Deep Learning

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

~~Deep Learning~~ Differential Programming?

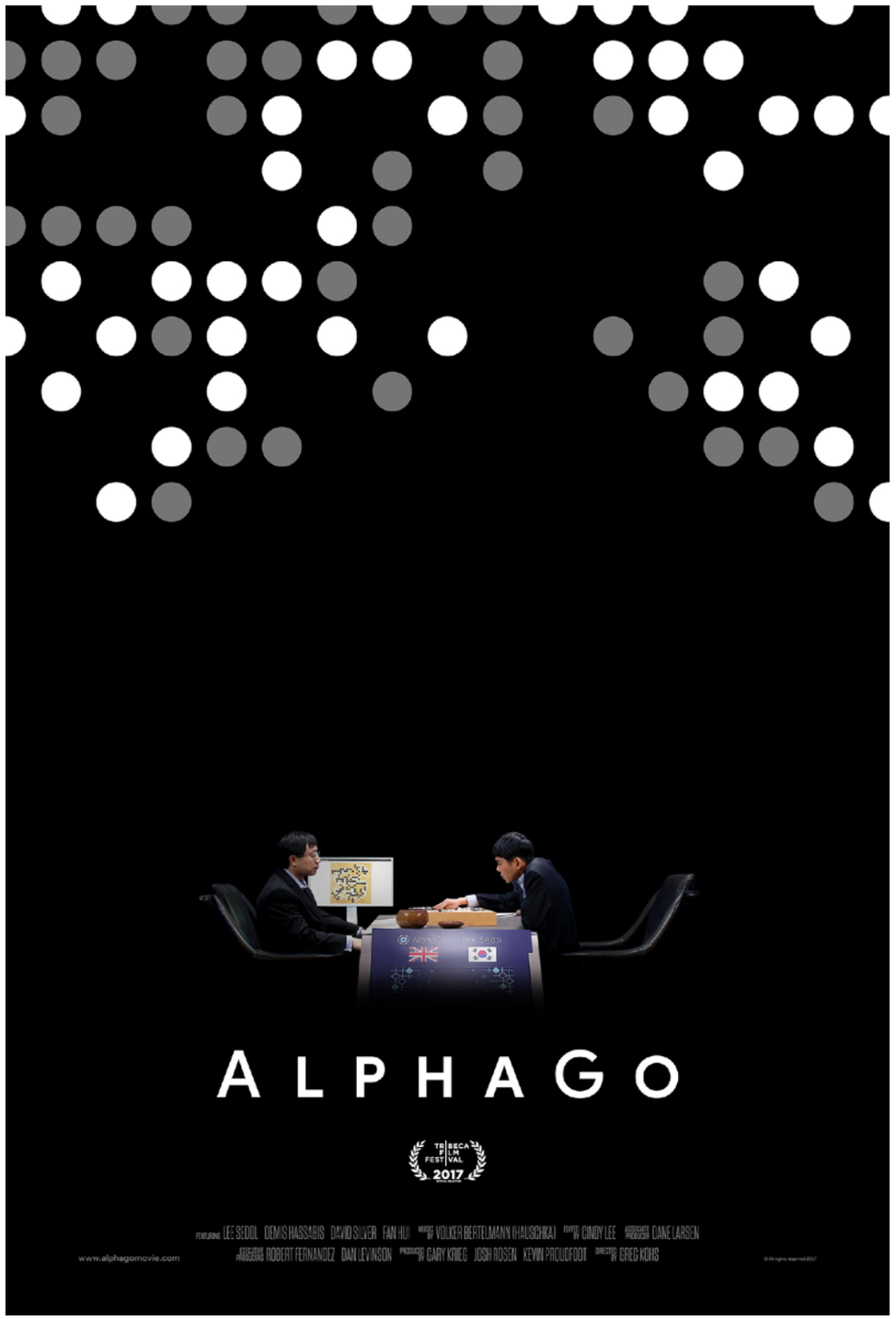
Yee Whye Teh (Oxford Statistics & DeepMind)
http://csml.stats.ox.ac.uk/people/teh

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

Yeah, Differentiable Programming is little more than a rebranding of the modern collection of 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.

Artificial Intelligence



Artificial Intelligence

- AI problems are difficult and complex.
  - Impossible to manually programme explicit solutions.

- Modern deep learning approach developed to tackle this difficulty and complexity.
  - We "programme" a solution space by specifying **neural network architecture** and **objective** function.
  - The system then searches in solution space by optimizing (**learning**) on **large data sets**, taking advantage of **modern computing hardware**.

Deep Learning Infrastructure

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

Platforms



Microsoft Azure

aws

Frameworks

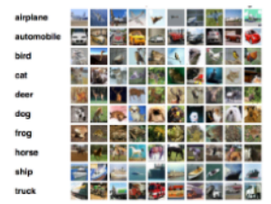
scikit learn

Caffe K Keras

Microsoft CNTK mxnet theano

PYTORCH Caffe2

Datasets

Caltech 101 IMAGENET

LDC Linguistic Data Consortium

UCI Machine Learning Repository
Center for Machine Learning and Intelligent Systems

WMT Workshop 2014

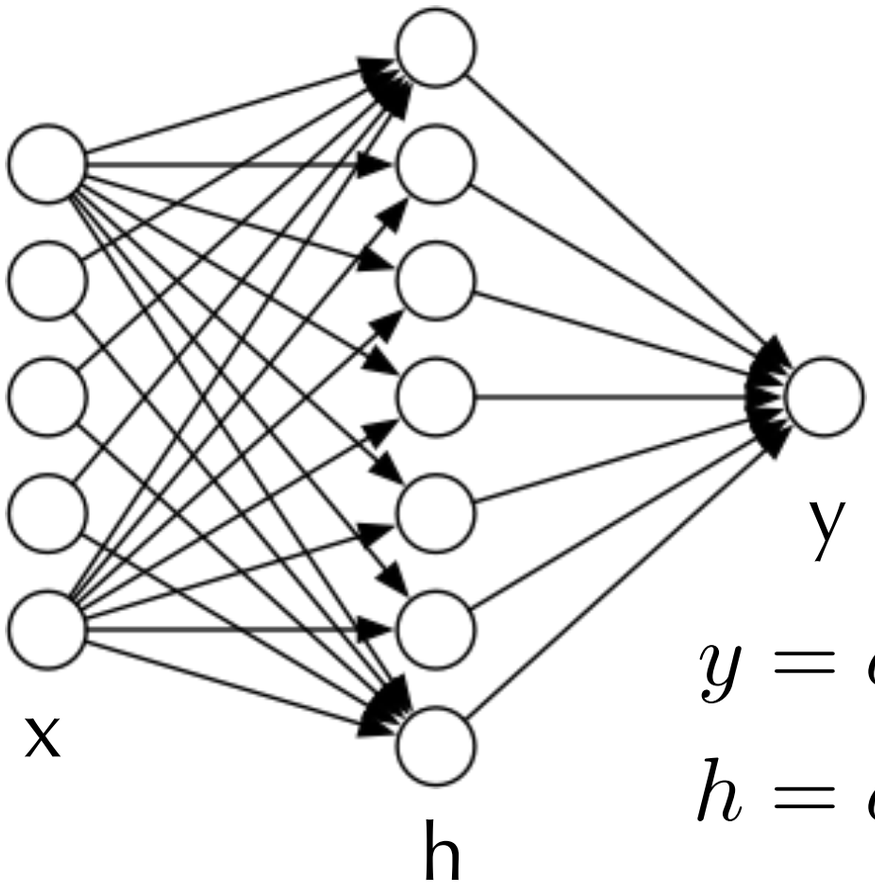
Learning Parameterised Functions

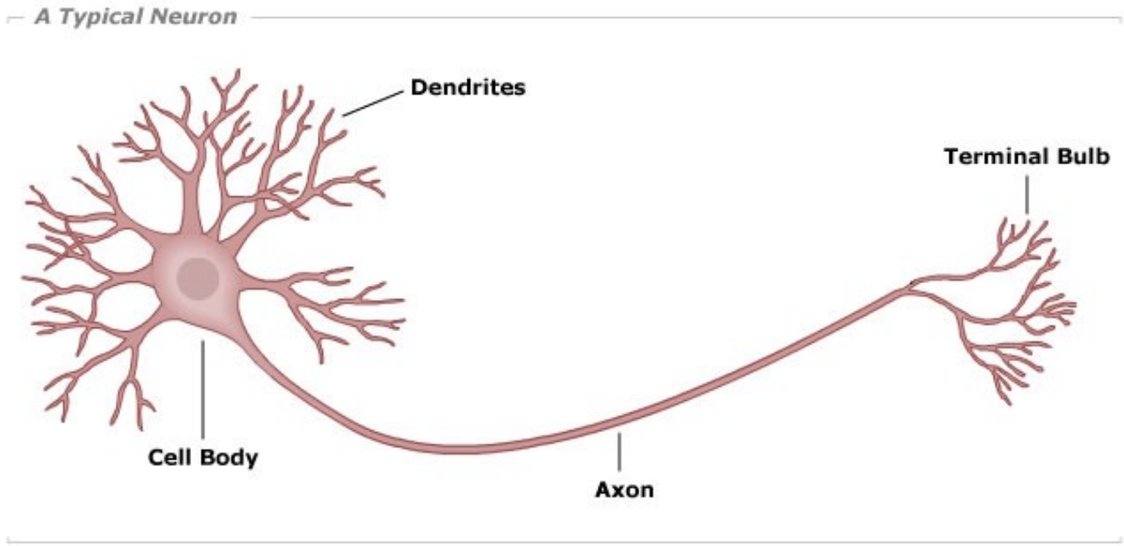
$$\theta^* = \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \lambda \|\theta\|$$

- 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** and **scalable**.

