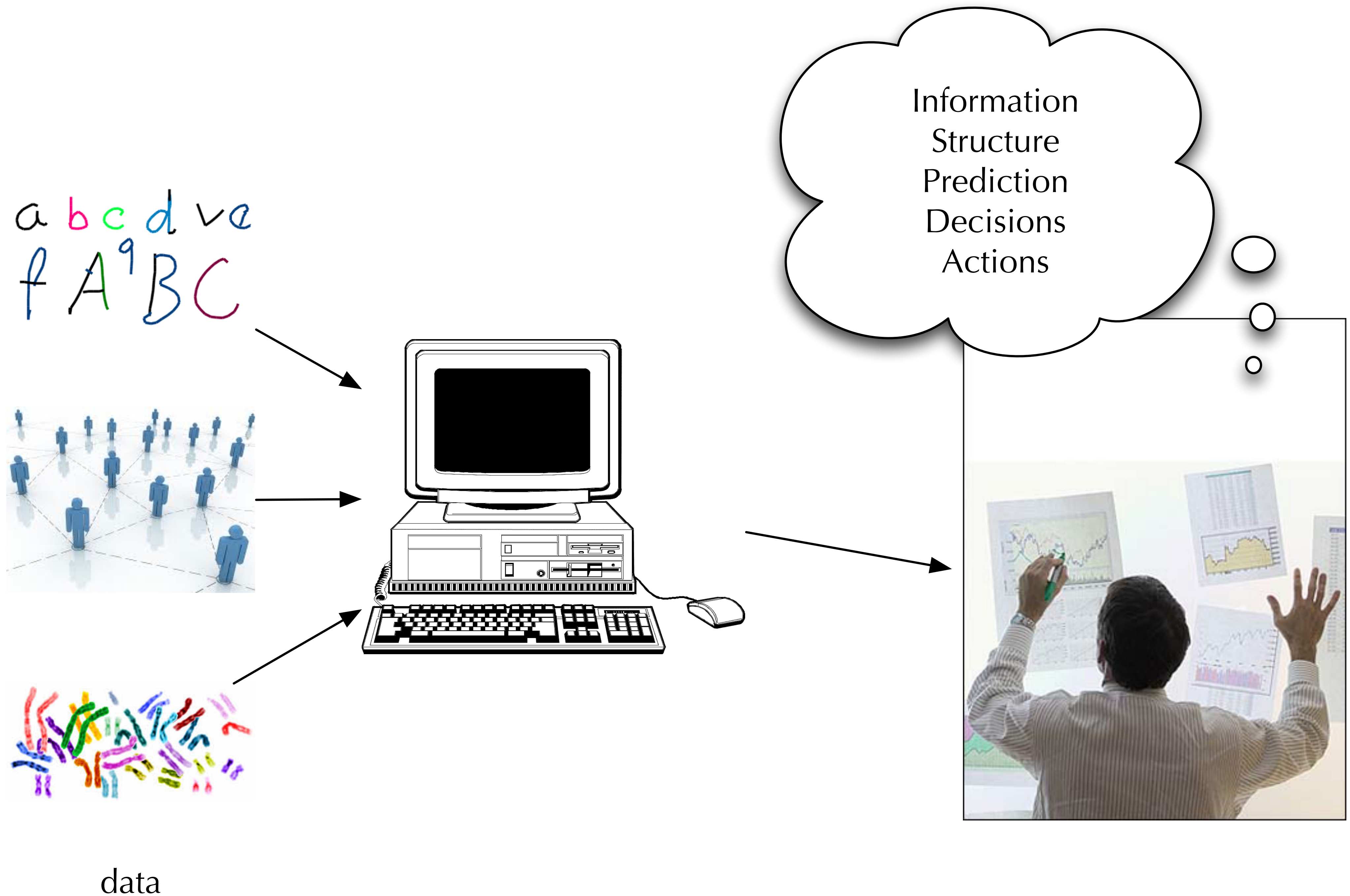
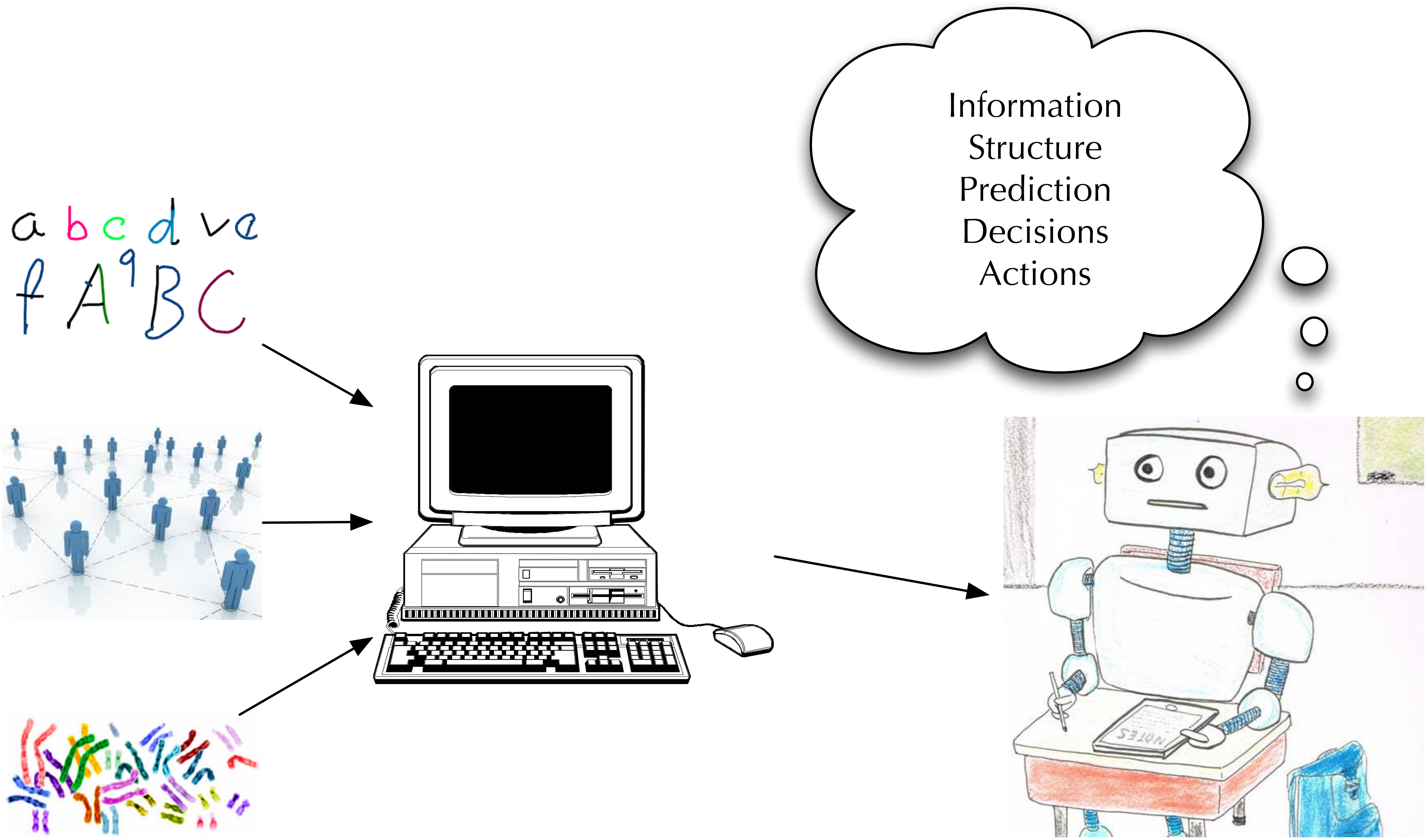
Deep Learning

Yee Whye Teh (Oxford Statistics & DeepMind)
http://csml.stats.ox.ac.uk/people/teh

What is Machine Learning?



What is Machine Learning?



data

Learning Parameterised Functions

$$\theta^* = \arg\min_\theta \frac{1}{n} \sum_{i=1}^n L(y_i, f_\theta(x_i)) + \lambda \|\theta\|$$

Learning Parameterised Functions

$$\theta^* = \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \lambda \|\theta\|$$

- Modern deep learning frameworks allow for **construction** and **learning** of parameterised functions.
  - Consists of basic building blocks composed into **computation graphs**.
  - Highly **expressive** and **flexible**.
  - **Modular**: reusable complex building blocks are themselves composed of simpler building blocks.

Learning Parameterised Functions

$$\theta^* = \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \lambda \|\theta\|$$

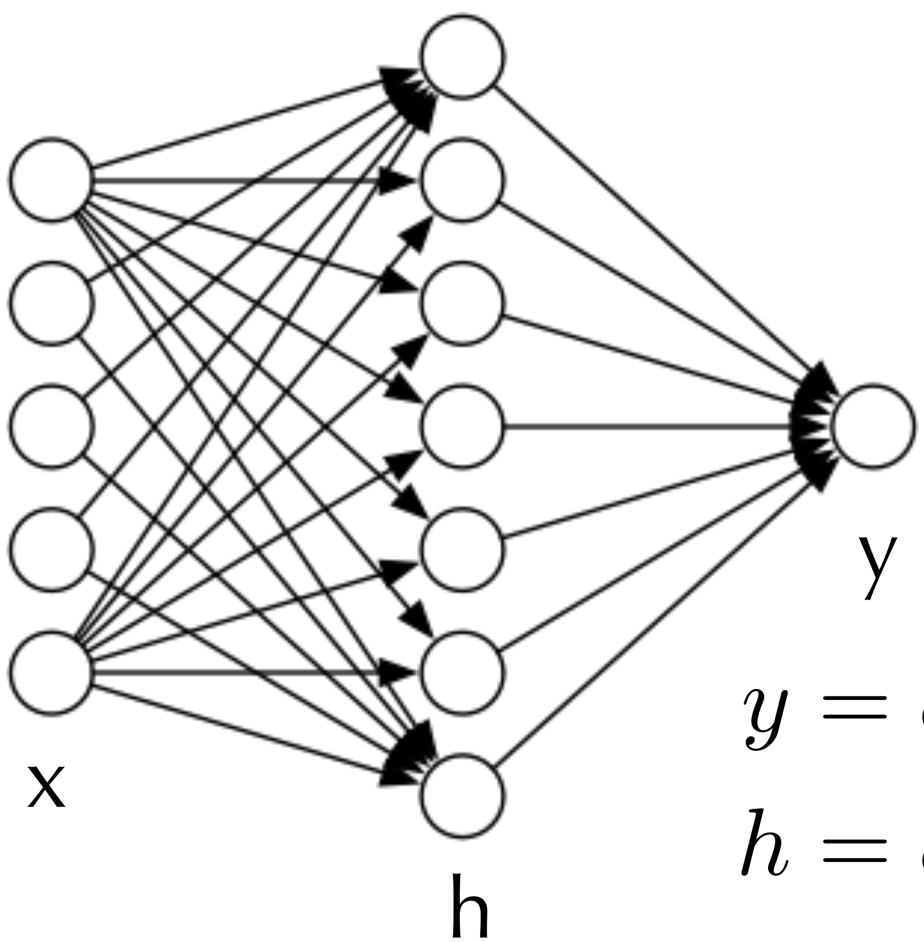
- Modern deep learning frameworks allow for **construction** and **learning** of parameterised functions.
  - Consists of basic building blocks composed into **computation graphs**.
  - Highly **expressive** and **flexible**.
  - **Modular**: reusable complex building blocks are themselves composed of simpler building blocks.
- Computation graph structure expresses **prior knowledge**.

