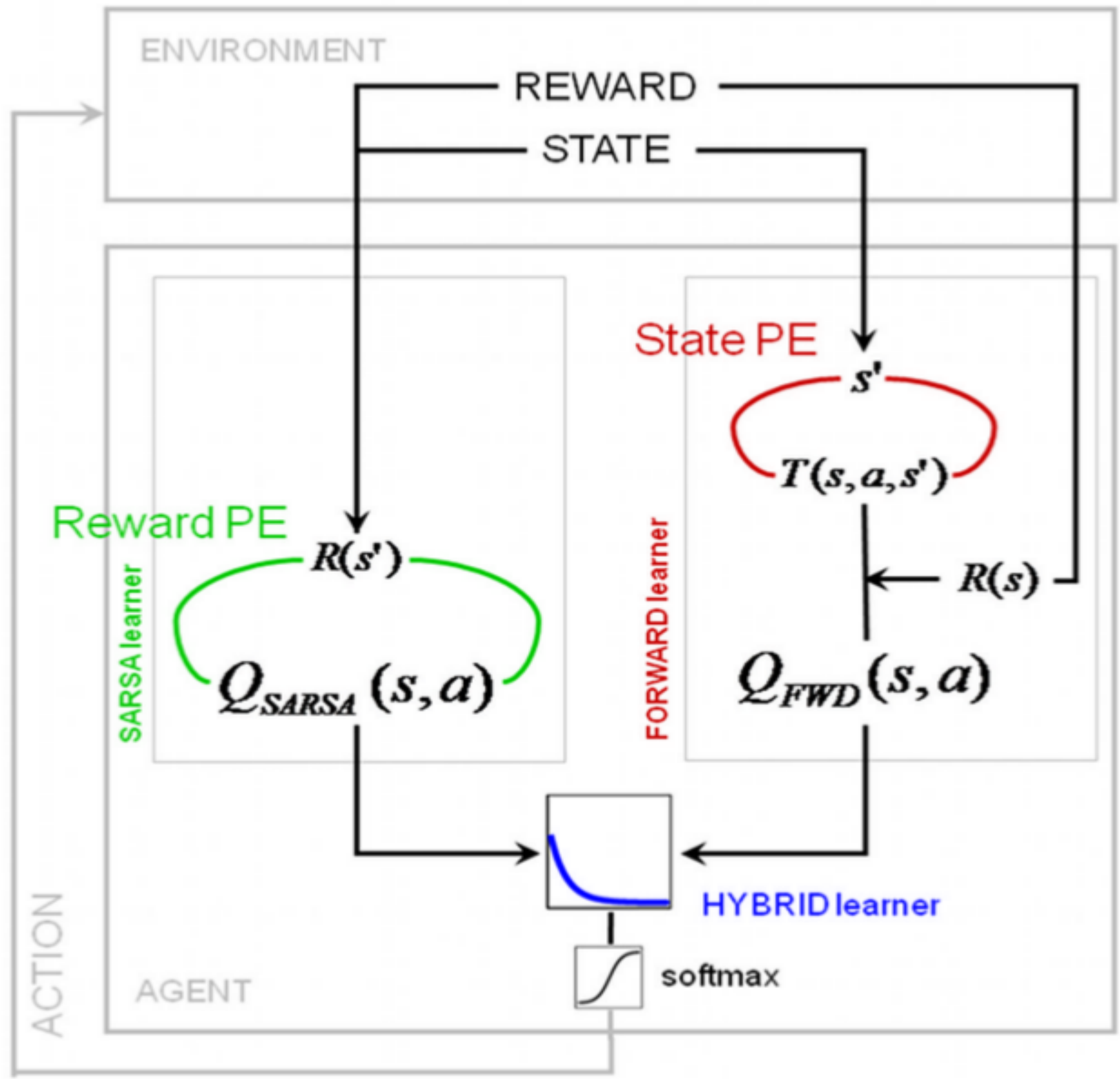
An interpretation:

Dopamine cells signals the difference between the expected and received reward.