Building Blocks

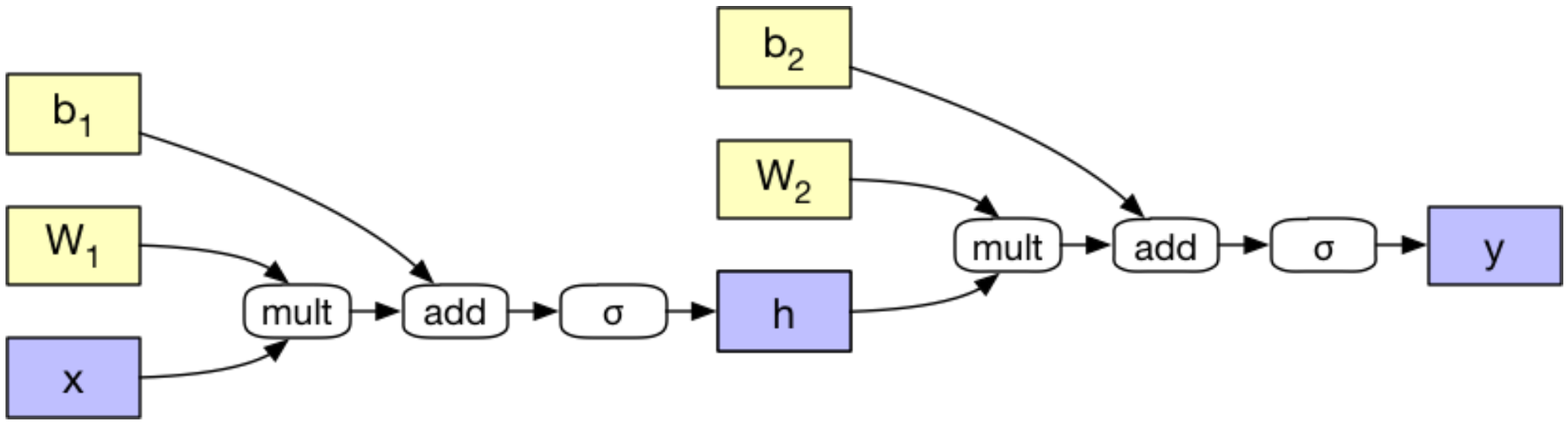
Neural Networks





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

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



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

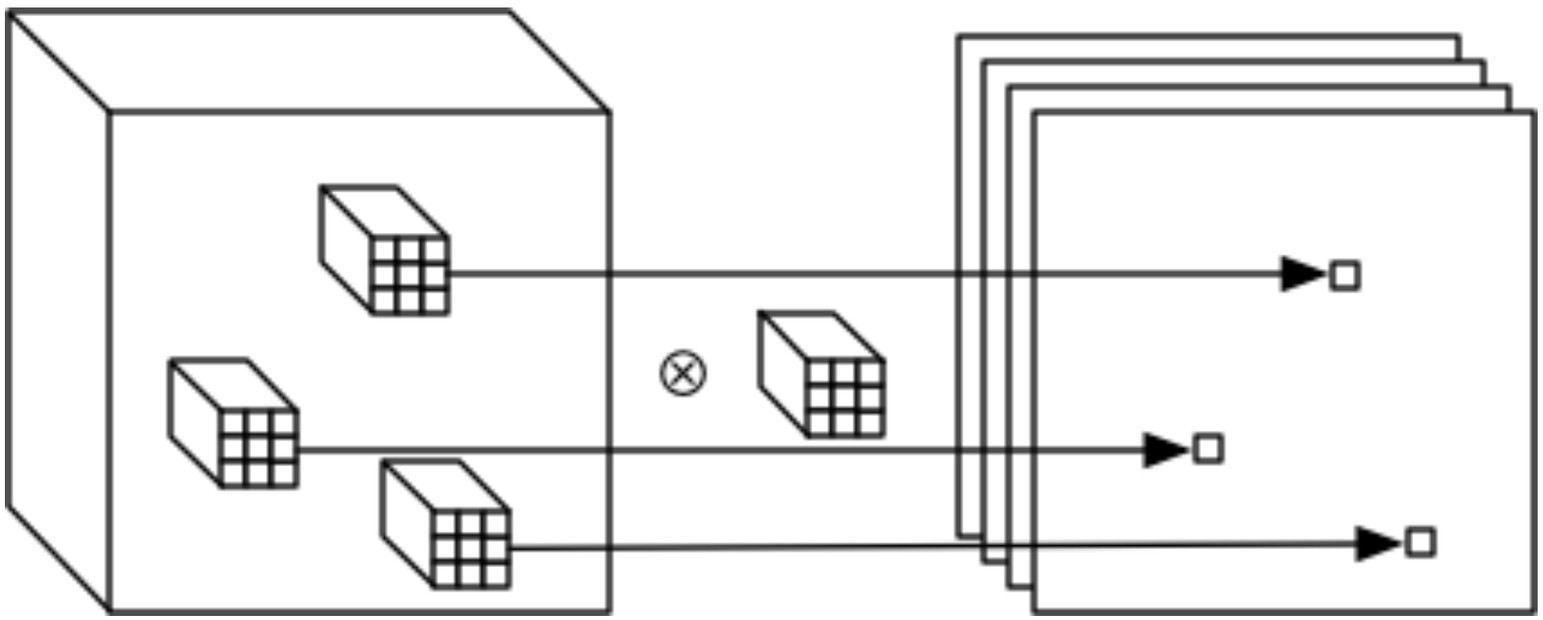
$$\text{CrossEntropy}(t, y) = \sum_i t_i \log y_i$$

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

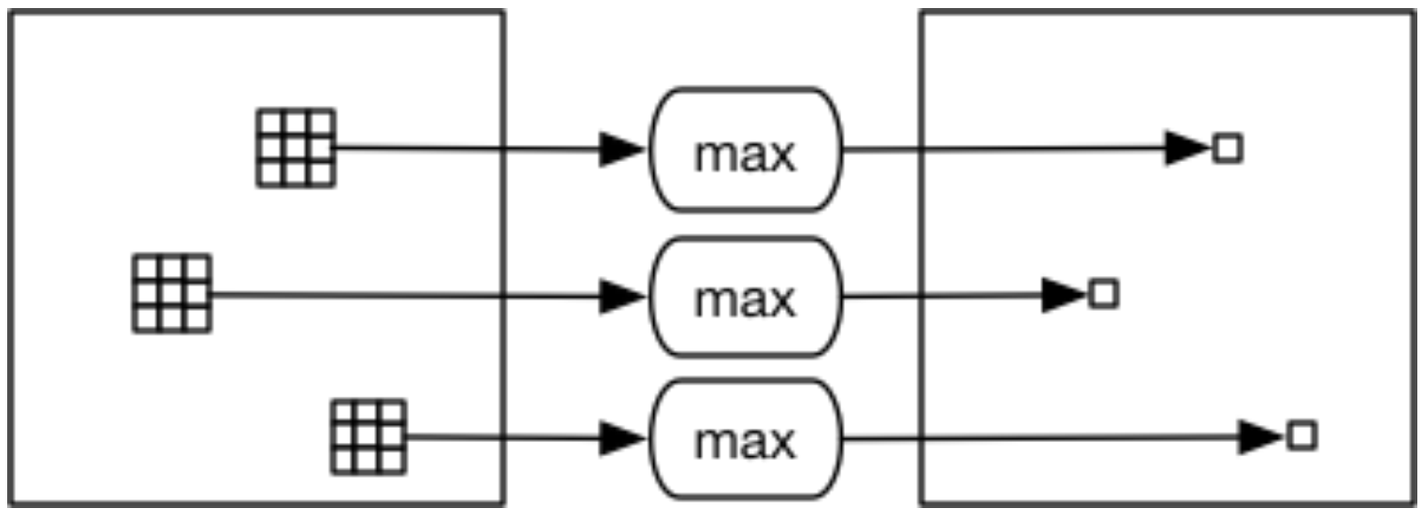
$$\text{Hinge}(t, y) = \max(0, 1 - t \cdot y)$$