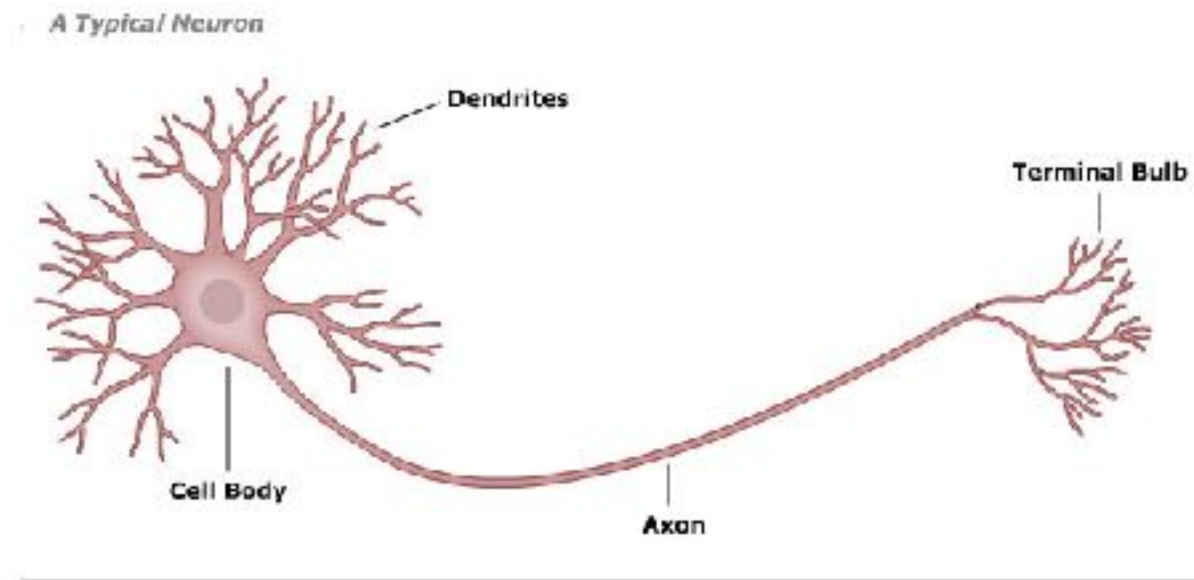
Learning Parameterised Functions

$$\theta^* = \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \lambda \|\theta\|$$

- Modern deep learning frameworks allow for **construction** and **learning** of parameterised functions.
  - Consists of basic building blocks composed into **computation graphs**.
  - Highly **expressive** and **flexible**.
  - **Modular**: reusable complex building blocks are themselves composed of simpler building blocks.
- Computation graph structure expresses **prior knowledge**.
- Learning using stochastic gradient descent (on multiple CPUs, GPUs, clusters) is **automated**.

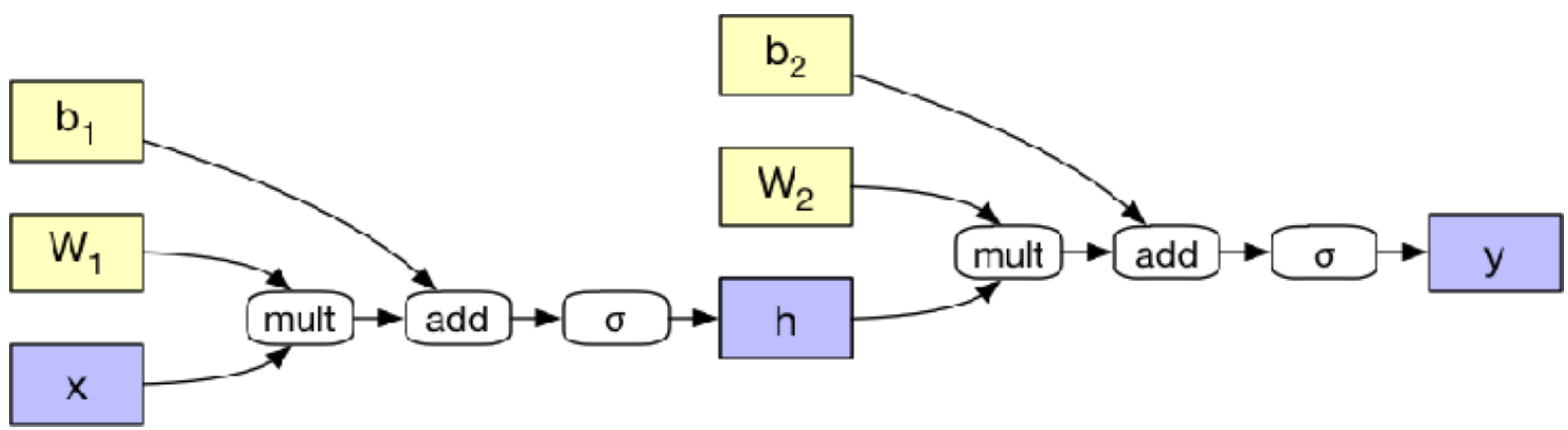
Artificial Neural Networks





$$y = \sigma(W_2 h + b_2)$$

$$h = \sigma(W_1 x + b_1)$$



Building Blocks

- Linear/fully-connected/dense

$$x \mapsto Wx + b$$

- sigmoid

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

- tanh

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

- relu

$$\text{relu}(x) = \max(0, x)$$

- softmax

$$\text{softmax}(x_1, \ldots, x_n)$$
$$= \left(\frac{\exp(x_1)}{\sum_i \exp(x_i)}, \ldots, \frac{\exp(x_n)}{\sum_i \exp(x_i)} \right)$$

- Losses

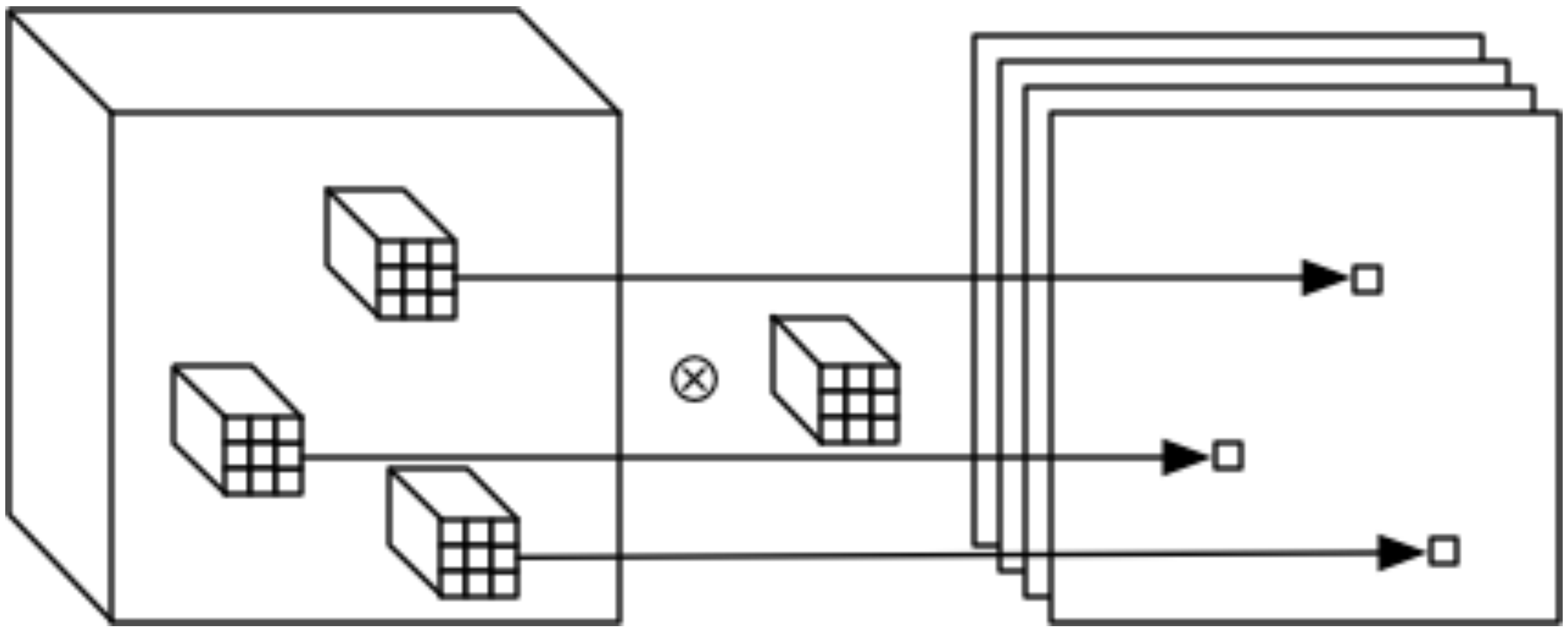
$$\text{CrossEntropy}(t, y) = \sum_i t_i \log y_i$$