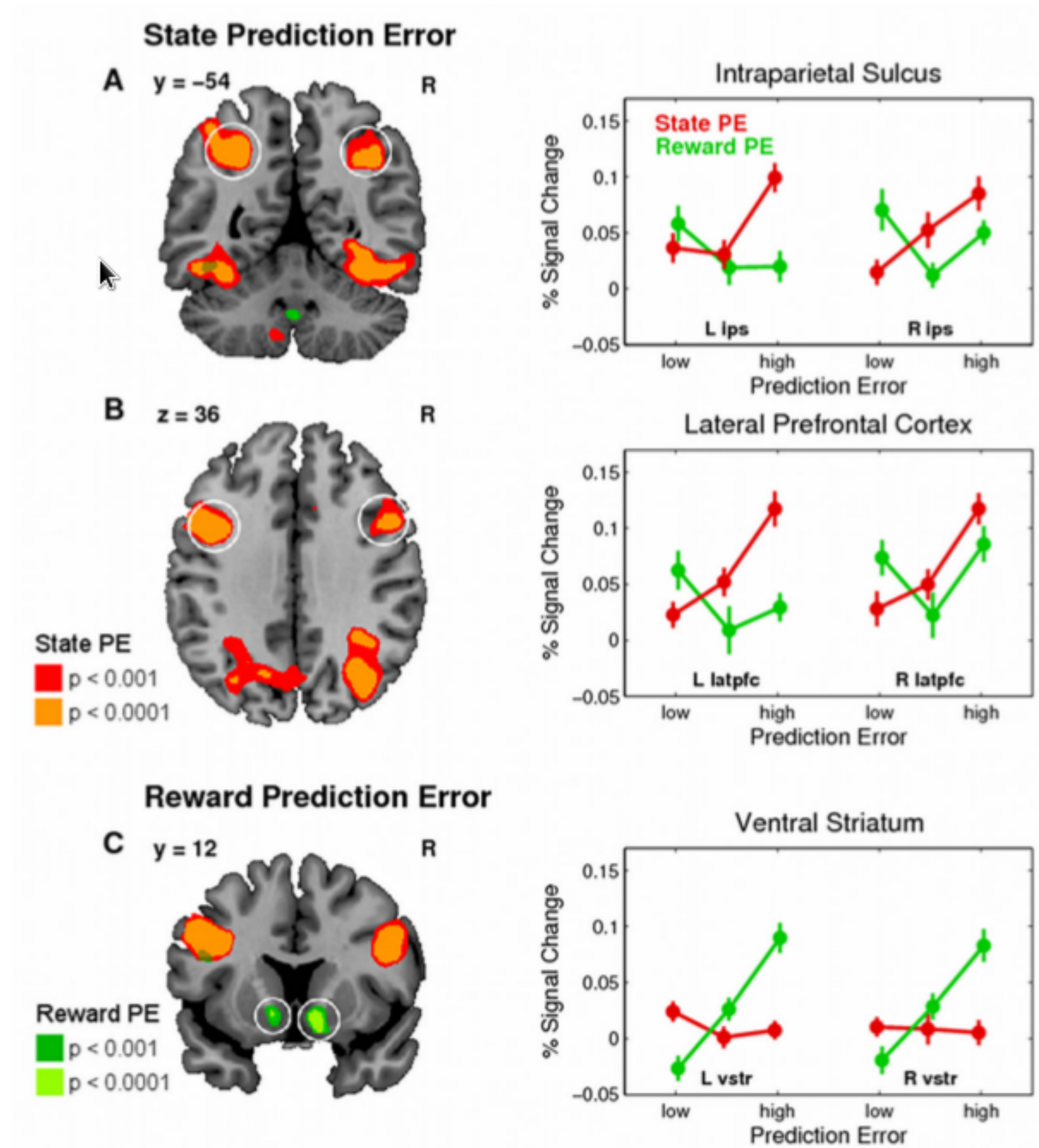


- Glascher, Daw Dayan, O'Doherty, *Neuron,* 2010.
- Correlation of brain activity with model-based and model-free reinforcement learning algorithms