Building Blocks

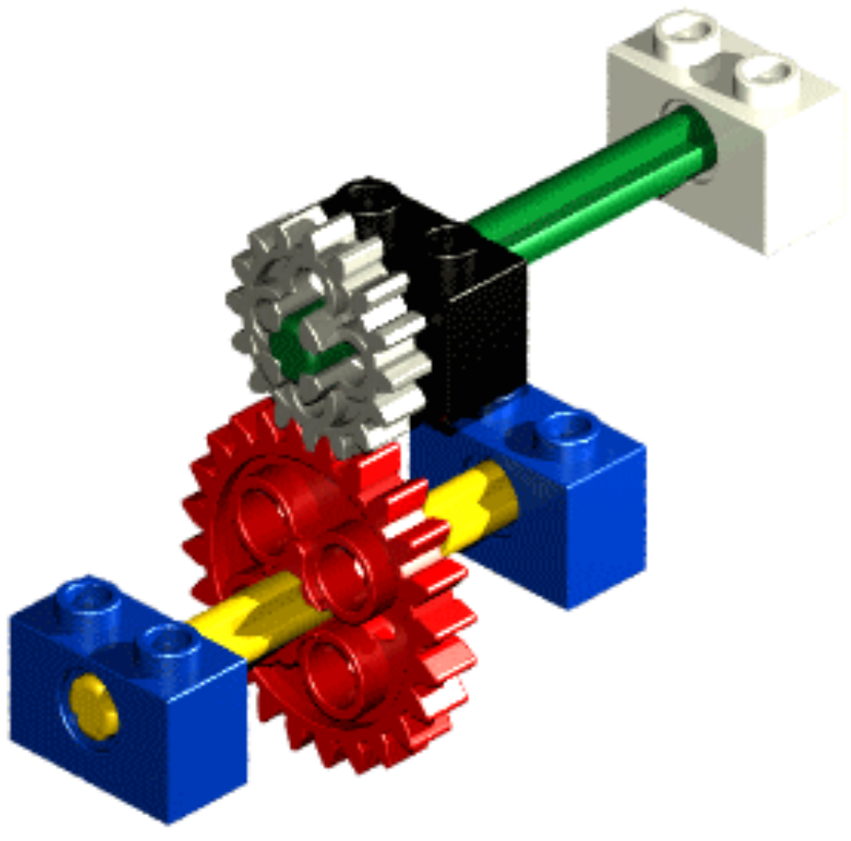
- Convolution



- max pooling



Optimization



Gradient Descent

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

- 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

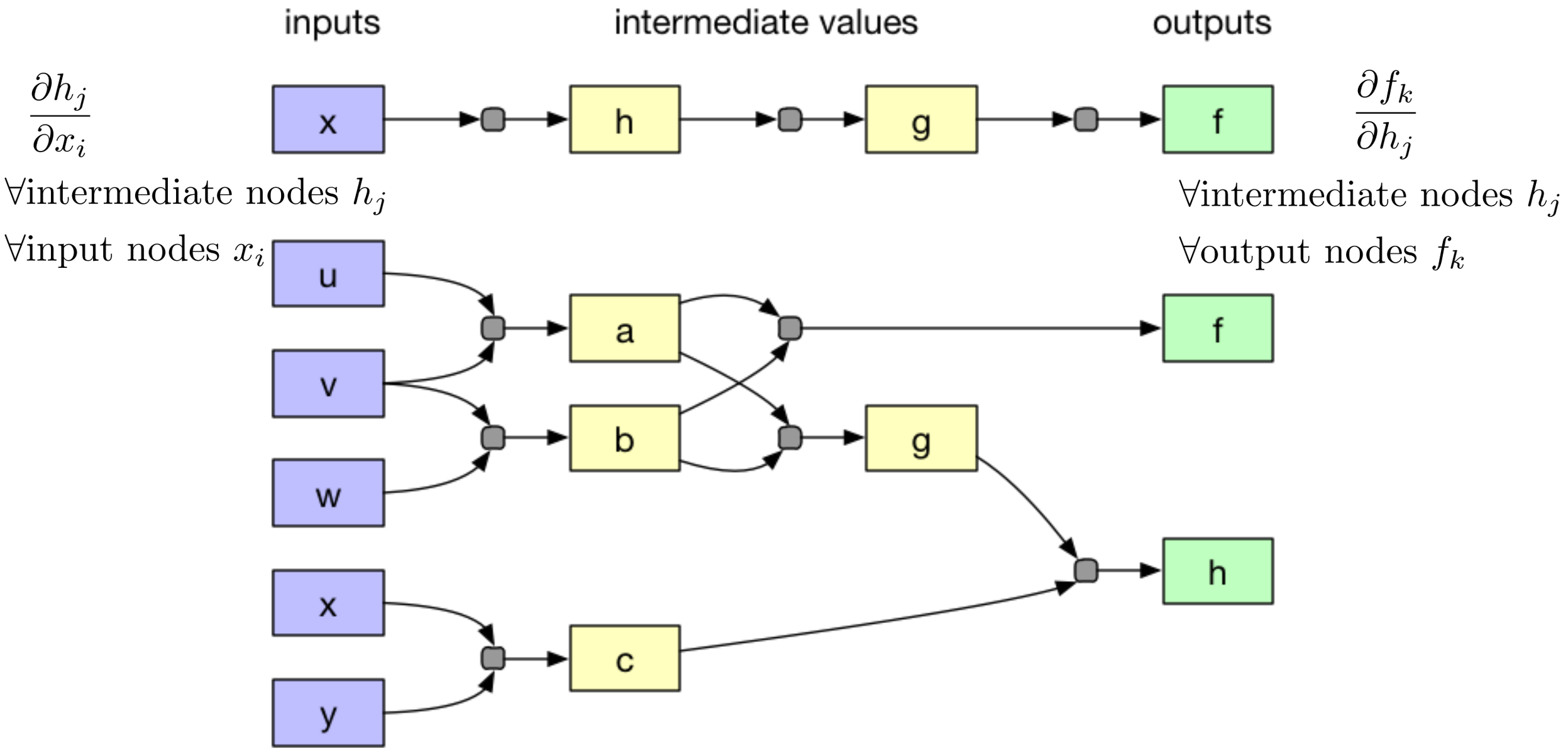
- Estimate gradient of loss using "minibatches" of data:

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

- 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

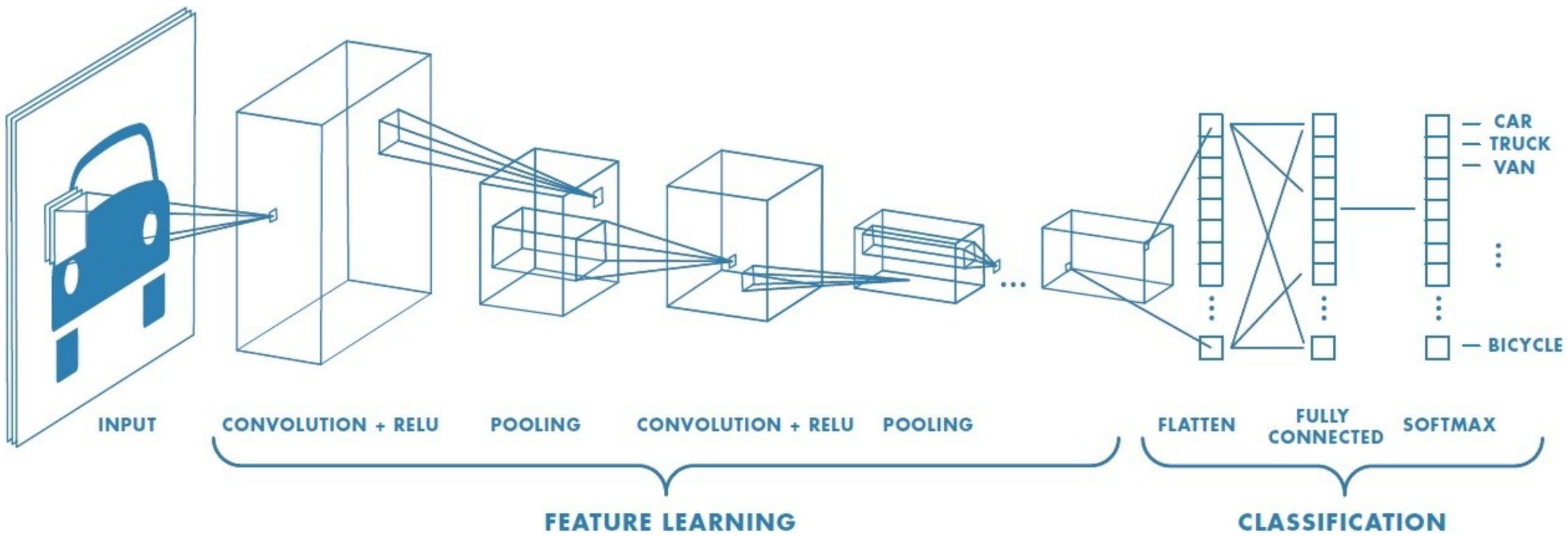
- Two major approaches: forward mode, and reverse mode AD.