$$\text{Square}(t, y) = \|t - y\|_2^2$$

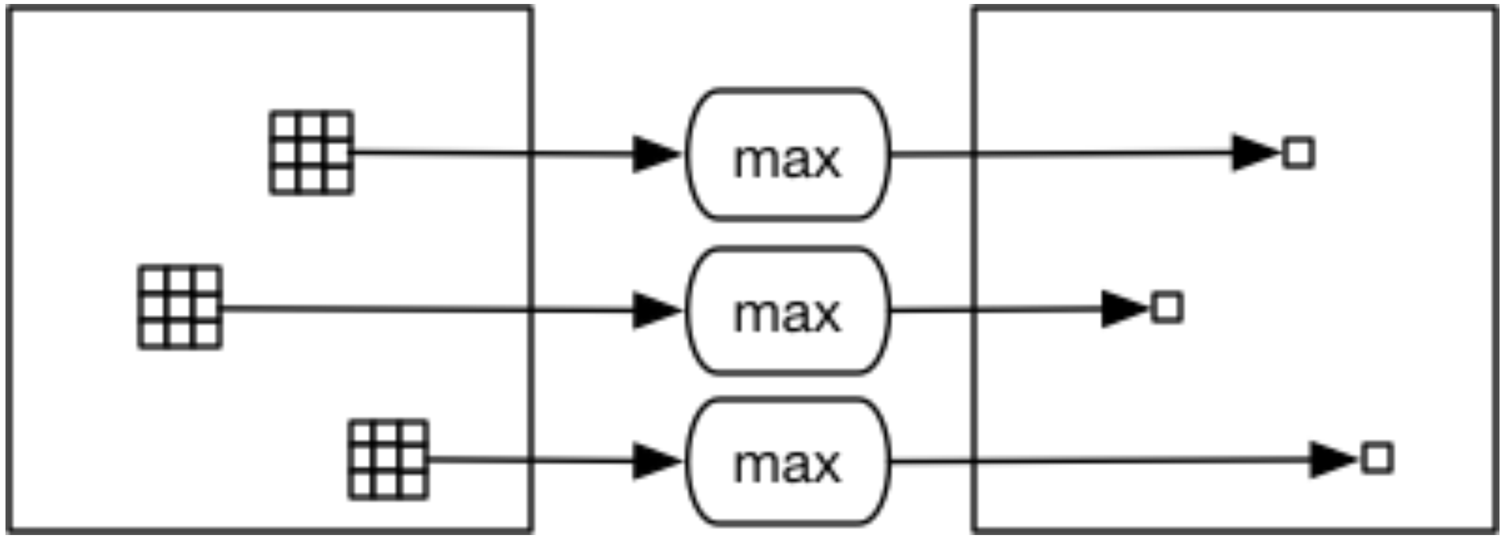
$$\text{Hinge}(t, y) = \max(0, 1 - t \cdot y)$$

Building Blocks

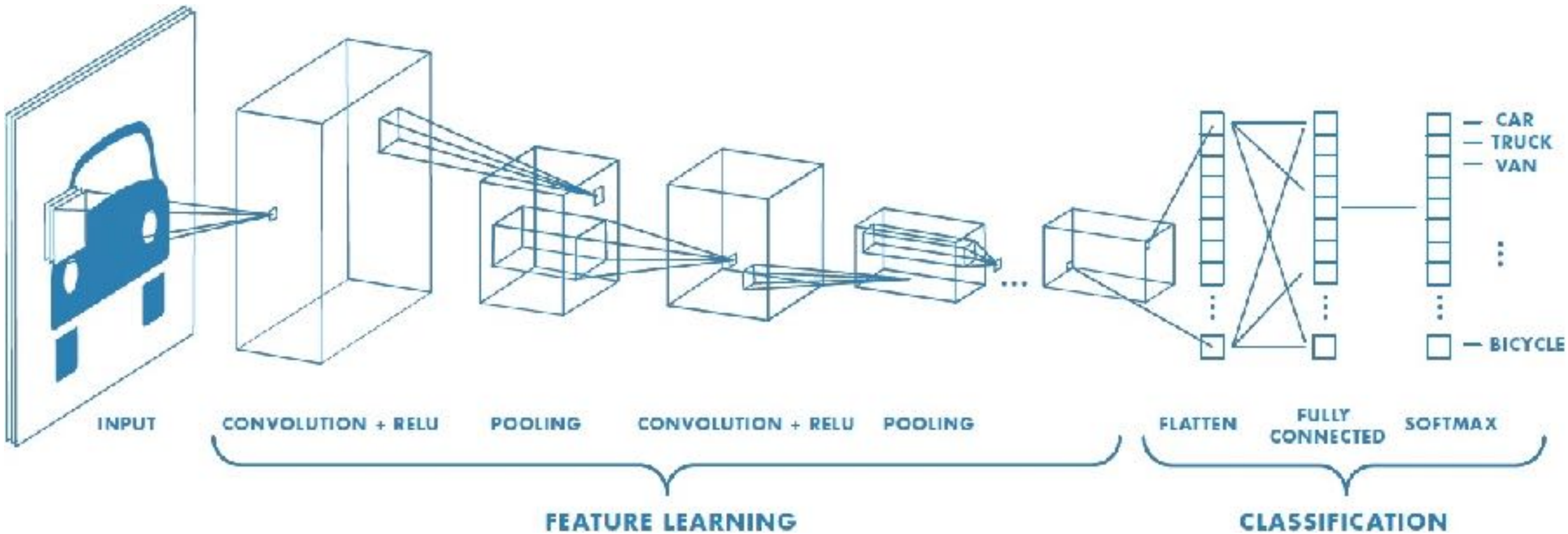
- Convolution



- max pooling

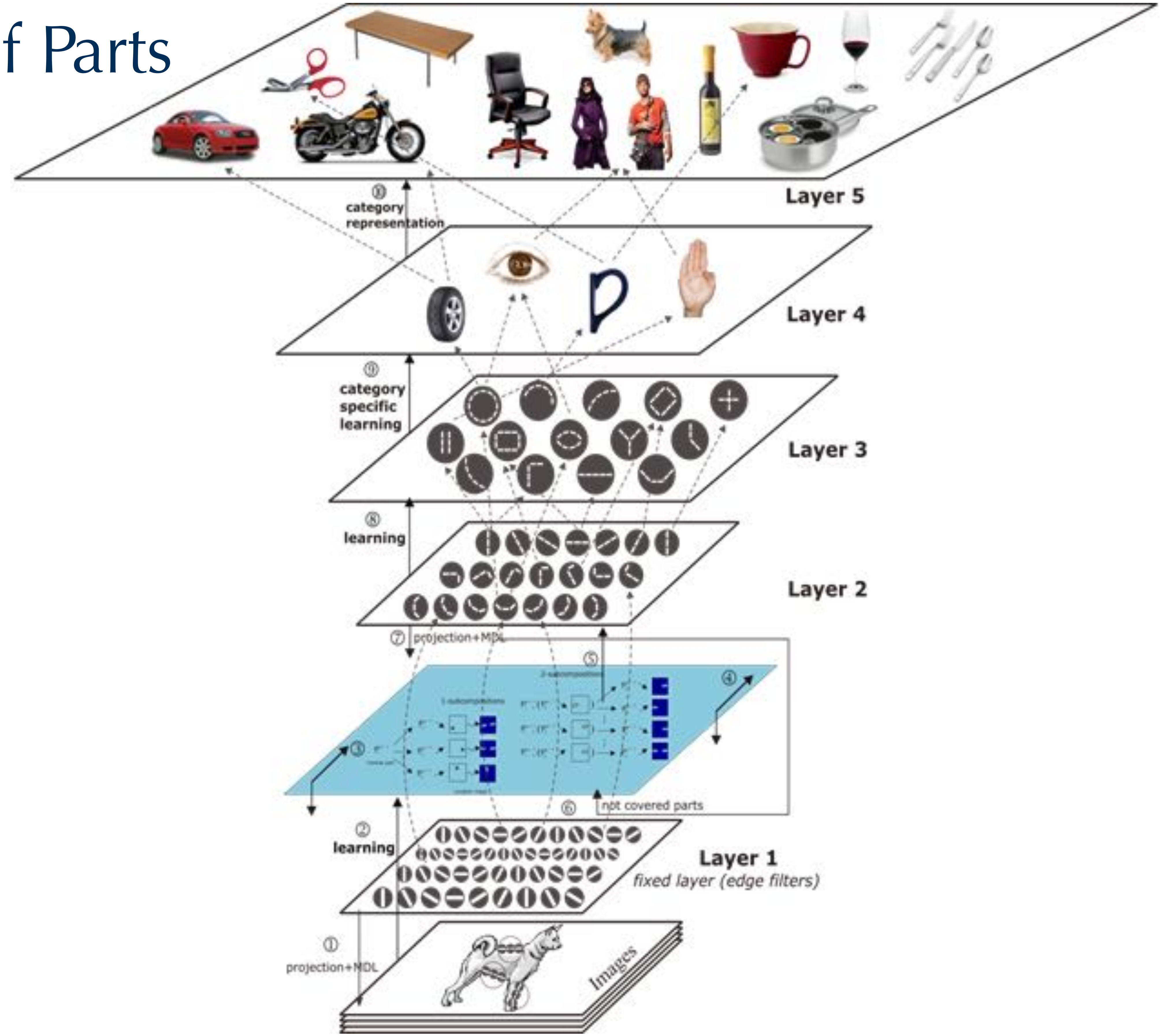


Convolutional Networks (Convnets)