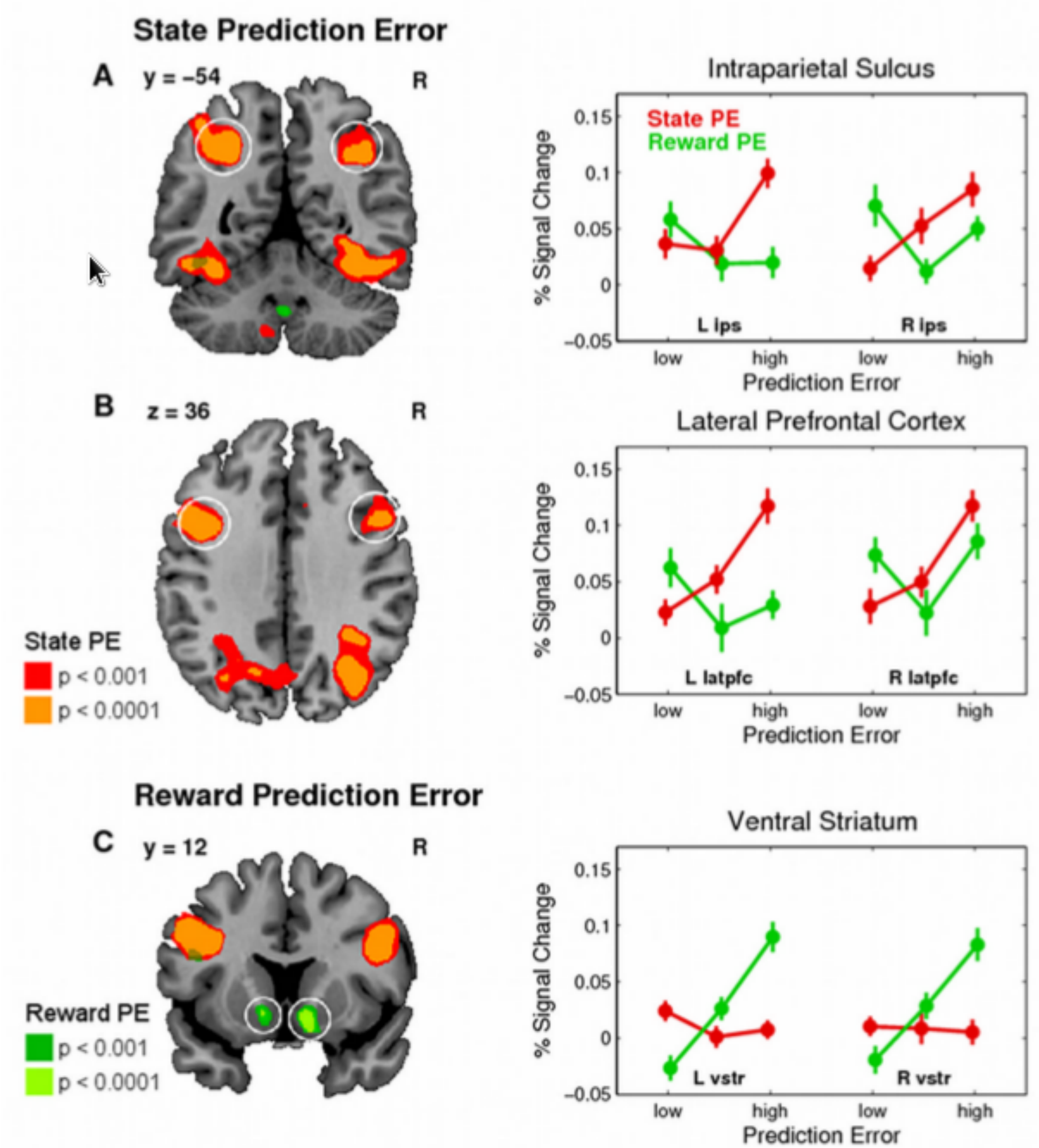


Strategy learning in the cortex

- Model-based and model-free RL
- correlates of quantities related to both can be found with fMRI