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

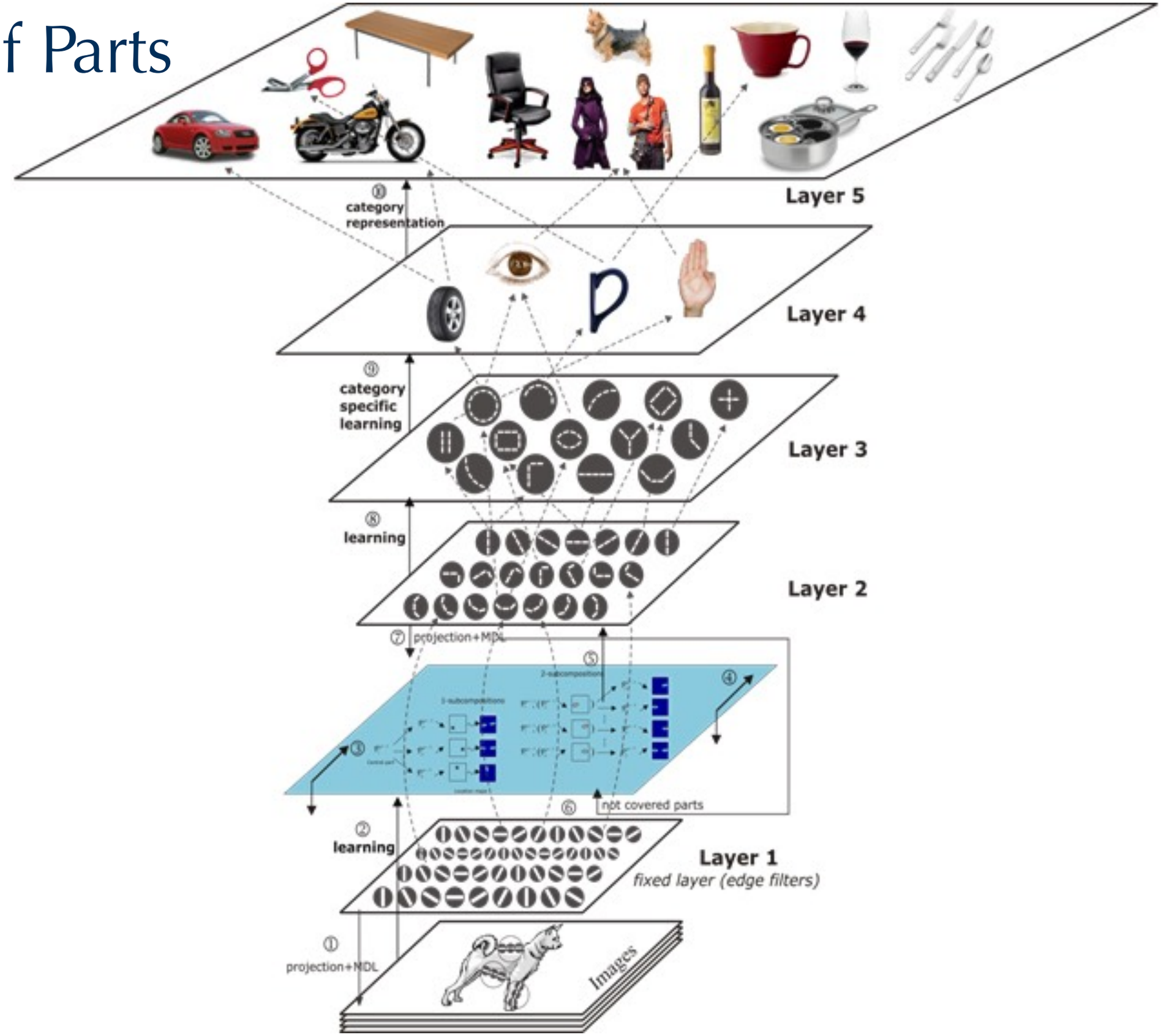
Some Examples

Convolutional Networks (Convnets)



- Both filter banks and layers are 4D tensors (arrays of numbers).

Hierarchy of Parts



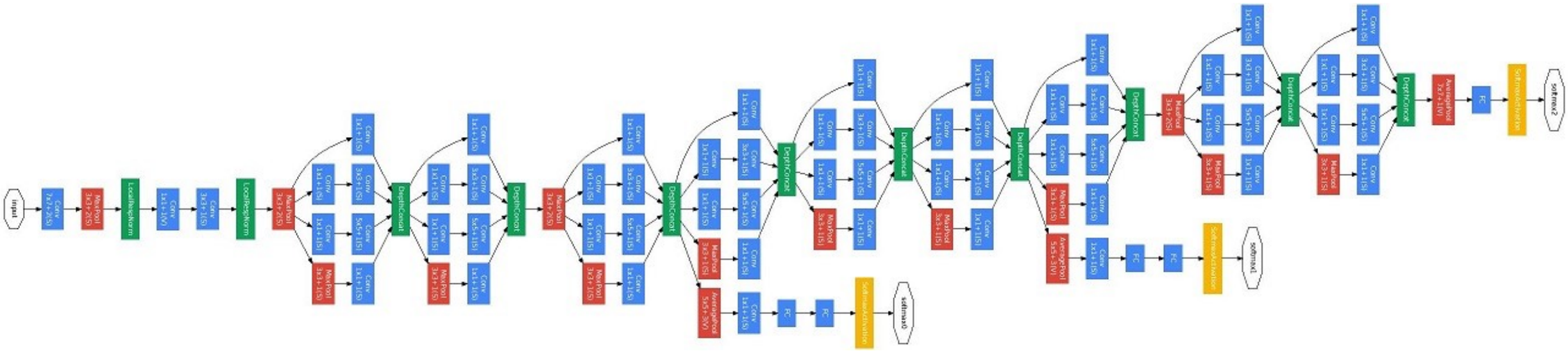
Convnet Implementation in Keras

```
(x_train, y_train), (x_test, y_test) =
        cifar10.load_data()
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
y_train = keras.utils.to_categorical(y_train,
        num_classes)
y_test = keras.utils.to_categorical(y_test,
        num_classes)
```

```
model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
        input_shape=x_train.shape[1:]))
model.add(Activation('relu'))
model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes))
model.add(Activation('softmax'))
```

```
opt = keras.optimizers.rmsprop(lr=0.0001,
        decay=1e-6)
model.compile(loss='categorical_crossentropy',
        optimizer=opt,
        metrics=['accuracy'])
model.fit(x_train, y_train,
        batch_size=batch_size,
        epochs=epochs,
        validation_data=(x_test, y_test),
        shuffle=True)
```