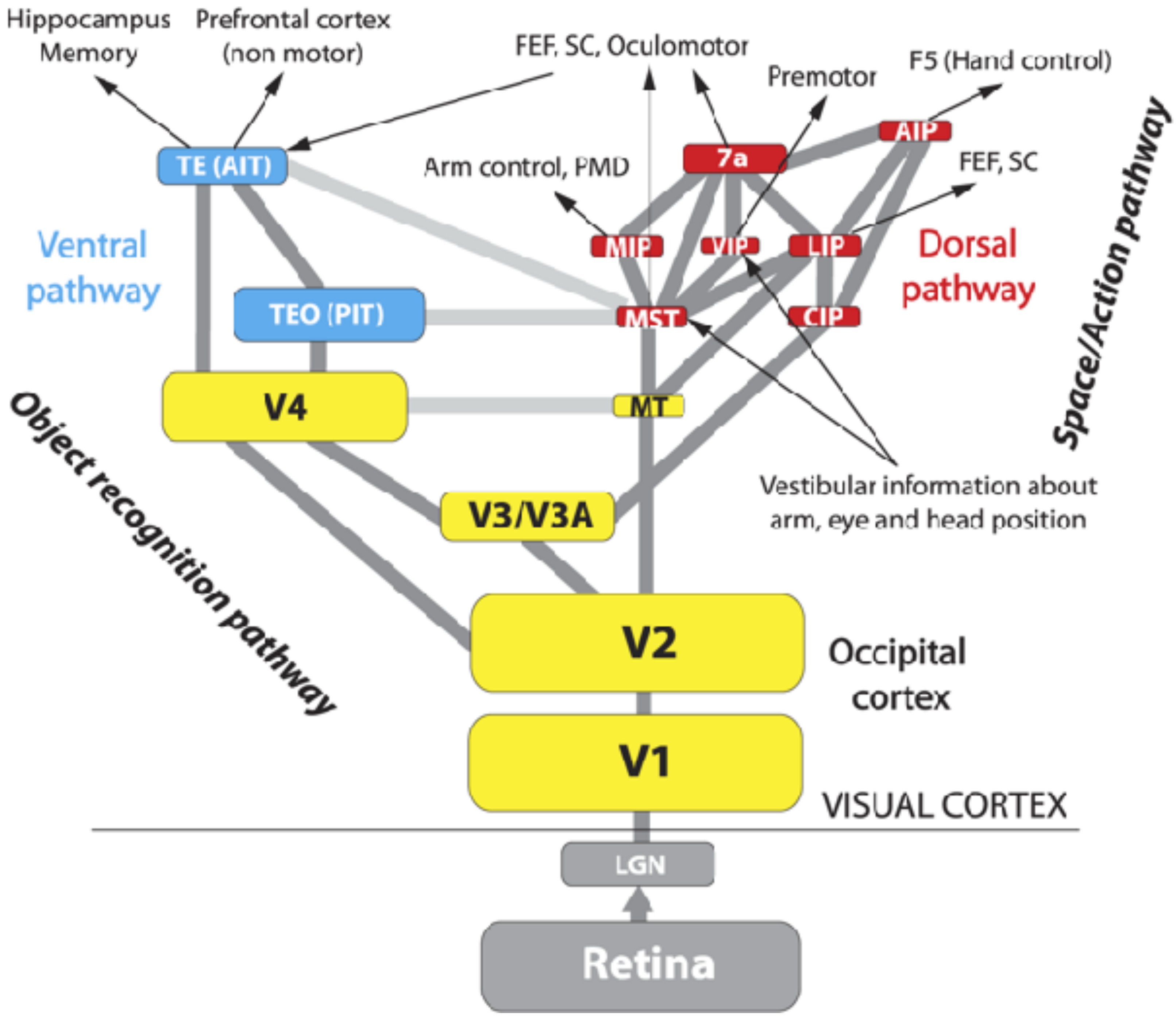


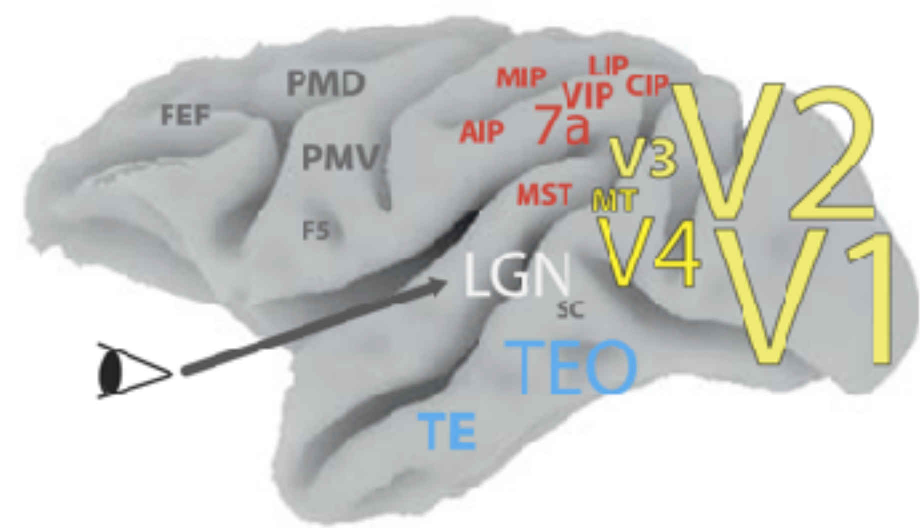
- Both filter banks and layers are 4D tensors.

