Neural Networks & Translation

Andy Way*

ADAPT Centre, Dublin City University

andy.way@adaptcentre.ie

*With much gratitude to Marcello Federico, John Kelleher, Tariq Rashid & Philipp Koehn for some (excellent!) slides

Overview

- Basic building blocks: functions & neurons
- Neural Networks
  - Feed-Forward Neural Networks
  - Inputs, Outputs, Weights, Errors
  - Word Embeddings
  - Recurrent Neural Networks
- Neural Machine Translation: Architecture
  - Encoders
  - Decoders (language models)
  - Attention
- Neural Machine Translation: Improvements
- Future Work in NMT
- Concluding Remarks

Overview

- Basic building blocks: functions & neurons
- Neural Networks
  - Feed-Forward Neural Networks
  - Inputs, Outputs, Weights, Errors
  - Word Embeddings
  - Recurrent Neural Networks
- Neural Machine Translation: Architecture
  - Encoders
  - Decoders (language models)
  - Attention
- Neural Machine Translation: Improvements
- Future Work in NMT
- Concluding Remarks

What is a function?

A function maps a set of inputs (numbers) to an output (number)

$$sum(2, 5, 4) \rightarrow 11$$

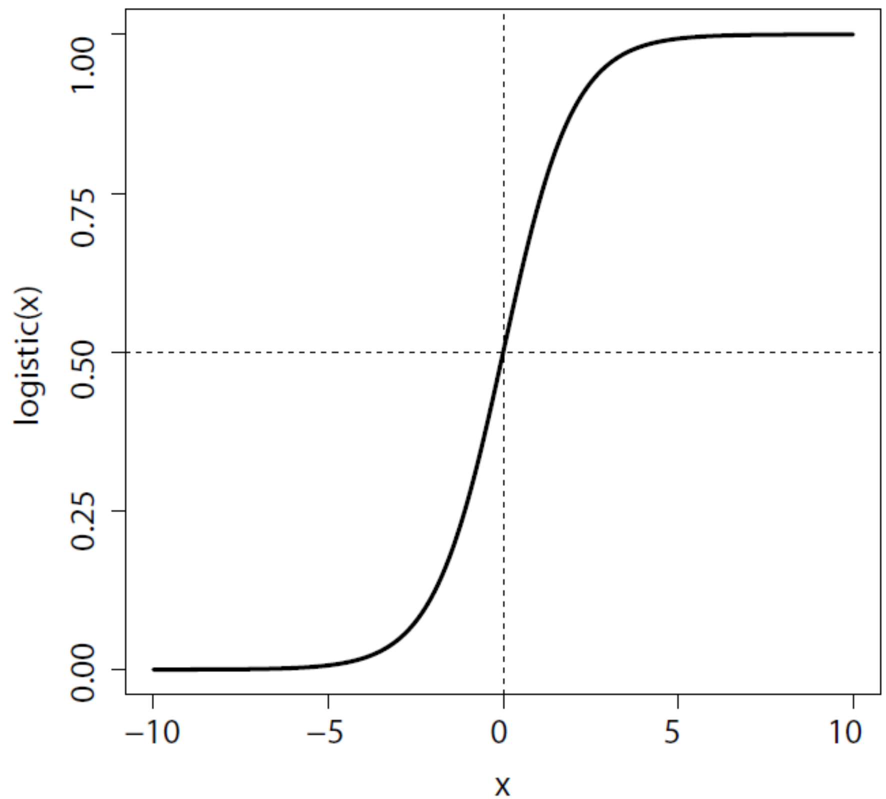
What is a $\textsc{weightedSum}$ function?

$$\textsc{weightedSum}(\underbrace{[n_1, n_2, \ldots, n_m]}_{\text{Input Numbers}}, \underbrace{[w_1, w_2, \ldots, w_m]}_{\textit{Weights}})$$
$$= (n_1 \times w_1) + (n_2 \times w_2) + \cdots + (n_m \times w_m)$$

$$\textsc{weightedSum}([3, 9], [-3, 1])$$
$$= (3 \times -3) + (9 \times 1)$$
$$= -9 + 9$$
$$= 0$$

What is an ACTIVATION function?

An ACTIVATION function takes the output of our WEIGHTEDSUM function and applies another mapping to it.