GoogLeNet Architecture

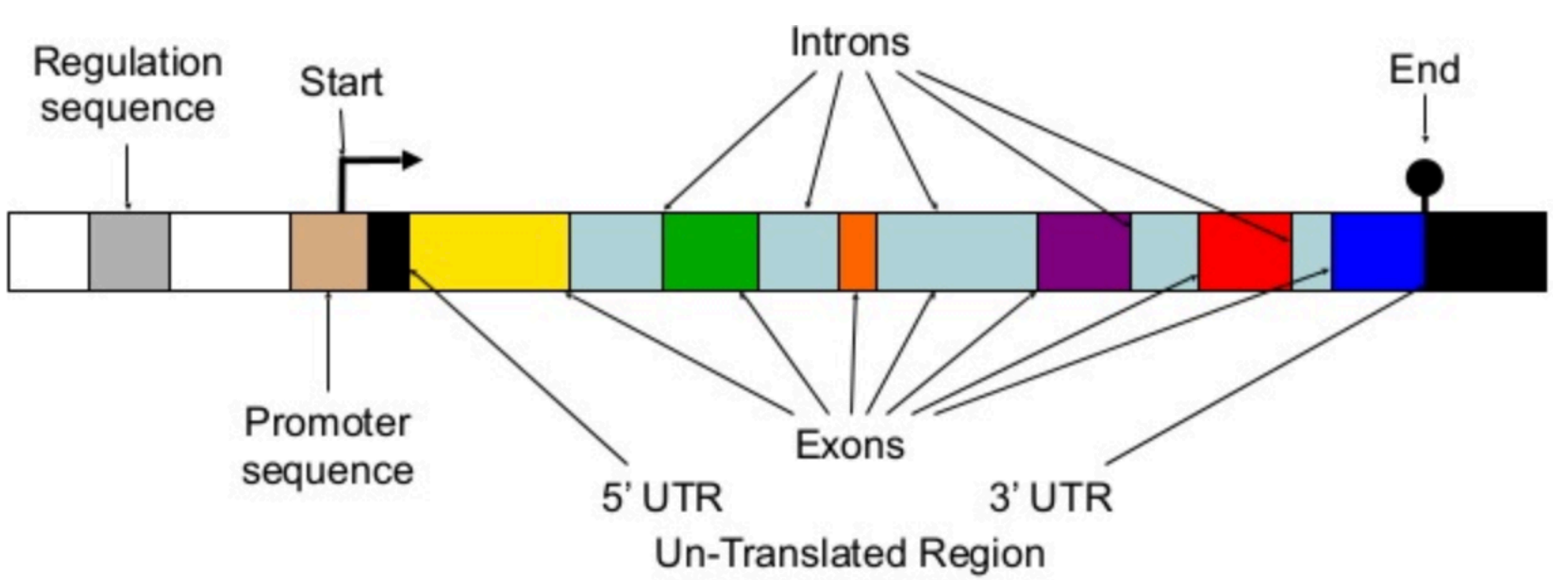


Sequence Models

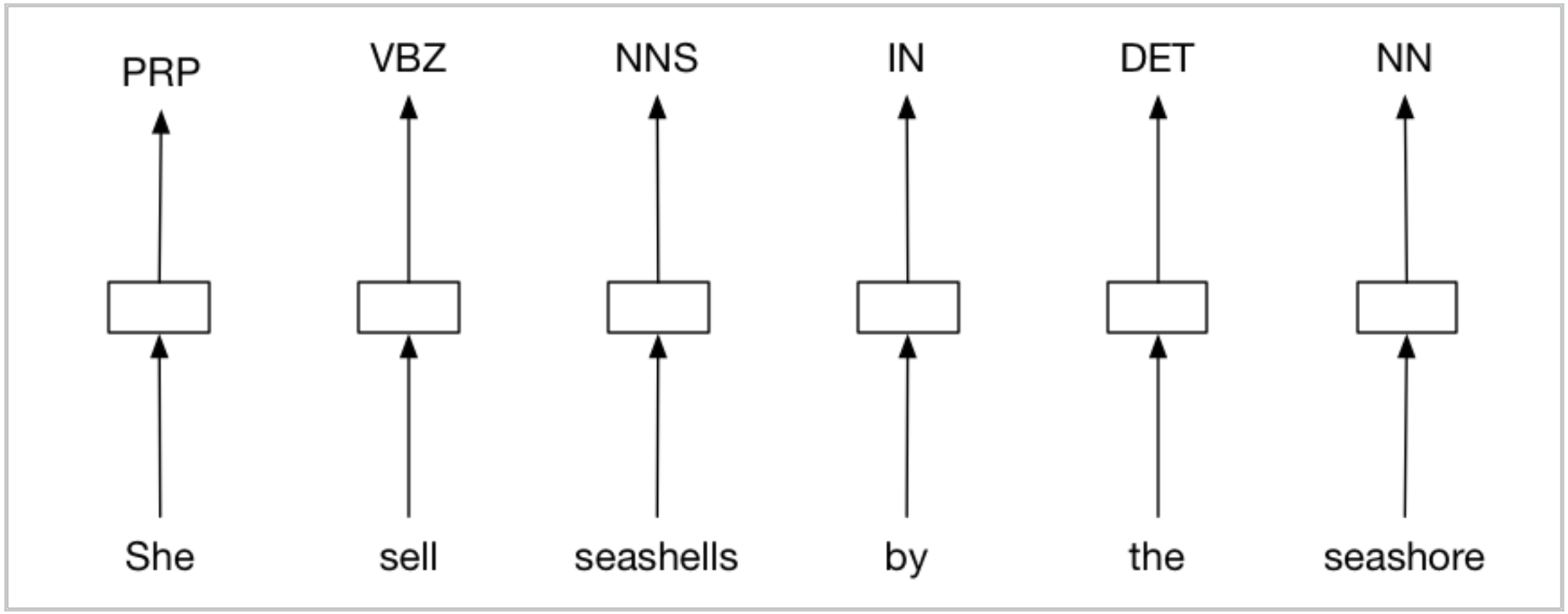
- Natural language processes

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

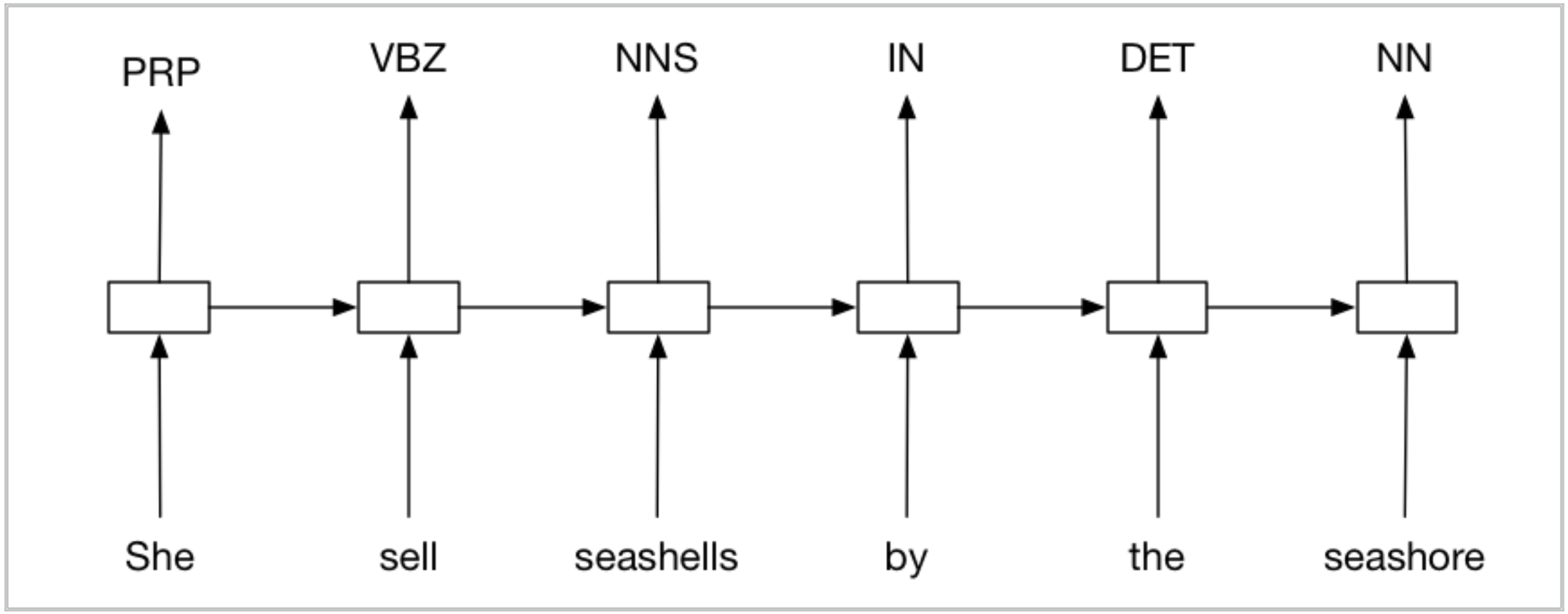
- Genomics



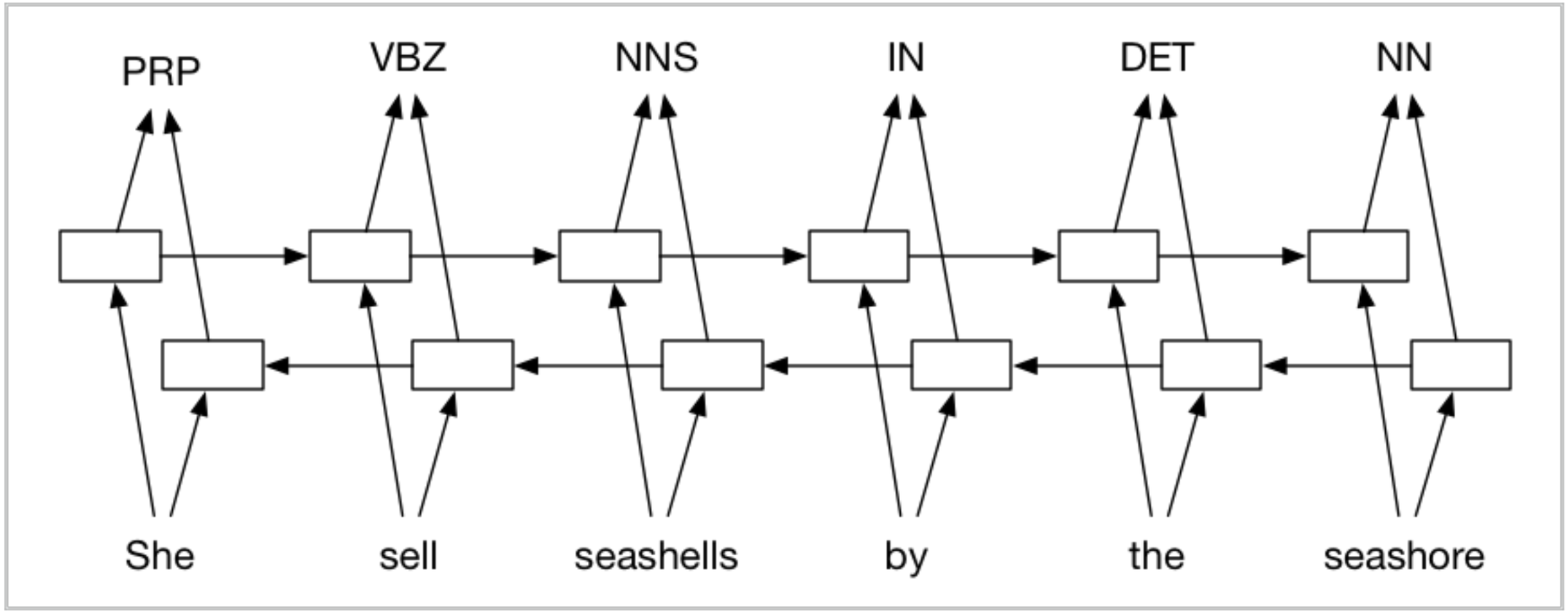
Recurrent Neural Networks



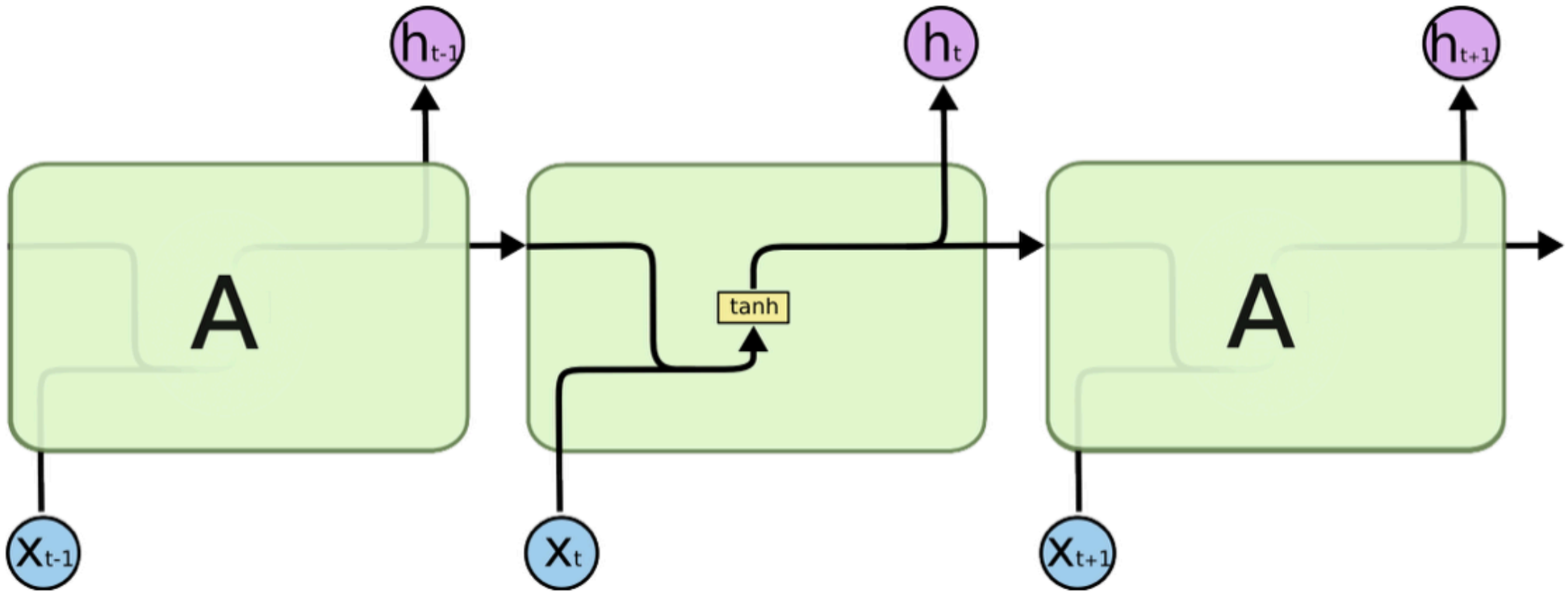
Recurrent Neural Networks



Recurrent Neural Networks