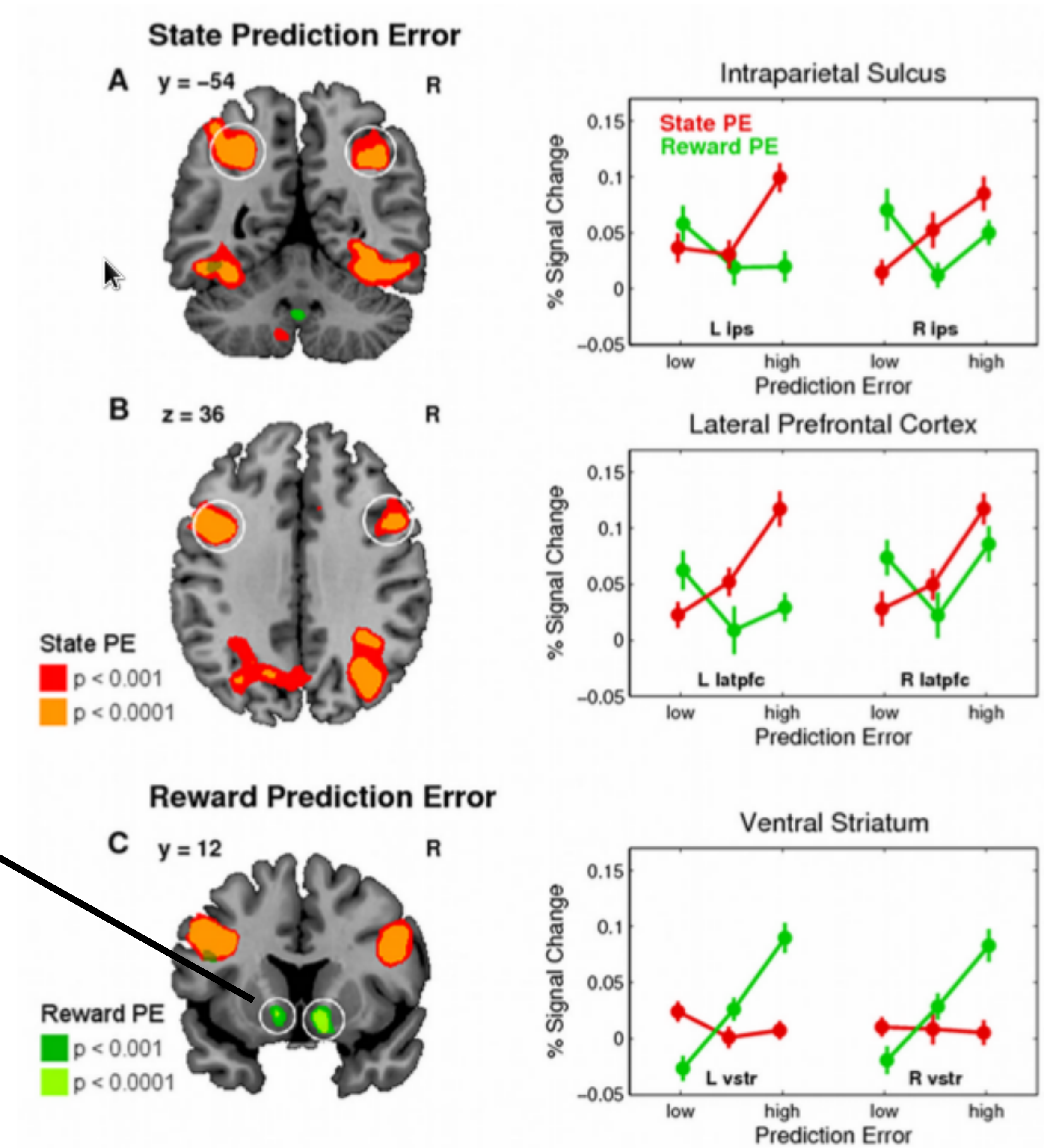


Neural correlates of RL solutions

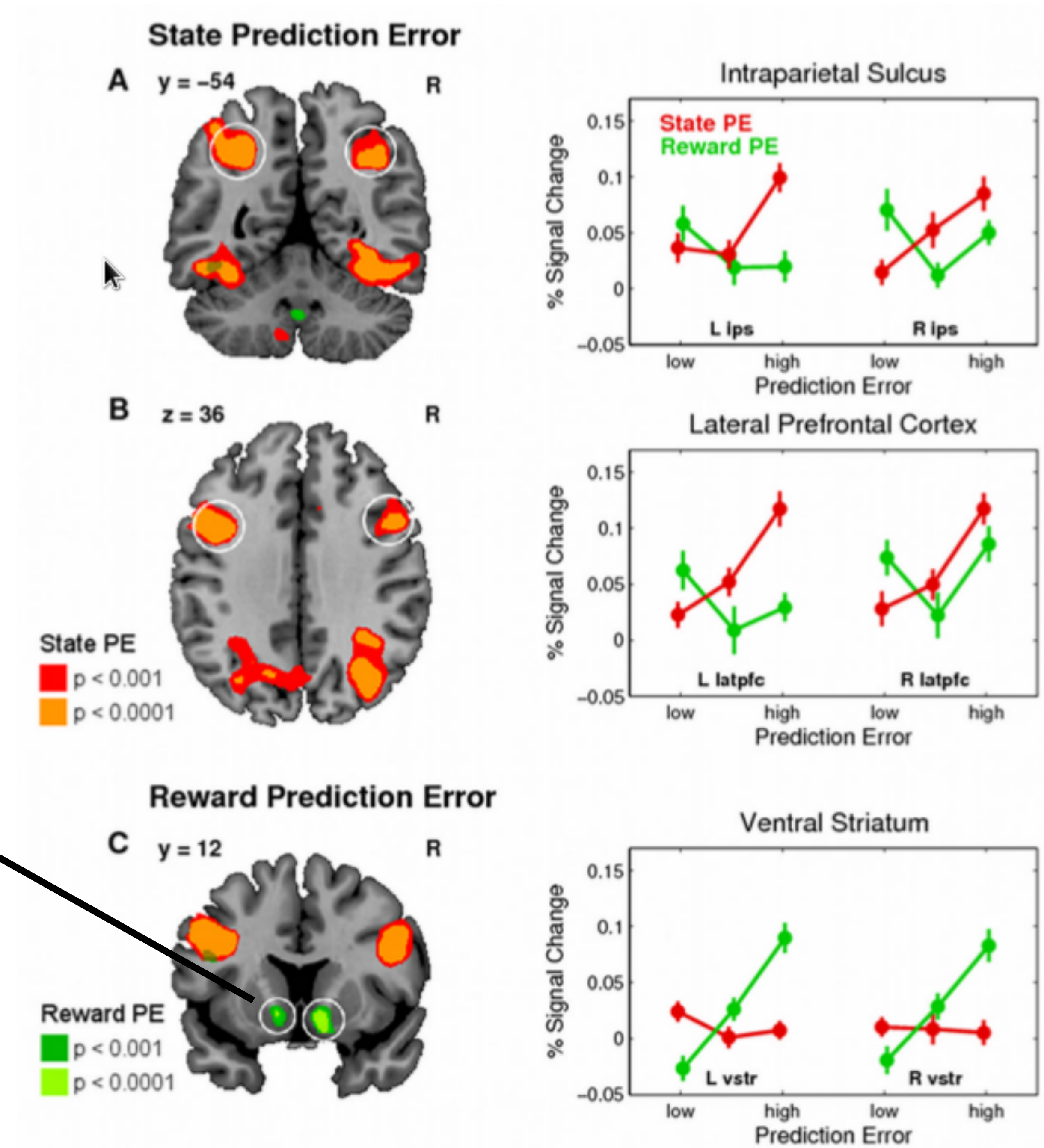


Neural correlates of RL solutions

Reactive model-free RL subcortical structures



Neural correlates of RL solutions