What is an $\textsc{activation}$ function?

$$\textsc{Activation} =$$

$$\textsc{logistic}(\textsc{weightedSum}((\underbrace{[n_1, n_2, \ldots, n_m]}_{\text{Input Numbers}}, \underbrace{[w_1, w_2, \ldots, w_m]}_{\textit{Weights}}))$$

$$\begin{aligned}
\textsc{logistic}(\textsc{weightedSum}([3, 9], [-3, 1])) \\
&= \textsc{logistic}((3 \times -3) + (9 \times 1)) \\
&= \textsc{logistic}(-9 + 9) \\
&= \textsc{logistic}(0) \\
&= 0.5
\end{aligned}$$

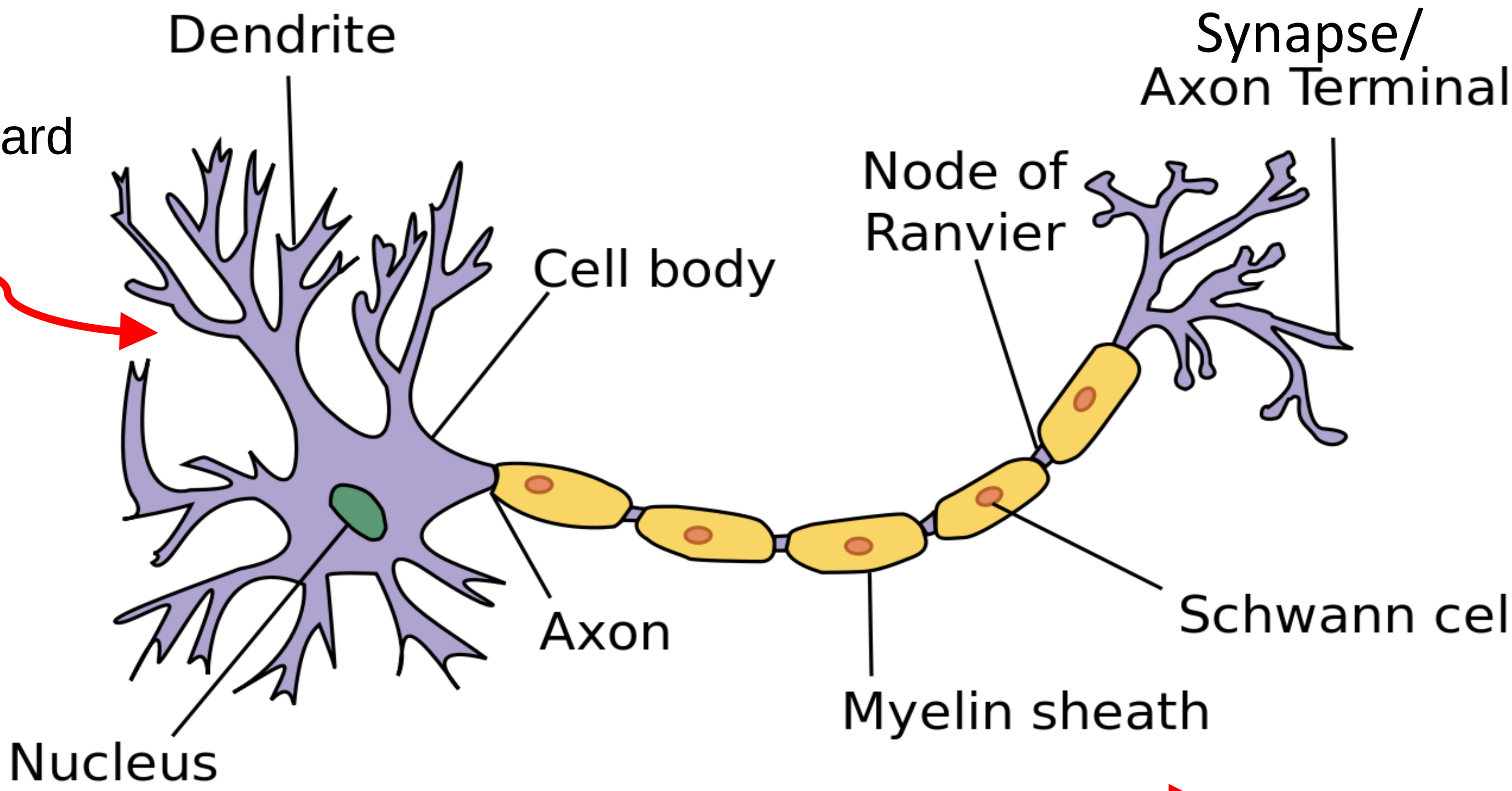
What is a NEURON?