Recurrent Neural Networks

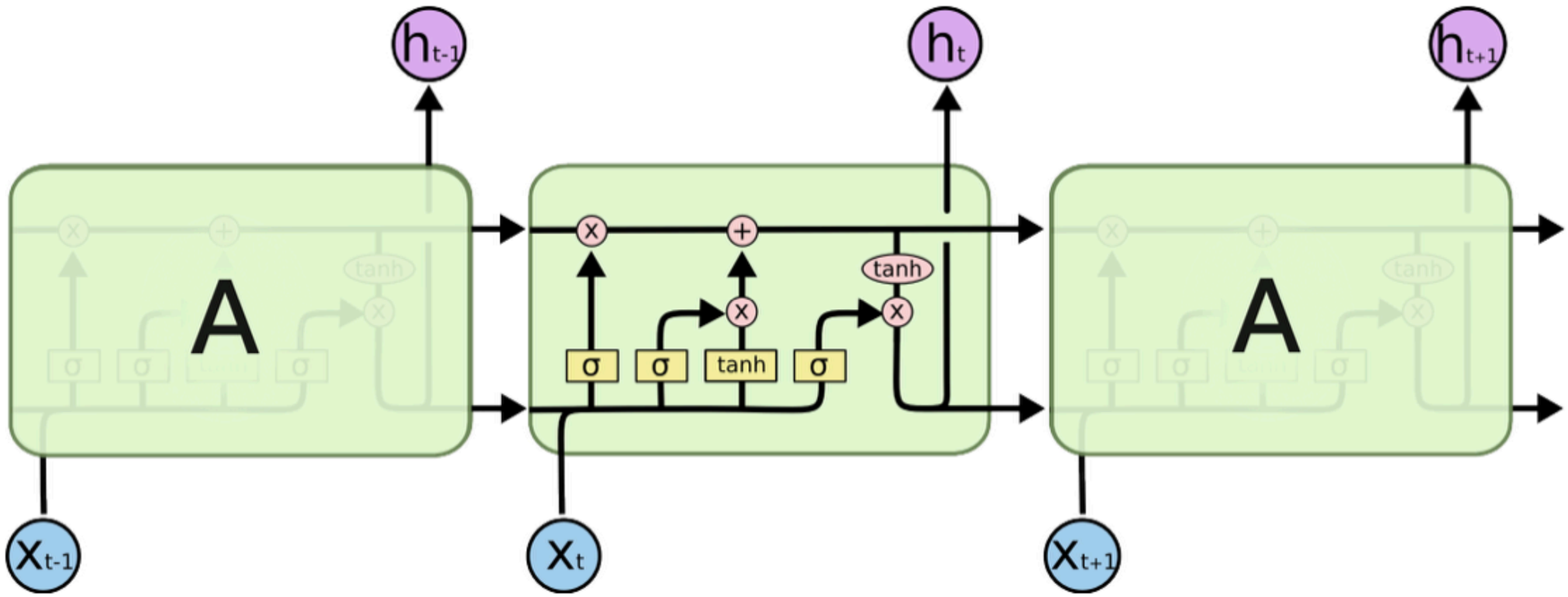


$$h_t = \sigma(W_h z_t + b_h)$$

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

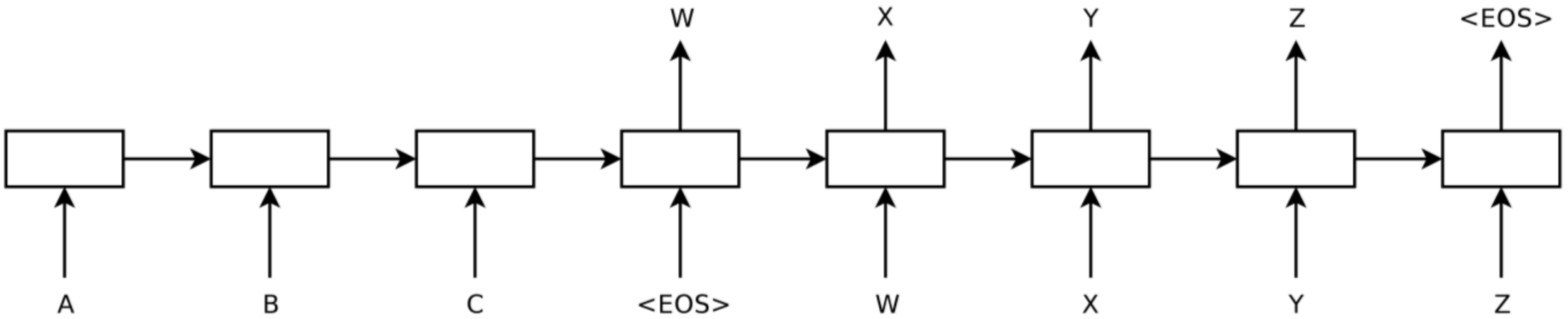
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

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

Attention

It is **fun** and **easy** to do sentiment analysis! 

I **don't like** reading all of the **negative** Tweets! 