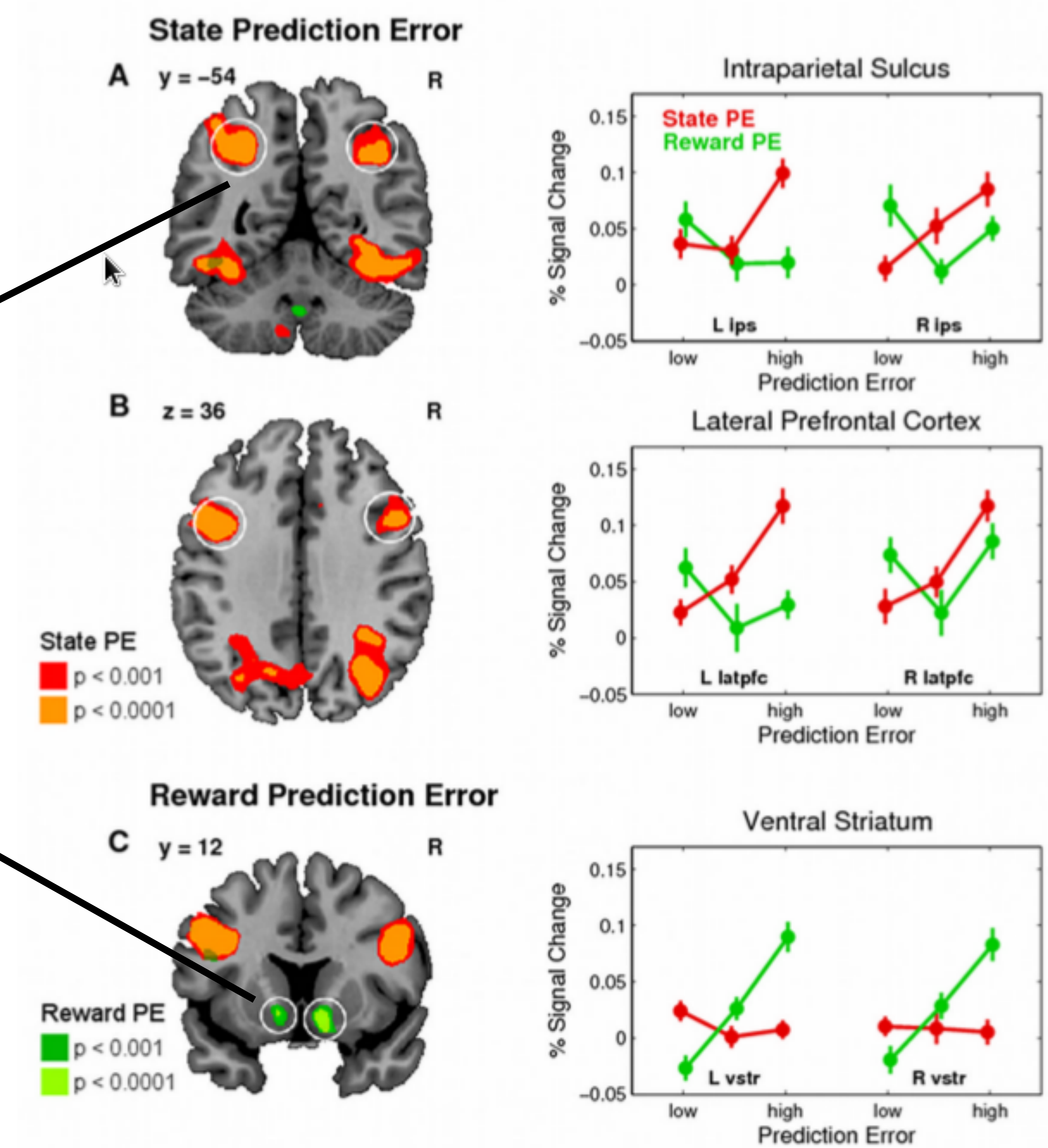


Reactive model-free RL subcortical structures

Neural correlates of RL solutions

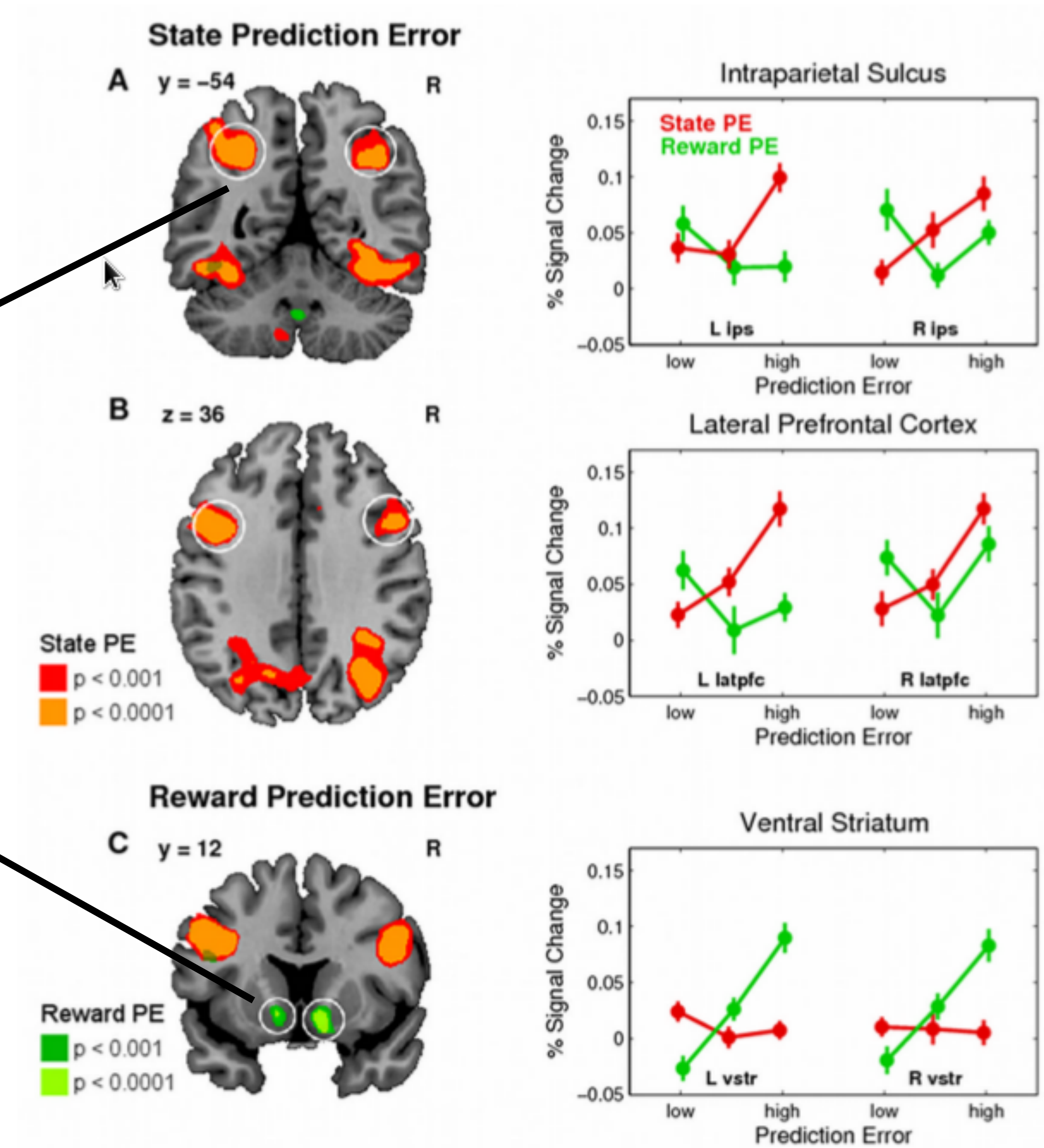
Deliberative model-based RL prefrontal and parietal cortex

Reactive model-free RL subcortical structures



Neural correlates of RL solutions

Deliberative model-based RL
prefrontal and parietal cortex



Reactive model-free RL
subcortical structures

The Big Program of the Brain

```
function MotorOutput = Brain(SensoryInput)

if (Dead ~= 1)
    for i = 1: NumSenses
        ExtractedFeatures(i) = ProcessSense(i, SensoryInput);
    end
    WorldModel = UpdateWorldModel(ExtractedFeatures,InternalState);
    MotorOutput = GenerateAction(WorldModel);
end
```