Hierarchy of Parts



Visual Processing in the Brain



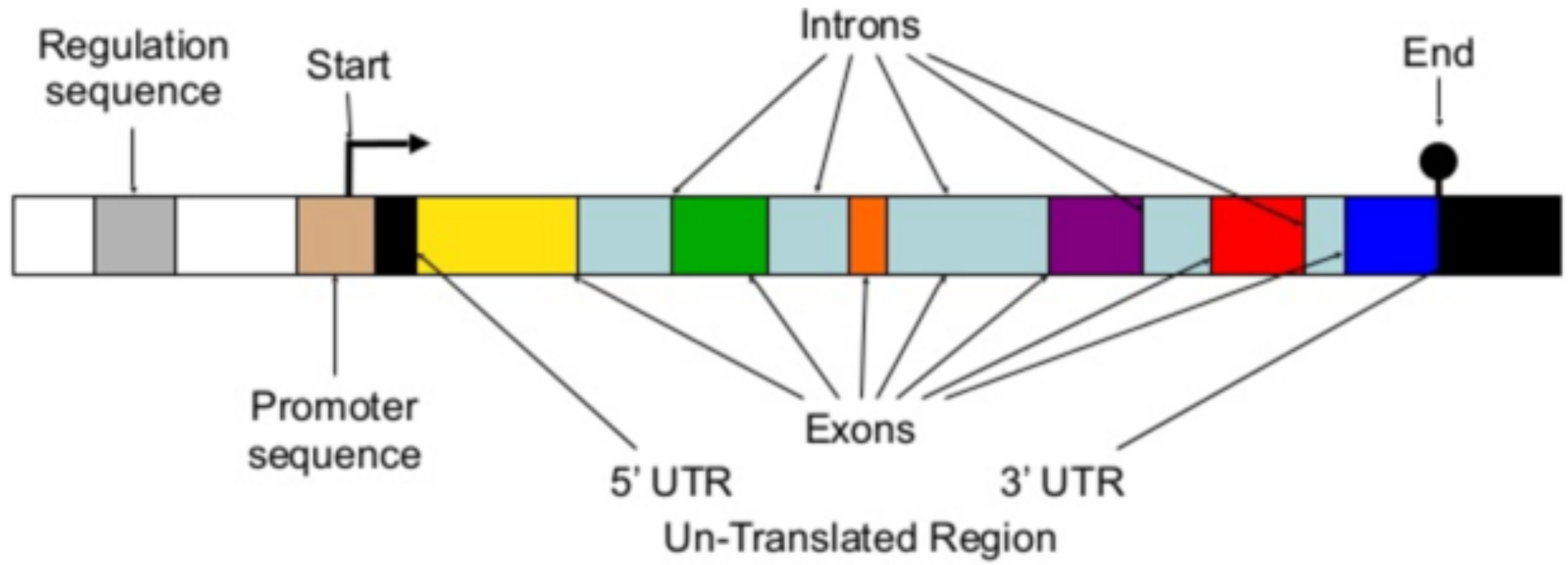


Sequence Models

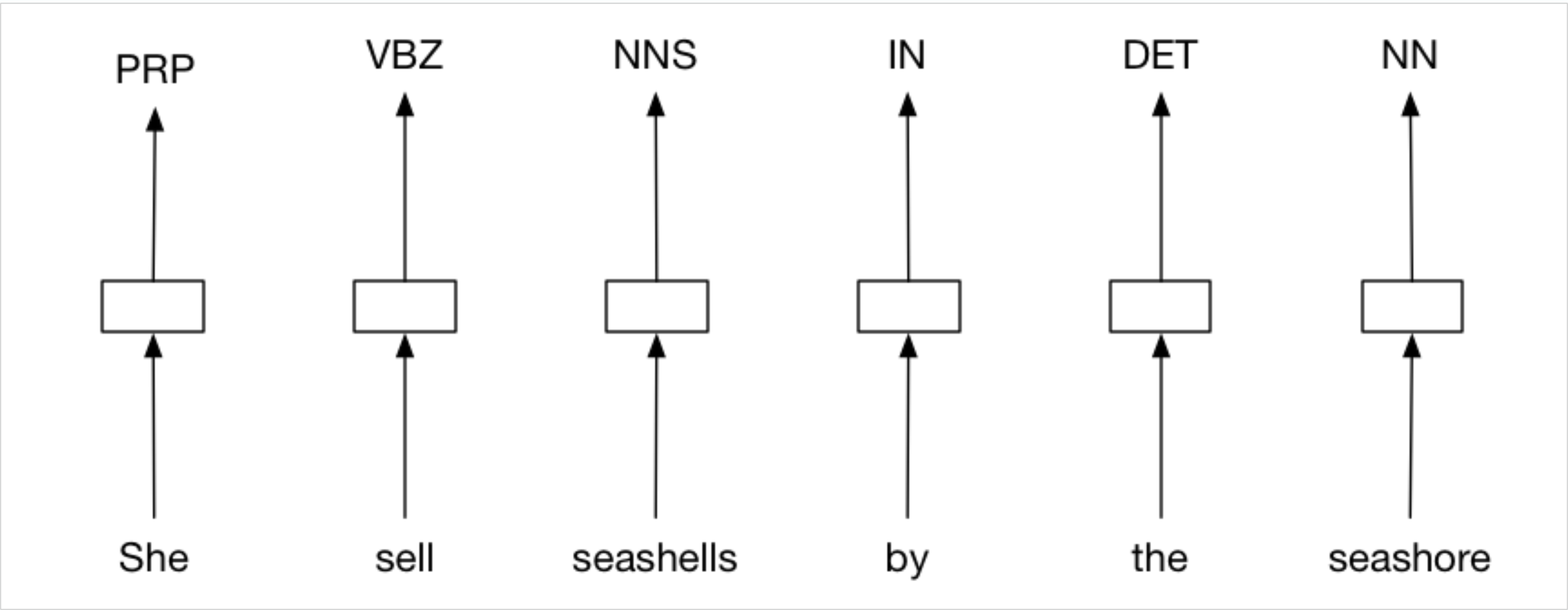
- Natural language processes

PRP VBZ NNS IN DET NN
She sells seashells by the seashore

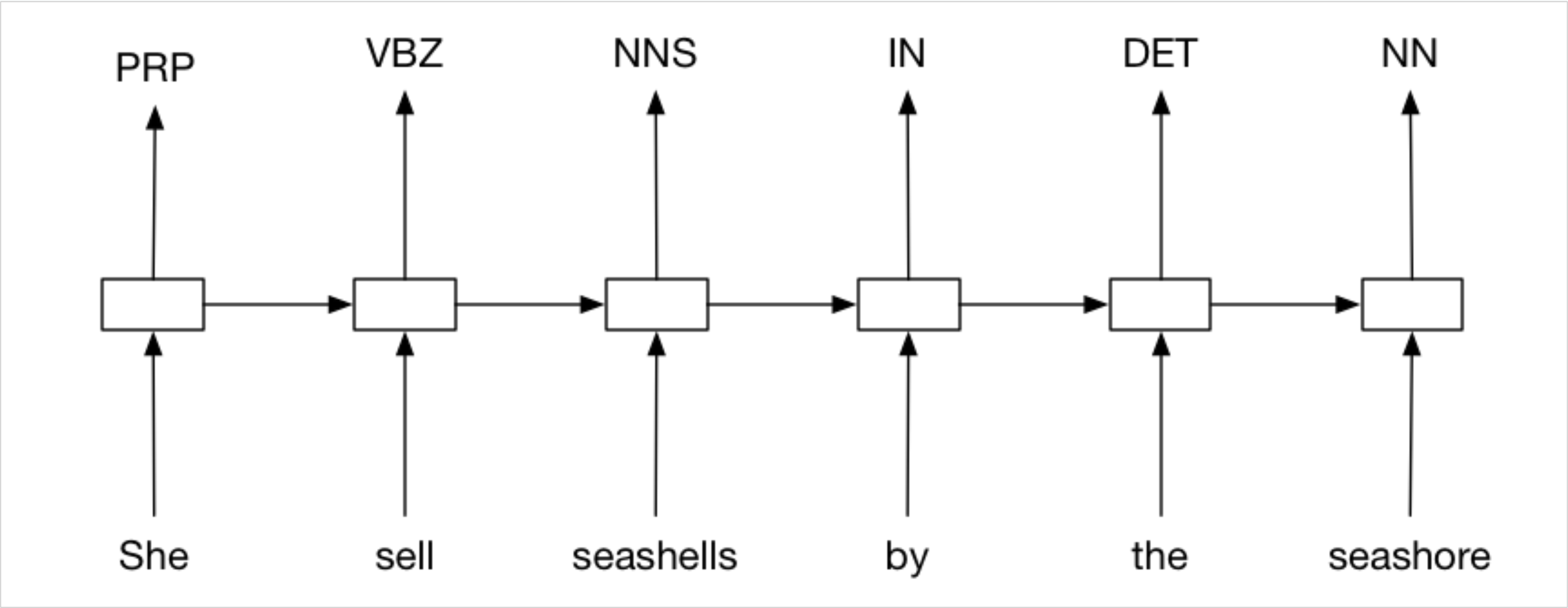
- Genomics



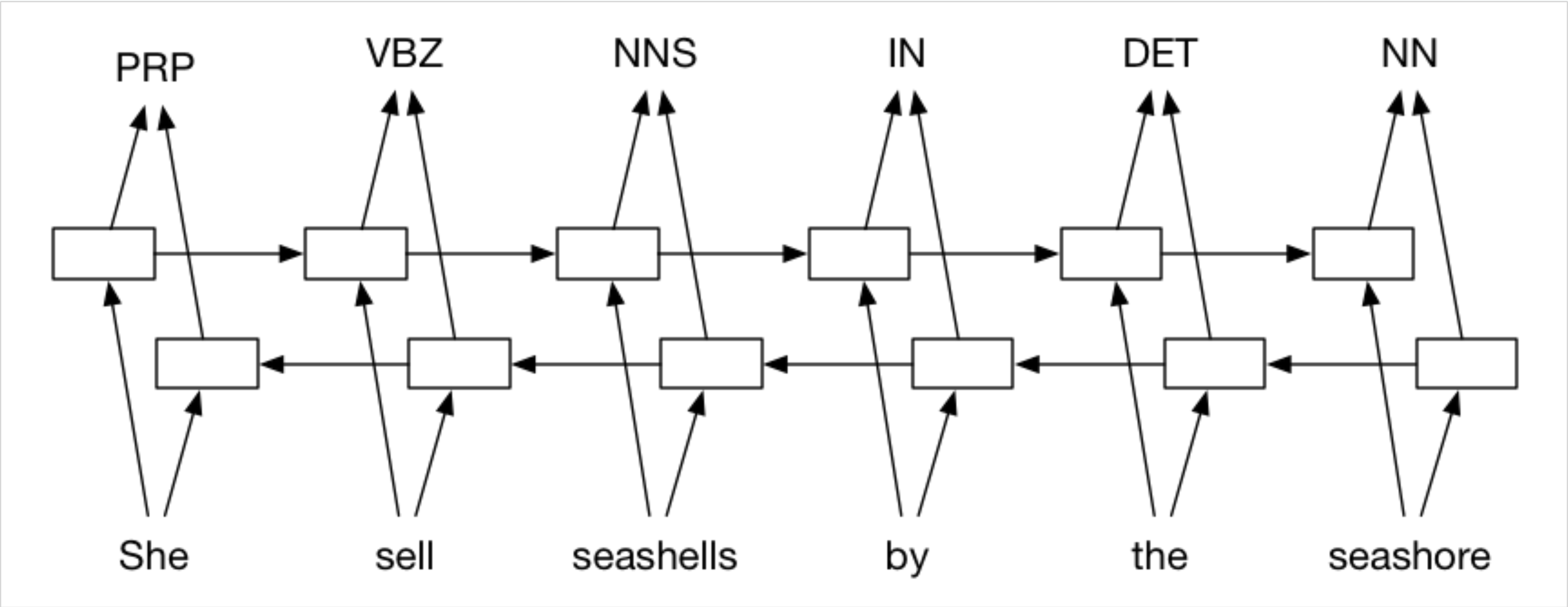
Recurrent Neural Networks



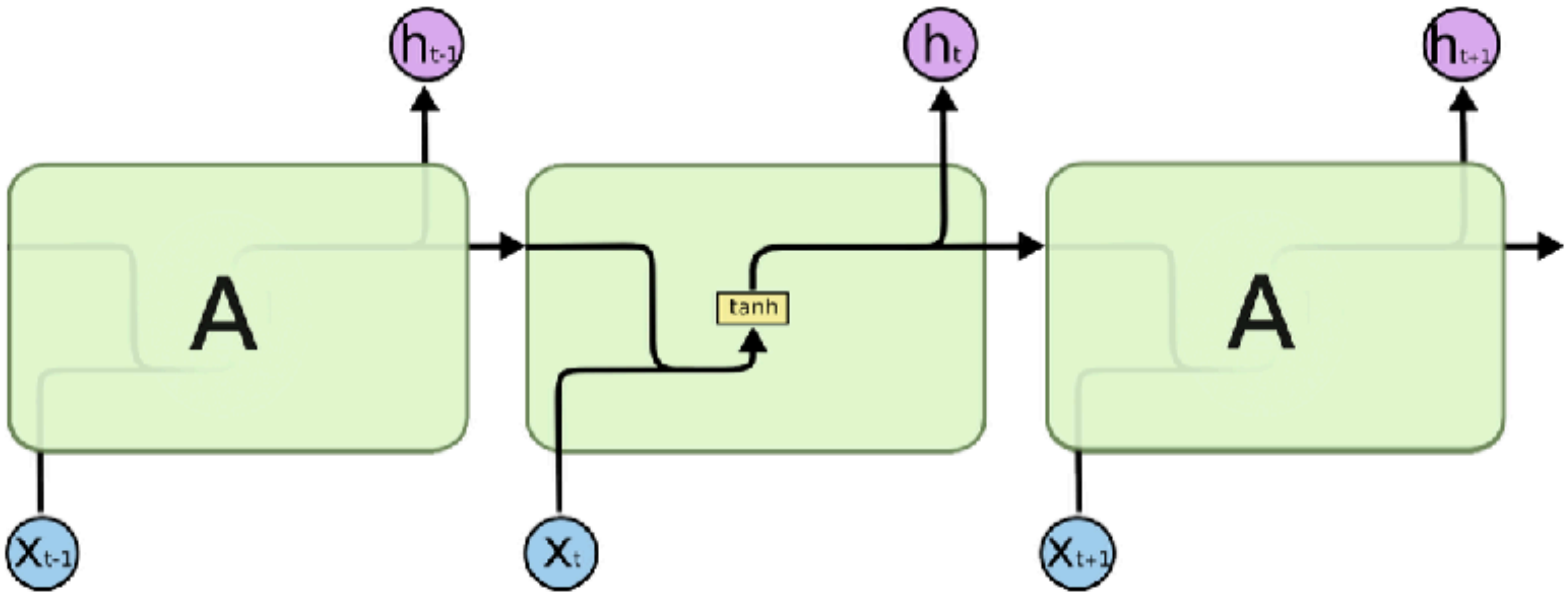
Recurrent Neural Networks



Recurrent Neural Networks



Recurrent Neural Networks

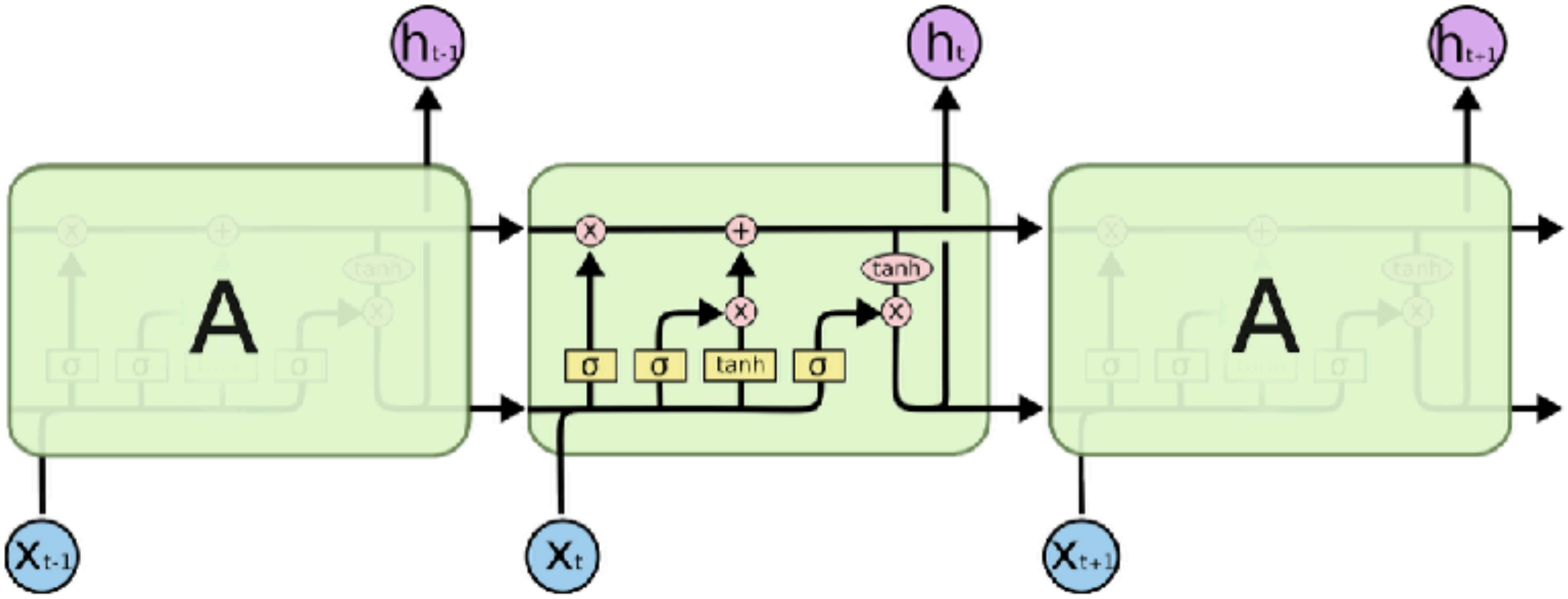


$$h_t = \sigma(W_h z_t + b_h)$$

$$z_t = \tanh(W_z z_{t-1} + W_x x_t + b_z)$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

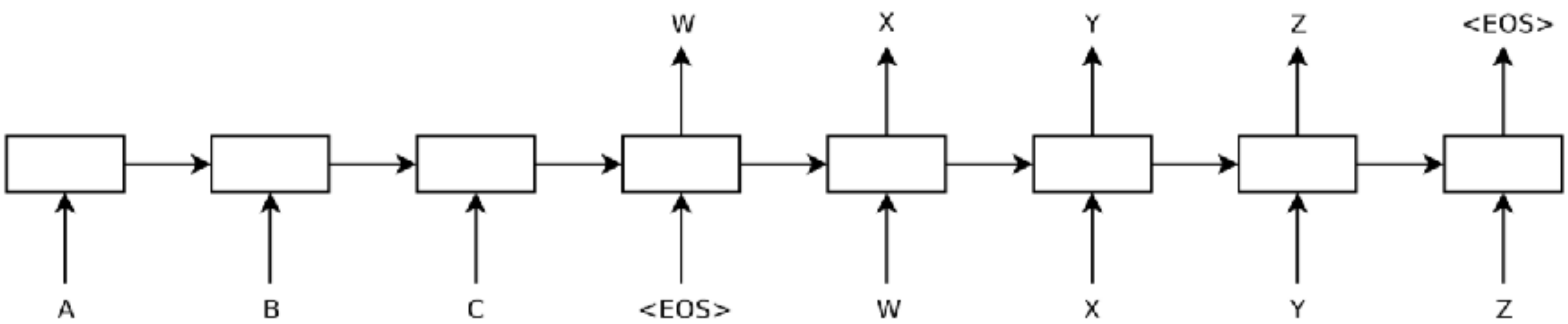
Long Short Term Memory (LSTM)



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Machine Translation with seq2seq