Attention

Key-value memory store

Query Q

K_1	V_1
K_2	V_2
K_3	V_3
K_4	V_4

Attention

Key-value memory store

Query Q

K_1	V_1
K_2	V_2
K_3	V_3
K_4	V_4

Match quality: $m_i = QK_i^\top$

Attention mask: $(a_1, \ldots, a_n) = \text{softmax}(m_1, \ldots, m_n)$

Output: $\sum_{i=1}^{n} a_i V_i$

Attention

Key-value memory store

Query Q

K_1	V_1
K_2	V_2
K_3	V_3
K_4	V_4

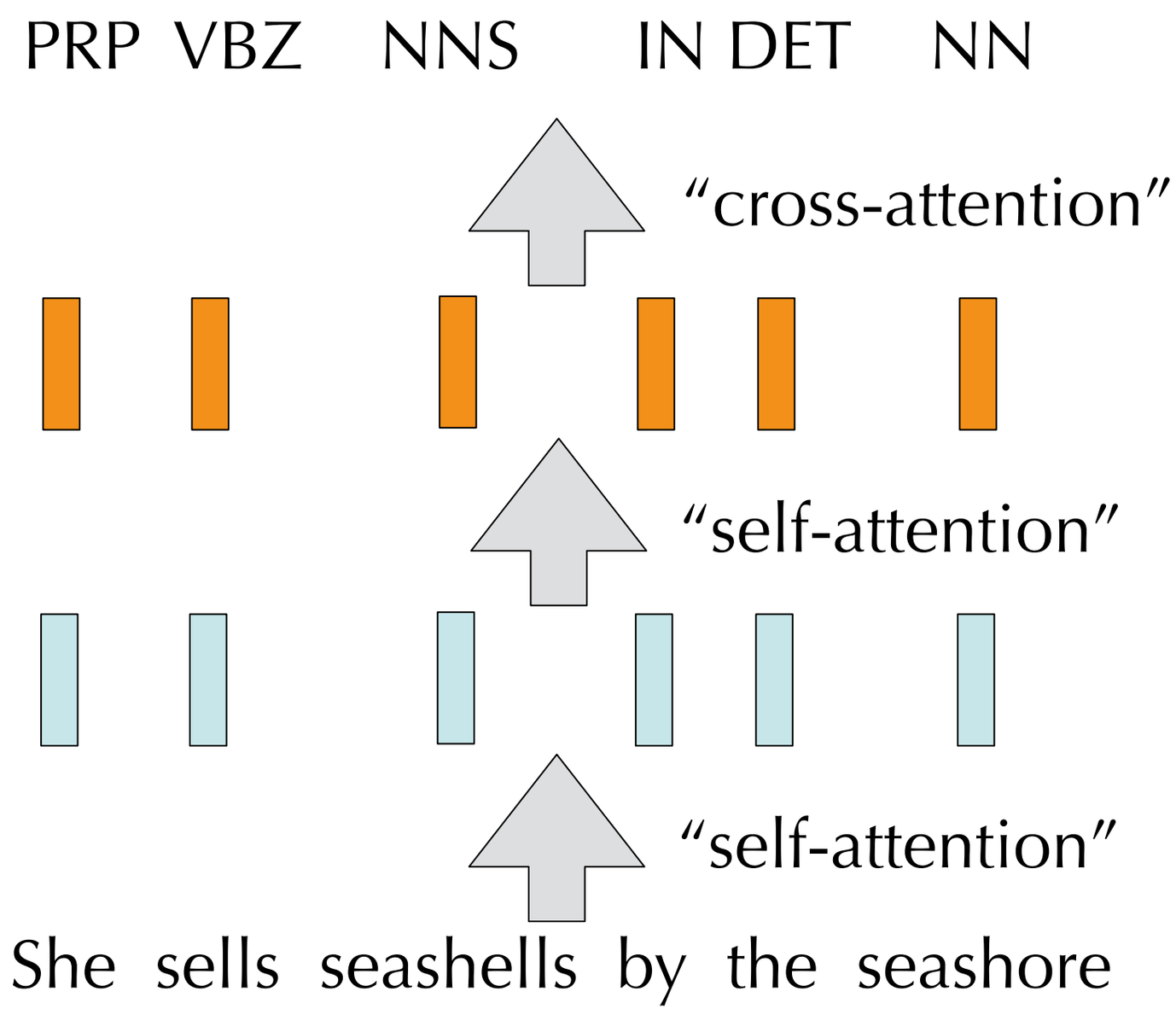
Match quality: $m_i = QK_i^\top$

Attention mask: $(a_1, \ldots, a_n) = \text{softmax}(m_1, \ldots, m_n)$

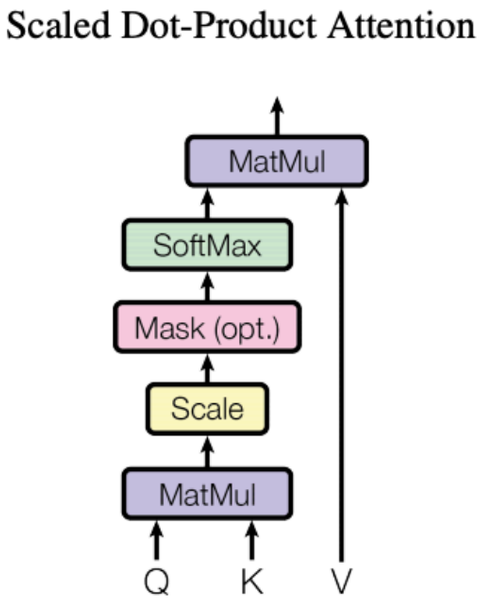
Output: $\sum_{i=1}^{n} a_i V_i$

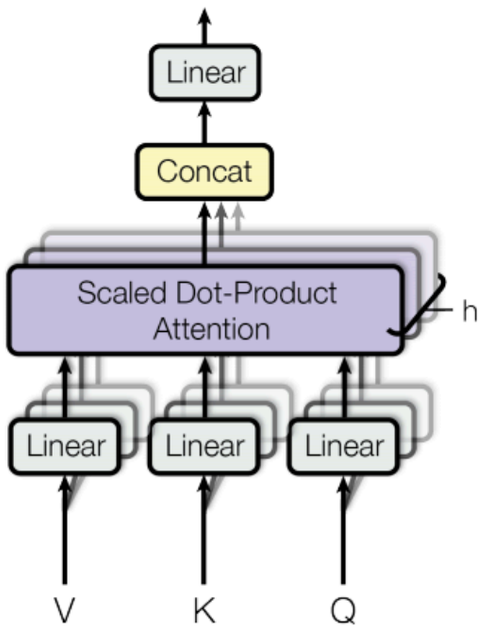
$$\text{Attention}(Q, K, V) = \text{softmax}\left(QK^\top/\sqrt{d}\right)V$$

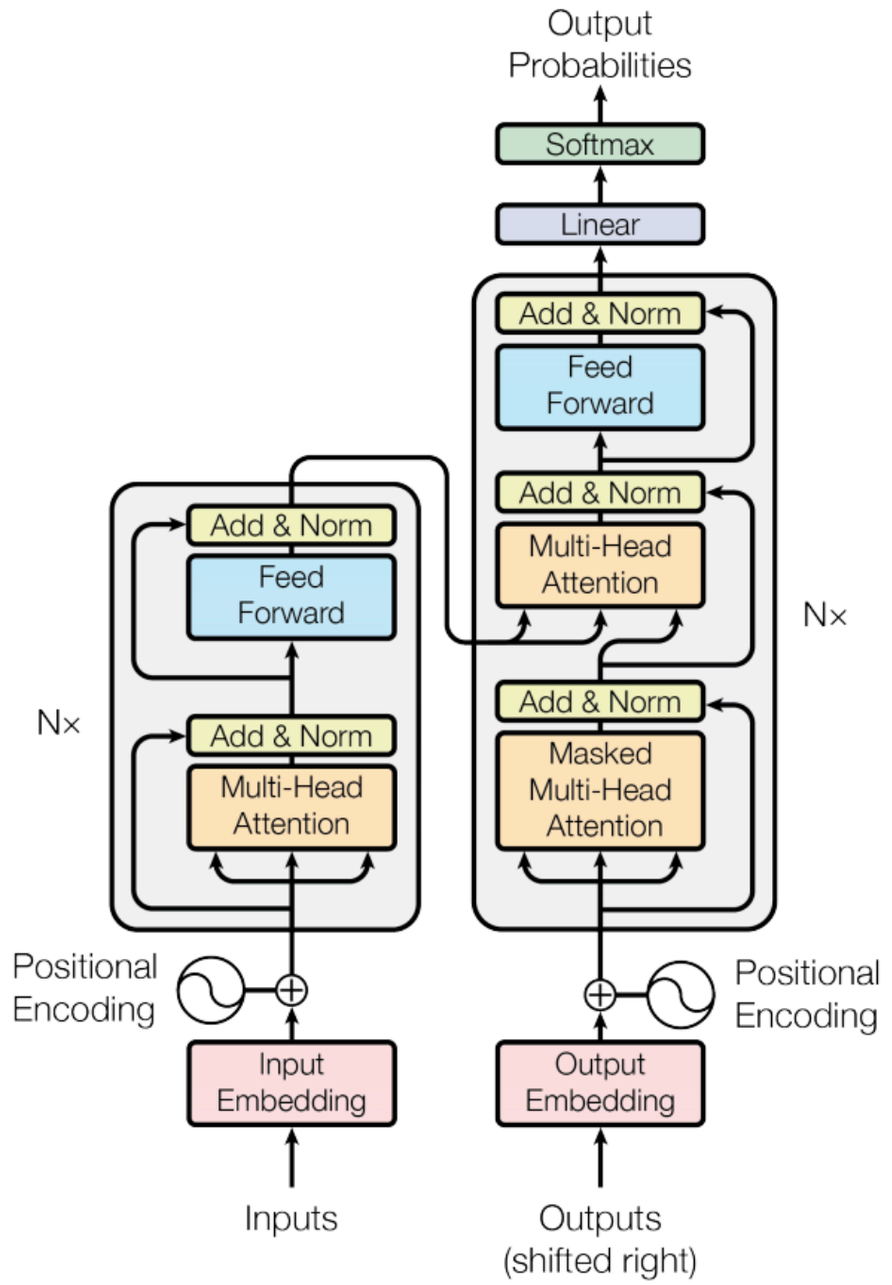
Attention


PRP VBZ NNS IN DET NN
"cross-attention"
"self-attention"
"self-attention"
She sells seashells by the seashore

Attention







https://papers.nips.cc/paper/7181-attention-is-all-you-need

Language Modelling

SYSTEM PROMPT (HUMAN-WRITTEN)

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

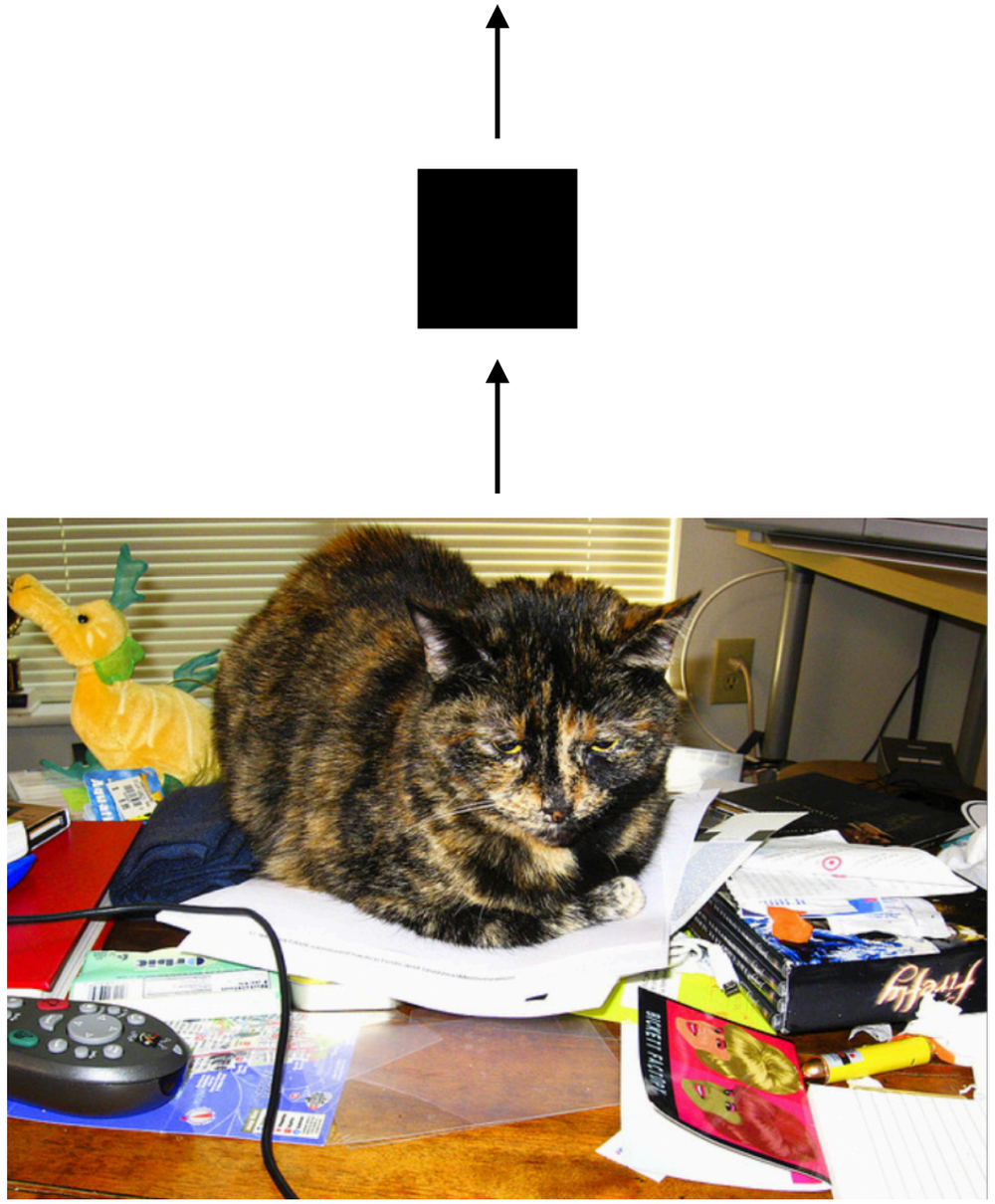
MODEL COMPLETION (MACHINE-WRITTEN, 10 TRIES)

```
The scientist named the population, after their distinctive horn,
Ovid's Unicorn. These four-horned, silver-white unicorns were
previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this
odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of
La Paz, and several companions, were exploring the Andes
Mountains when they found a small valley, with no other animals
or humans. Pérez noticed that the valley had what appeared to be
a natural fountain, surrounded by two peaks of rock and silver
snow.
```