The simple list of operations that we have just described defines the fundamental building block of a neural network: the NEURON.

$$\textsc{Neuron} = \textsc{activation}(\textsc{weightedSum}((\underbrace{[n_1, n_2, \ldots, n_m]}_{\text{Input Numbers}}, \underbrace{[w_1, w_2, \ldots, w_m]}_{\textit{Weights}}))$$

Inspired by Biological Neurons

Impulses
carried toward
cell body



Impules carried away from cell body

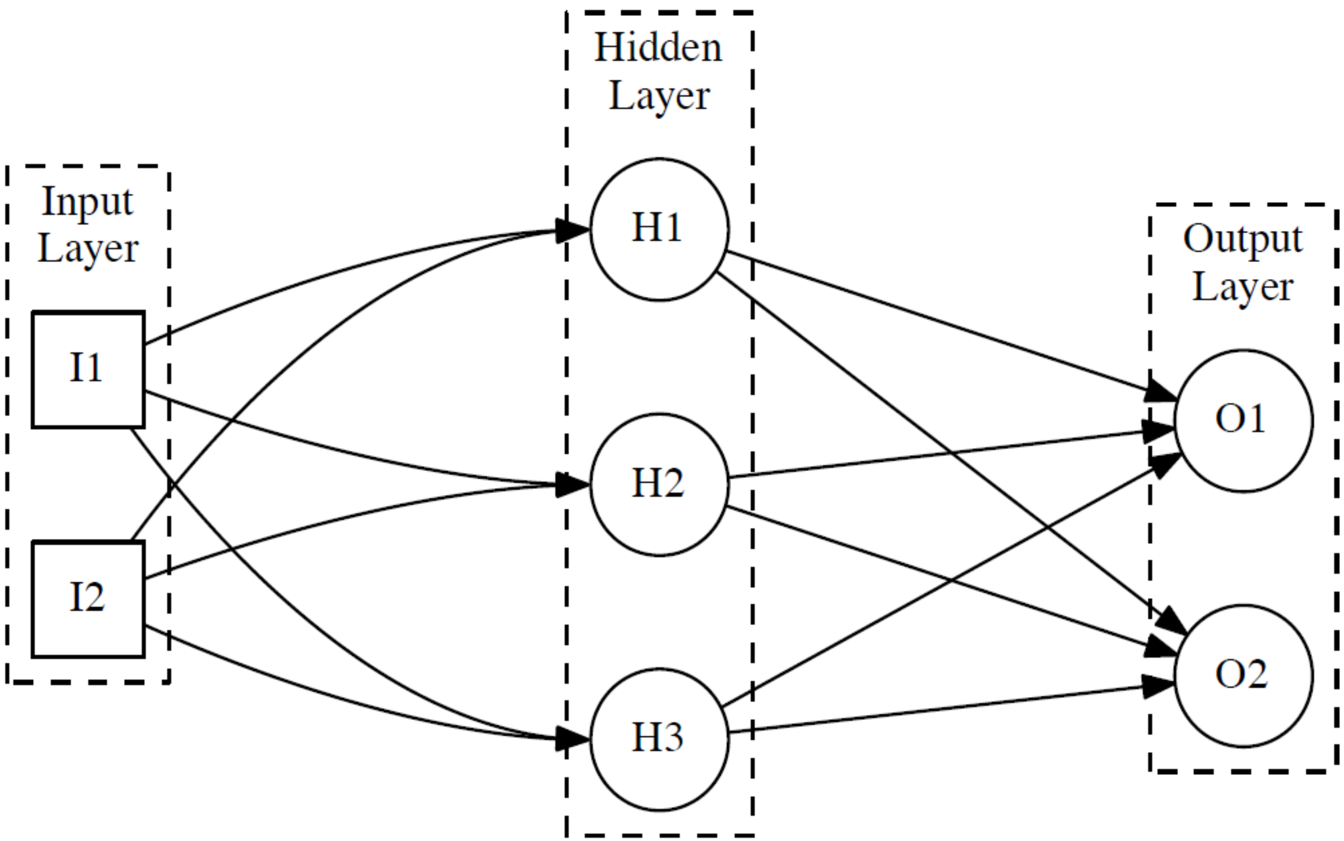
Image credit: http://cs231n.github.io/neuralnetworks-1/