- https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks



GoogLeNet Architecture

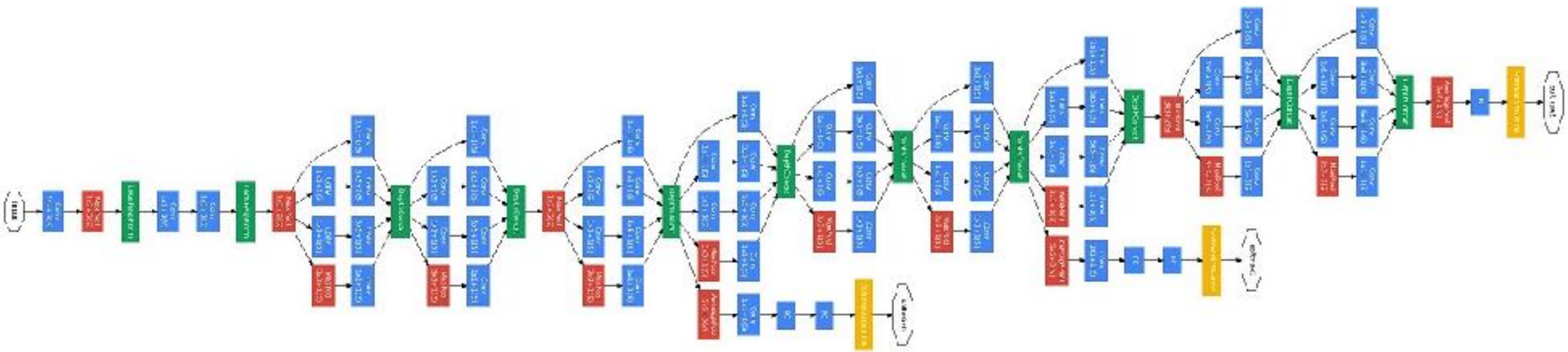
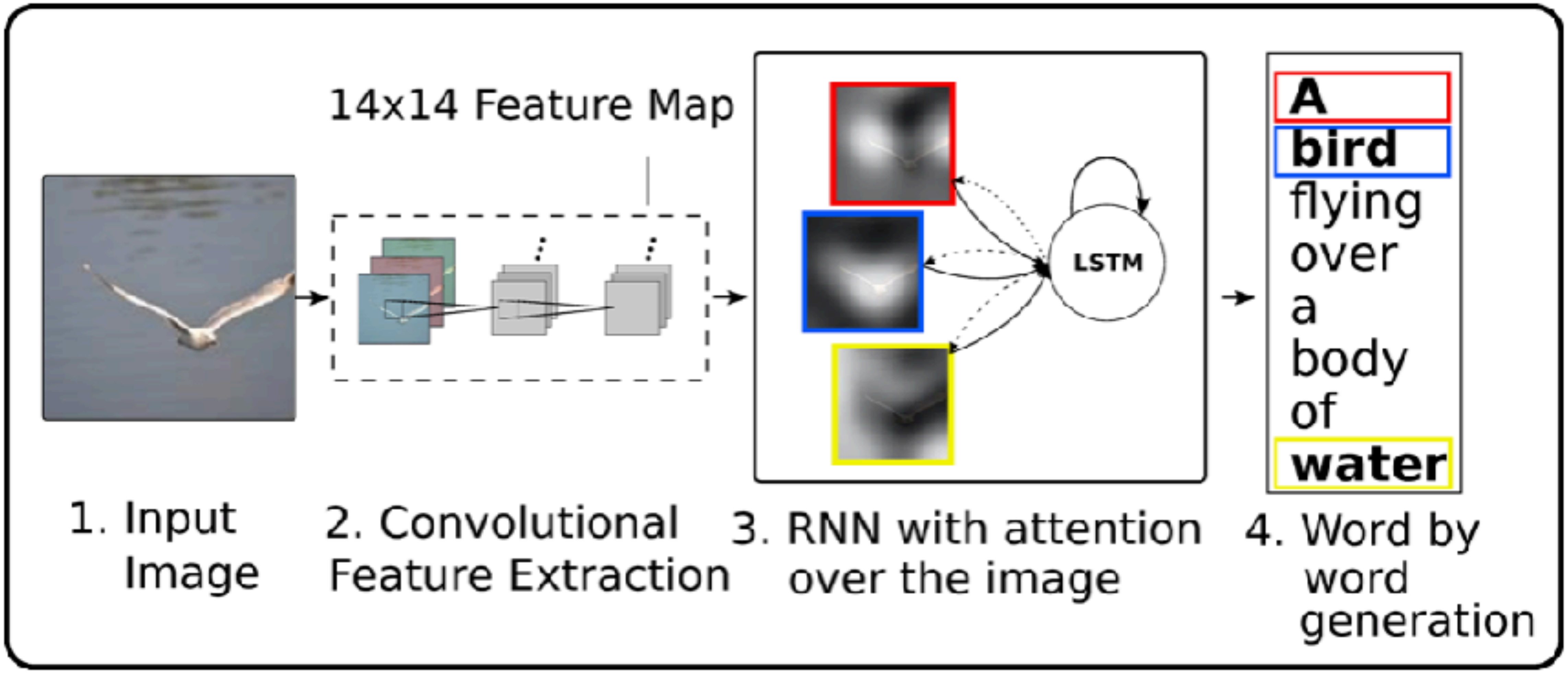


Image Caption Generation

black, orange and white cat laying on some paper on a desk.
cat with mussed up fur sitting discontentedly on a messy desk.
a cat lazily sits in the middle of a cluttered desk.
a cat sitting on top of a pile of papers on a desk.
a dark multicolored cat laying on a table cluttered with various items.

Show Attend and Tell

- http://kelvinxu.github.io/projects/capgen.html



Show Attend and Tell

Figure 2. Attention over time. As the model generates each word, its attention changes to reflect the relevant parts of the image. "soft" (top row) vs "hard" (bottom row) attention. (Note that both models generated the same captions in this example.)