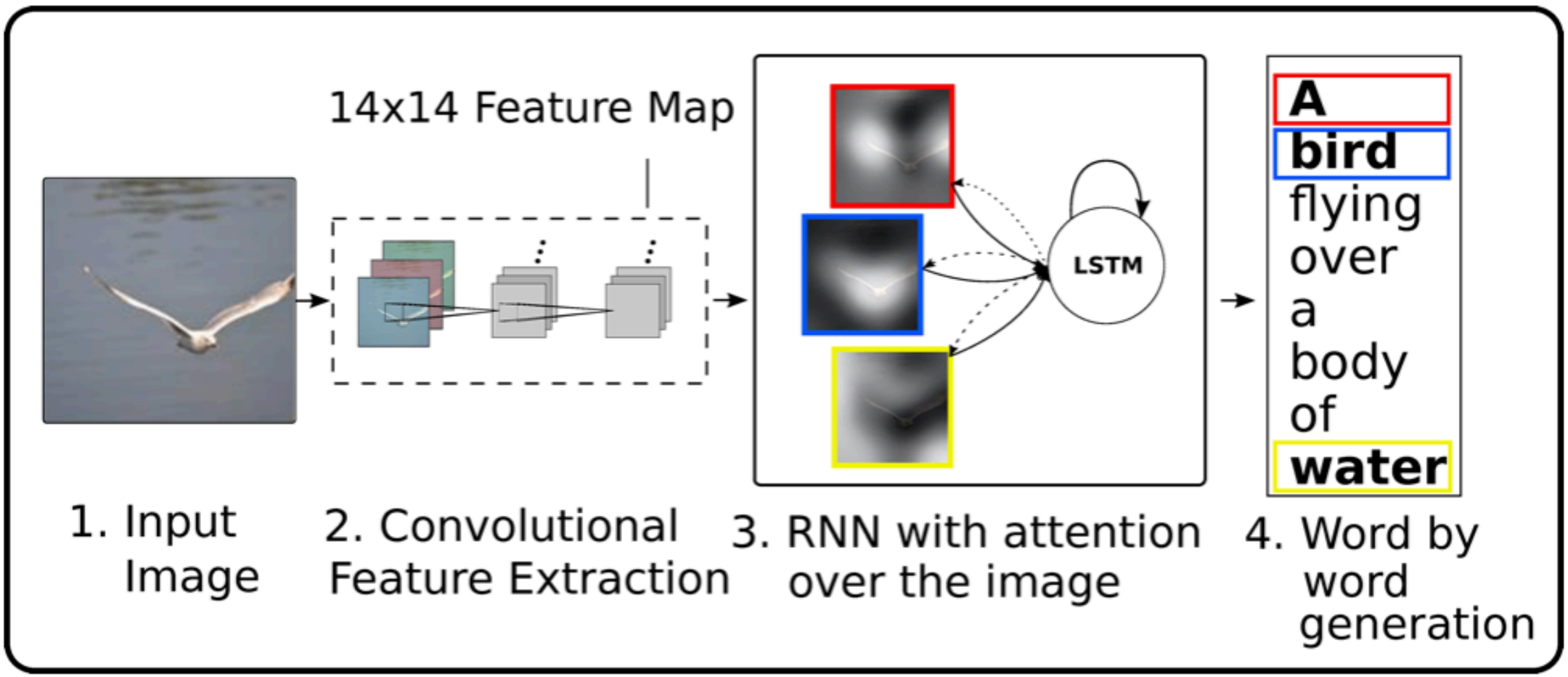
https://blog.openai.com/better-language-models/

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.)

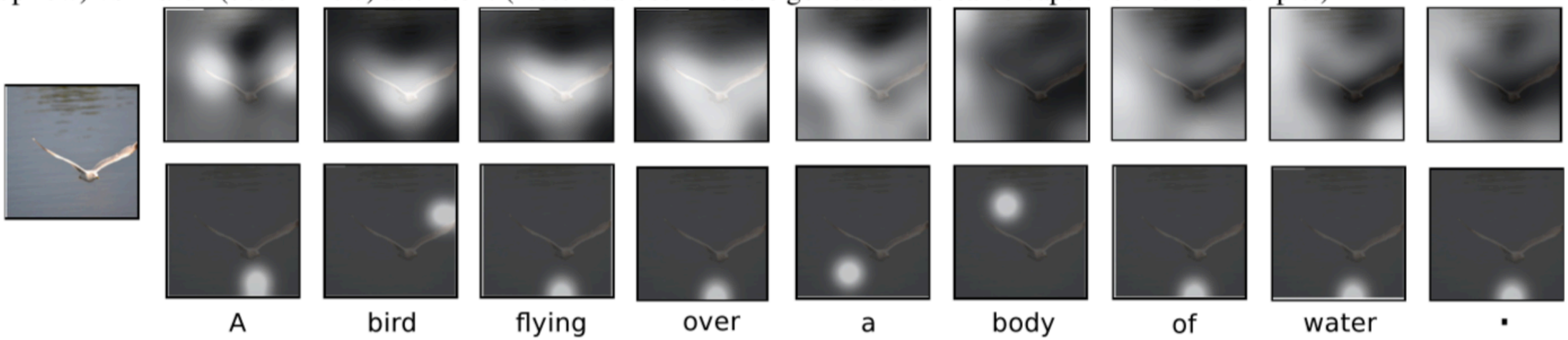
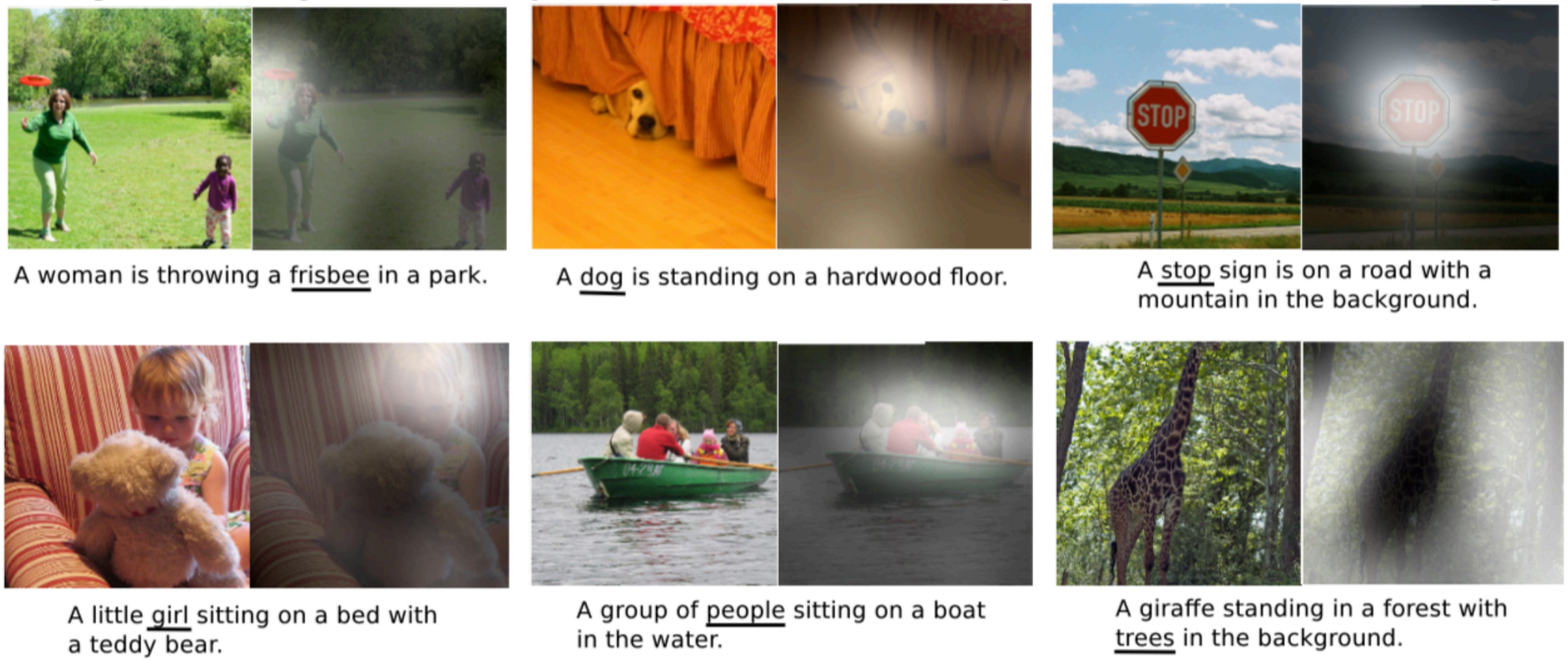


Figure 3. Examples of attending to the correct object (*white* indicates the attended regions, *underlines* indicated the corresponding word)



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