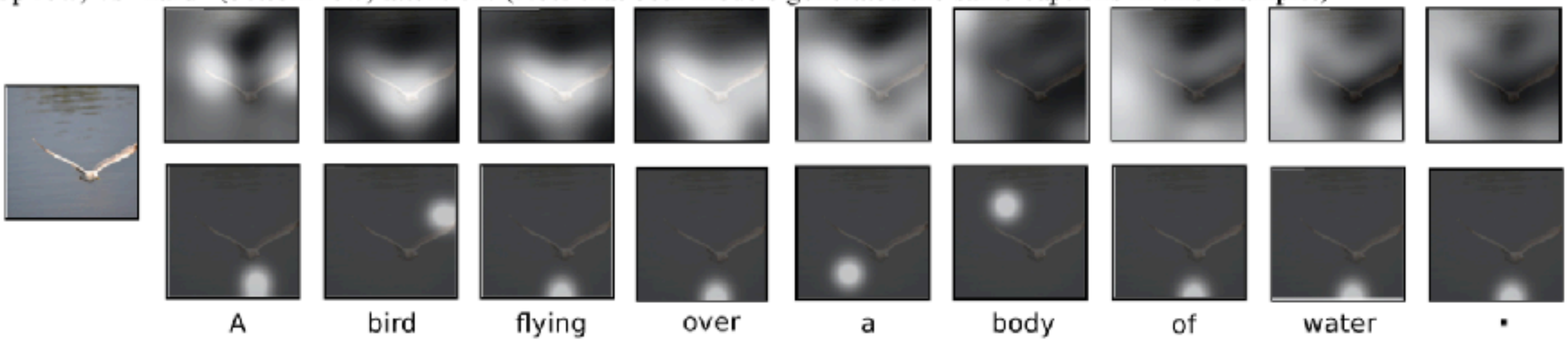
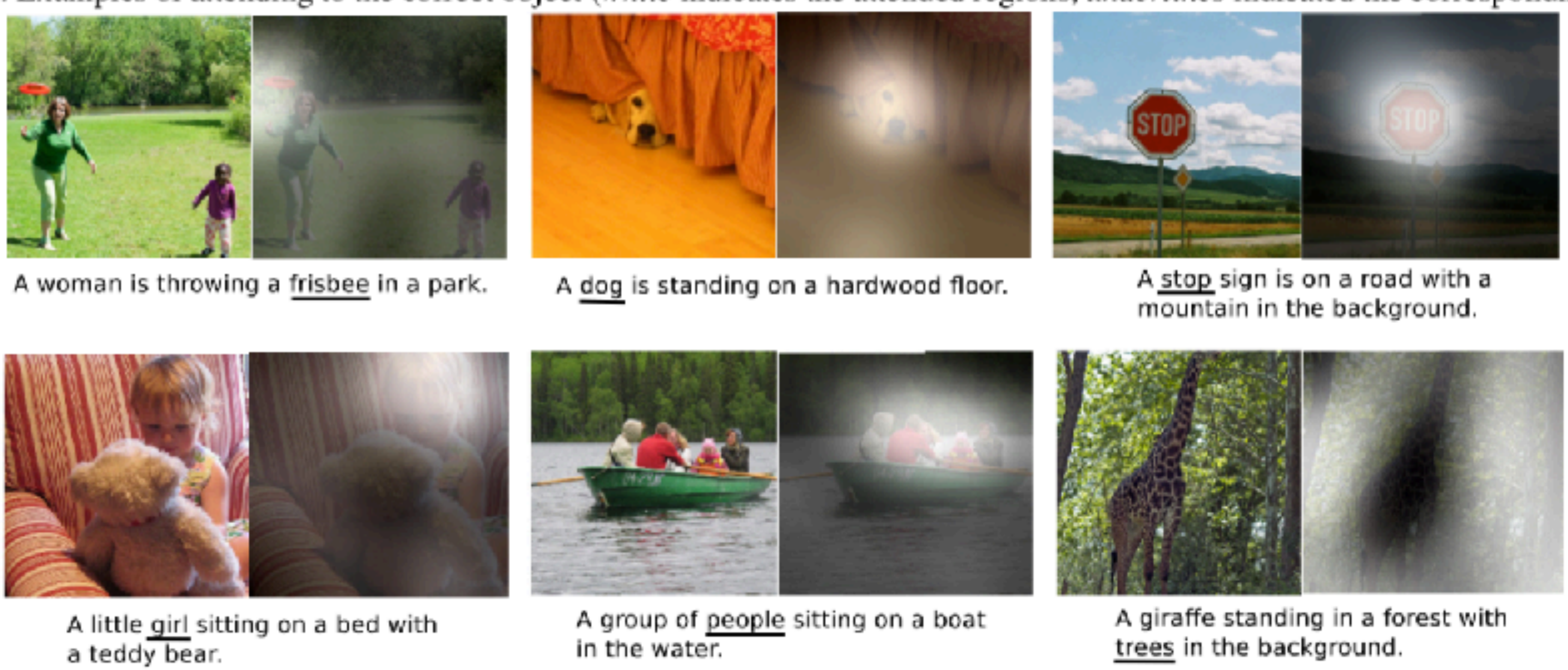


Figure 3. Examples of attending to the correct object (*white* indicates the attended regions, *underlines* indicated the corresponding word)



Gradient Descent

$$\theta^* = \arg\min_\theta \frac{1}{n}\sum_{i=1}^{n} L(y_i, f_\theta(x_i))] + \lambda D(\theta \| \theta_0)$$

- Patrick Rebeschini will introduce optimization for machine learning later in the afternoon.

- Iterative procedure:

$$\theta^{(t+1)} = \theta^{(t)} - \epsilon_t \left(\frac{1}{n}\sum_{i=1}^{n} \nabla L(y_i, f_{\theta^{(t)}}(x_i)) + \lambda \nabla D(\theta^{(t)} \| \theta_0) \right)$$

- Two questions:
  - scalability to large data sets?
  - how to compute derivatives?

Stochastic Gradient Descent

- Estimate gradient of loss using "minibatches" of data:

$$\theta^{(t+1)} = \theta^{(t)} - \epsilon_t \left(\frac{1}{|B_t|} \sum_{i \in B_t} \nabla L(y_i, f_{\theta^{(t)}}(x_i)) + \lambda \nabla D(\theta^{(t)} \| \theta_0) \right)$$

- Reduce computation cost from O(n) to O($|B_t|$).
  - More data is always better, as long as you have the compute to handle it.
- Stochastic gradients are unbiased estimates ⇒ convergence theory.
- Stochasticity can help regularise and alleviate over-fitting

Automatic Differentiation

$$\frac{\partial h_j}{\partial x_i}$$

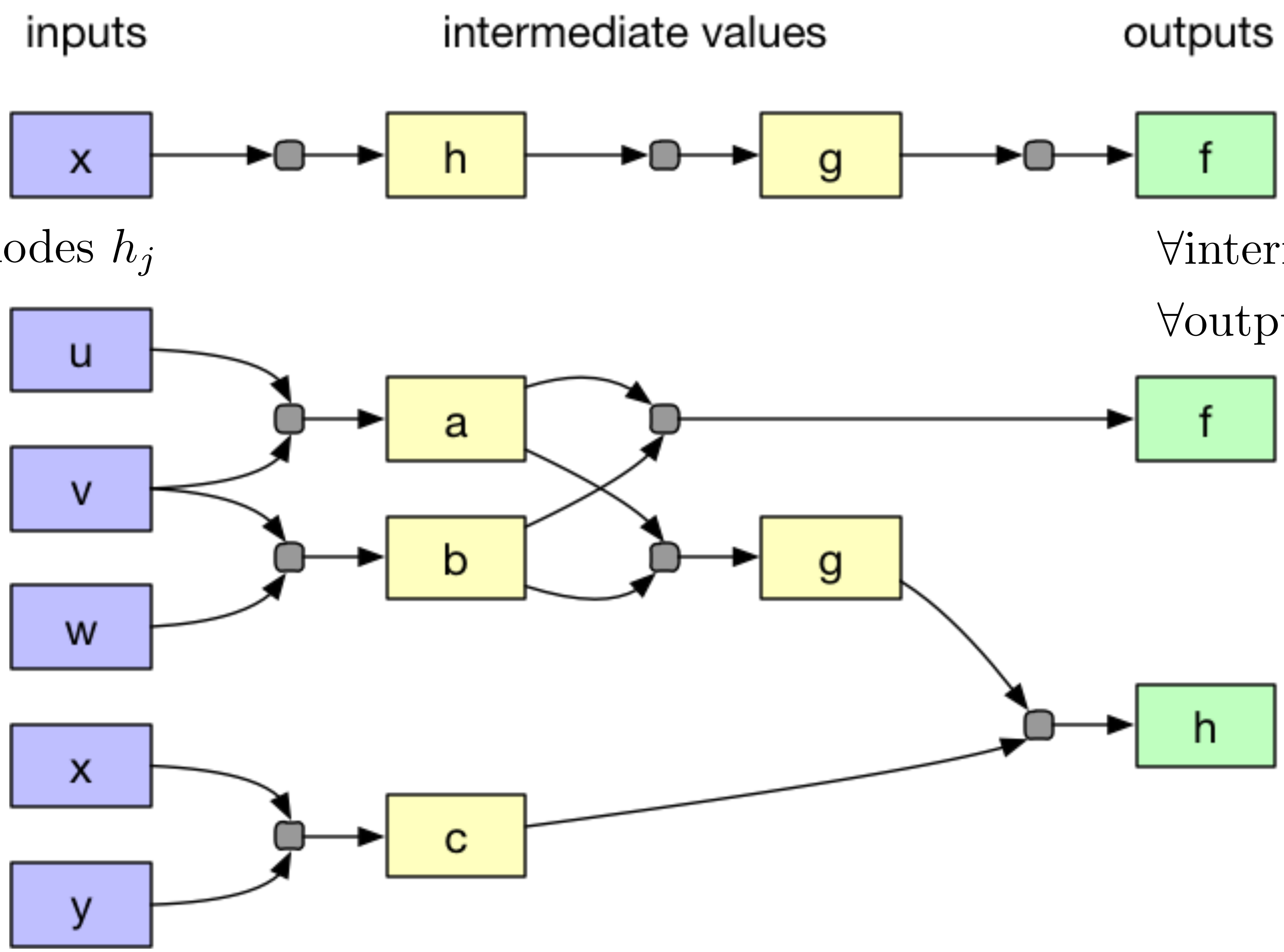
$\forall$intermediate nodes h_j

$\forall$input nodes x_i

$$\frac{\partial f_k}{\partial h_j}$$

$\forall$intermediate nodes h_j

$\forall$output nodes f_k



Automatic Differentiation

- Two major approaches: forward mode, and reverse mode AD.

$$\frac{\partial h_j}{\partial x_i}$$

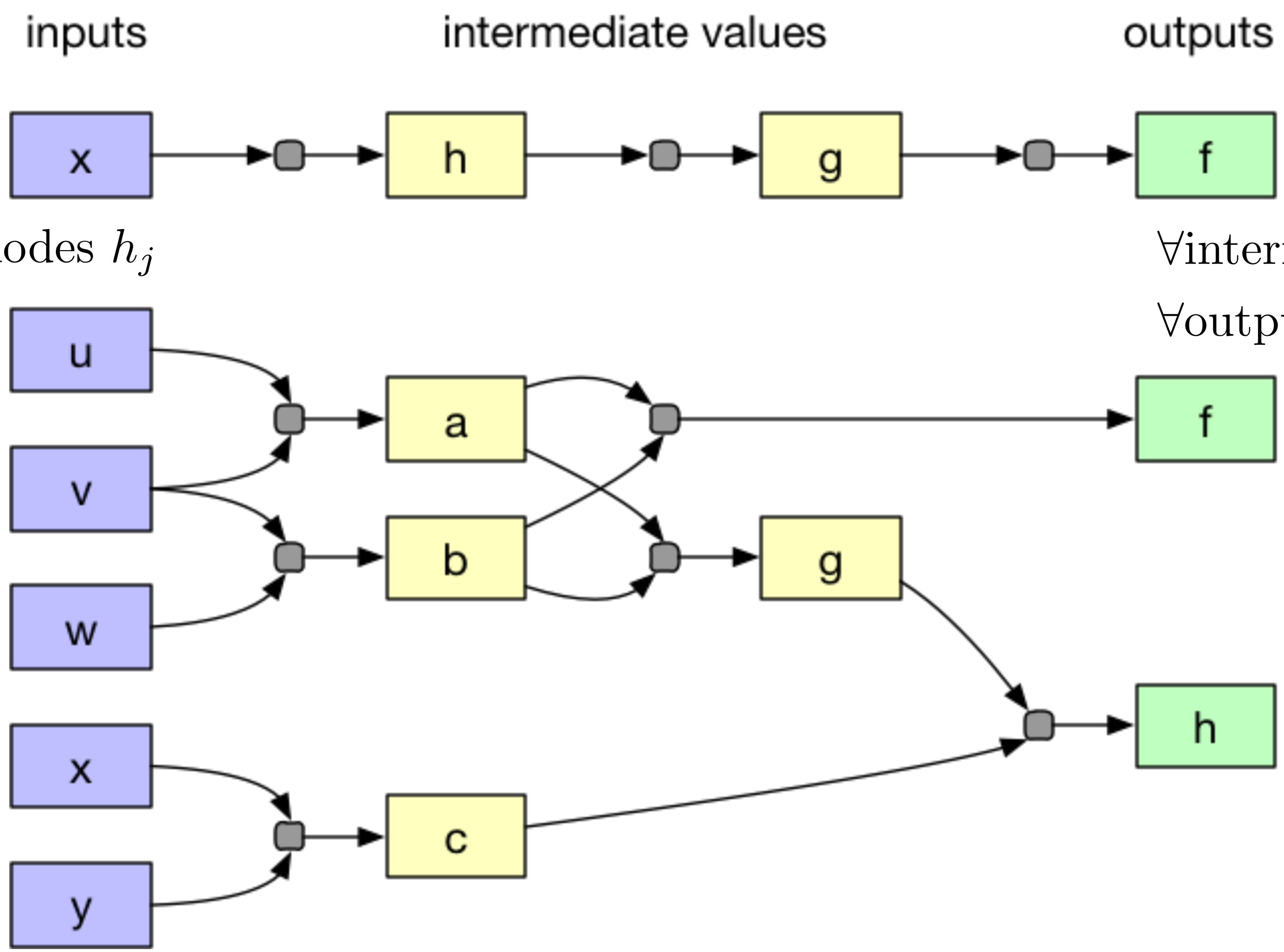
$\forall$intermediate nodes h_j

$\forall$input nodes x_i

$$\frac{\partial f_k}{\partial h_j}$$

$\forall$intermediate nodes h_j

$\forall$output nodes f_k



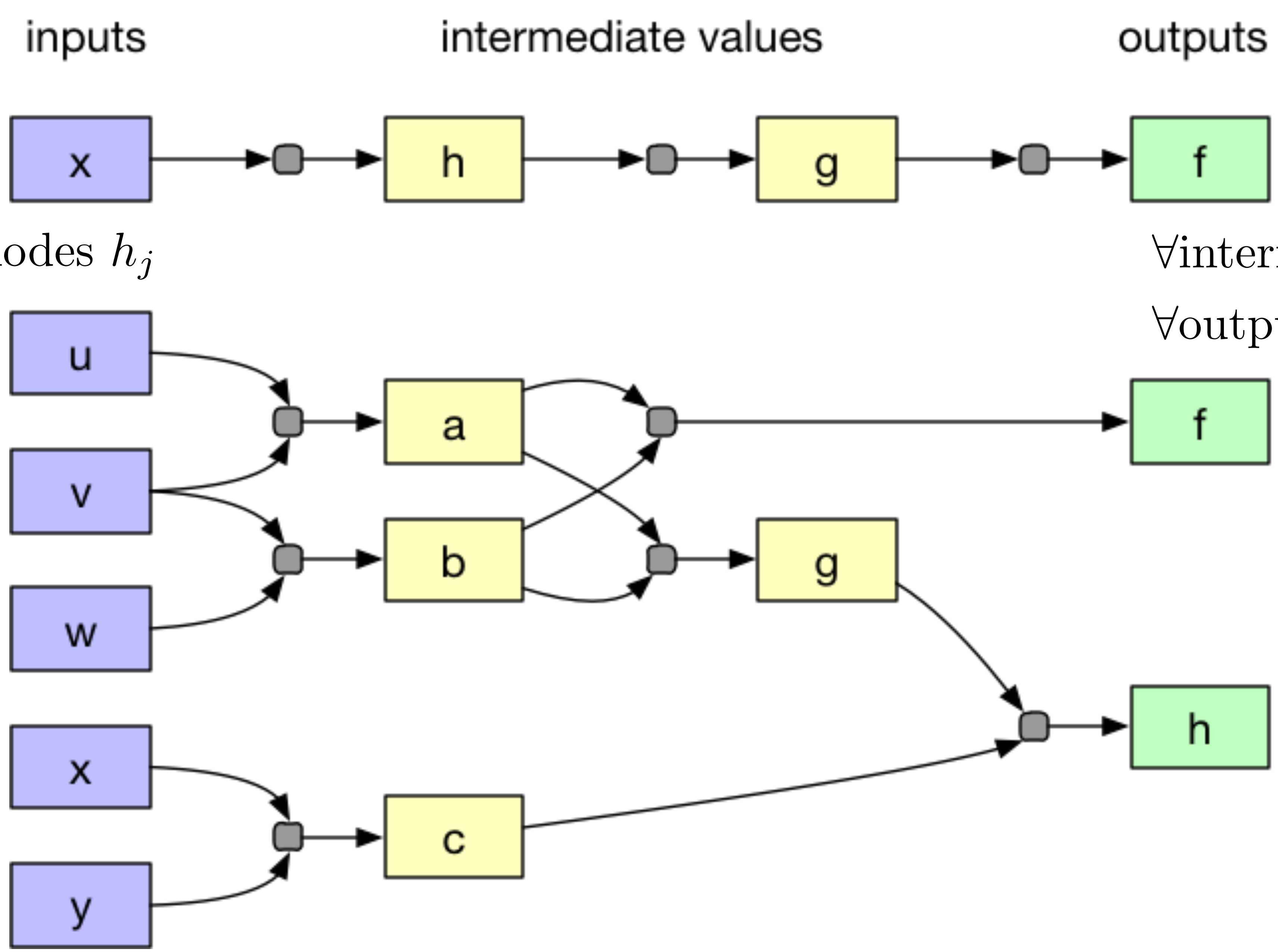
Automatic Differentiation

- Two major approaches: forward mode, and reverse mode AD.

$$\frac{\partial h_j}{\partial x_i}$$

$\forall$intermediate nodes h_j

$\forall$input nodes x_i



$$\frac{\partial f_k}{\partial h_j}$$

$\forall$intermediate nodes h_j

$\forall$output nodes f_k

- Forward: O(#inputs*#nodes). Reverse: O(#outputs*#nodes).

Infrastructure

- Infrastructure support critical to deep learning (and ML in general):
  - **software frameworks** allow fast model building, automating away most low-level operations.
  - Culture of sharing code via **open source** releases.
  - **hardware** allows fast training, and scalable productionisation.
  - **large datasets** and **difficult challenges** pushing frontier forward.

VAE in Keras/TensorFlow
Colab Demo

https://goo.gl/yWaM9P

“Deep Learning est mort. Vive Differentiable Programming!” - Yann LeCun

Yeah, Differentiable Programming is little more than a rebranding of the modern collection Deep Learning techniques, the same way Deep Learning was a rebranding of the modern incarnations of neural nets with more than two layers.

The important point is that people are now building a new kind of software by assembling networks of parameterized functional blocks and by training them from examples using some form of gradient-based optimization….It’s really very much like a regular program, except it’s parameterized, automatically differentiated, and trainable/optimizable.

More Resources

- Tutorials and courses:
  - http://www.cs.ucl.ac.uk/current_students/syllabus/compgi/compgi22_advanced_deep_learning_and_reinforcement_learning/
  - https://www.coursera.org/learn/machine-learning
  - http://videolectures.net/deeplearning2015_salakhutdinov_deep_learning/
  - https://www.youtube.com/watch?v=F1ka6a13S9I
- Summer schools: MLSS, DLSS, RLSS
- Conferences: NIPS, ICML, UAI, AISTATS
- Journals: JMLR
- ArXiv