Overview

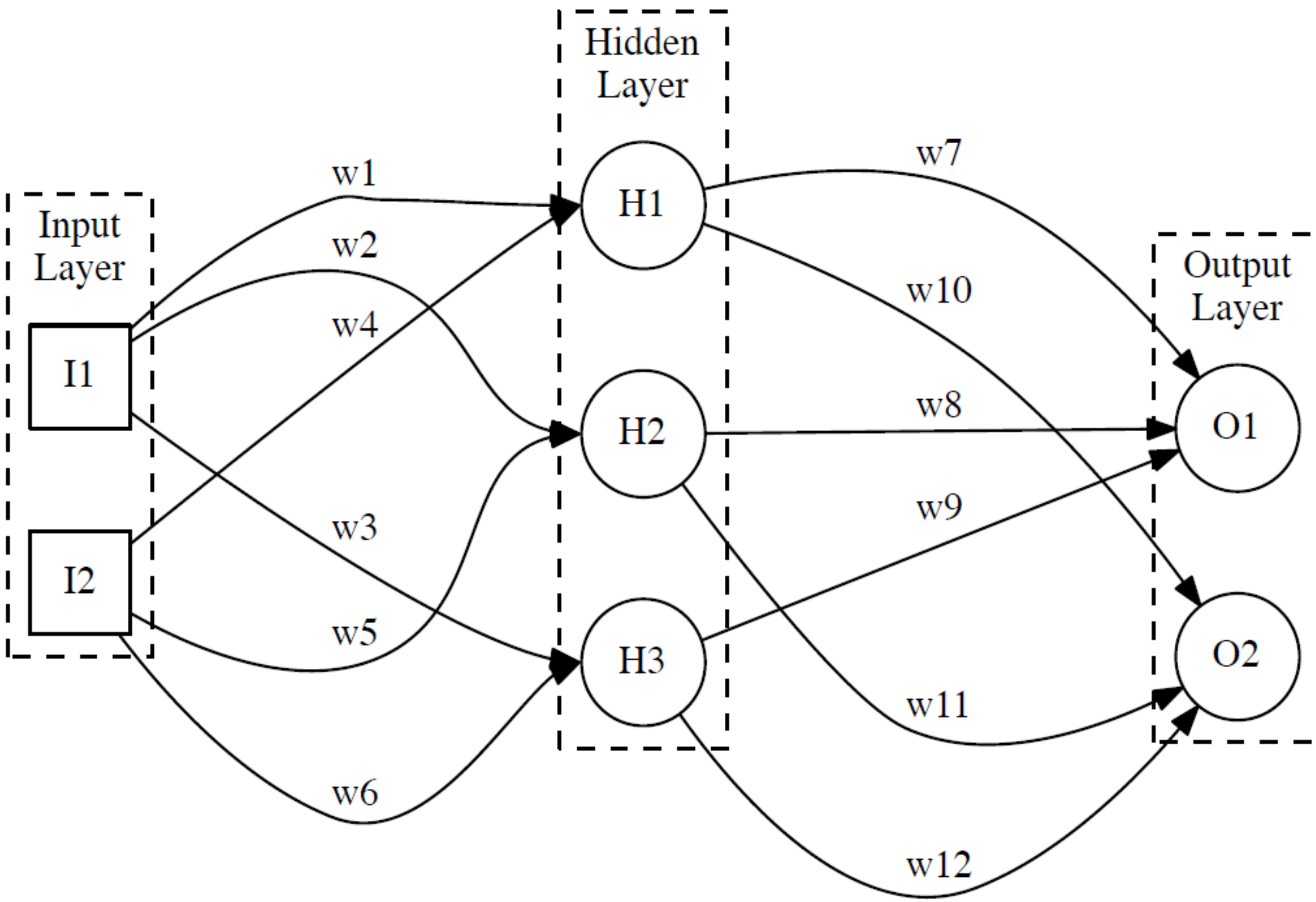
- Basic building blocks: functions & neurons
- Neural Networks
  - Feed-Forward Neural Networks
  - Inputs, Outputs, Weights, Errors
  - Word Embeddings
  - Recurrent Neural Networks
- Neural Machine Translation: Architecture
  - Encoders
  - Decoders (language models)
  - Attention
- Neural Machine Translation: Improvements
- Future Work in NMT
- Concluding Remarks

Engaging Content
Engaging People

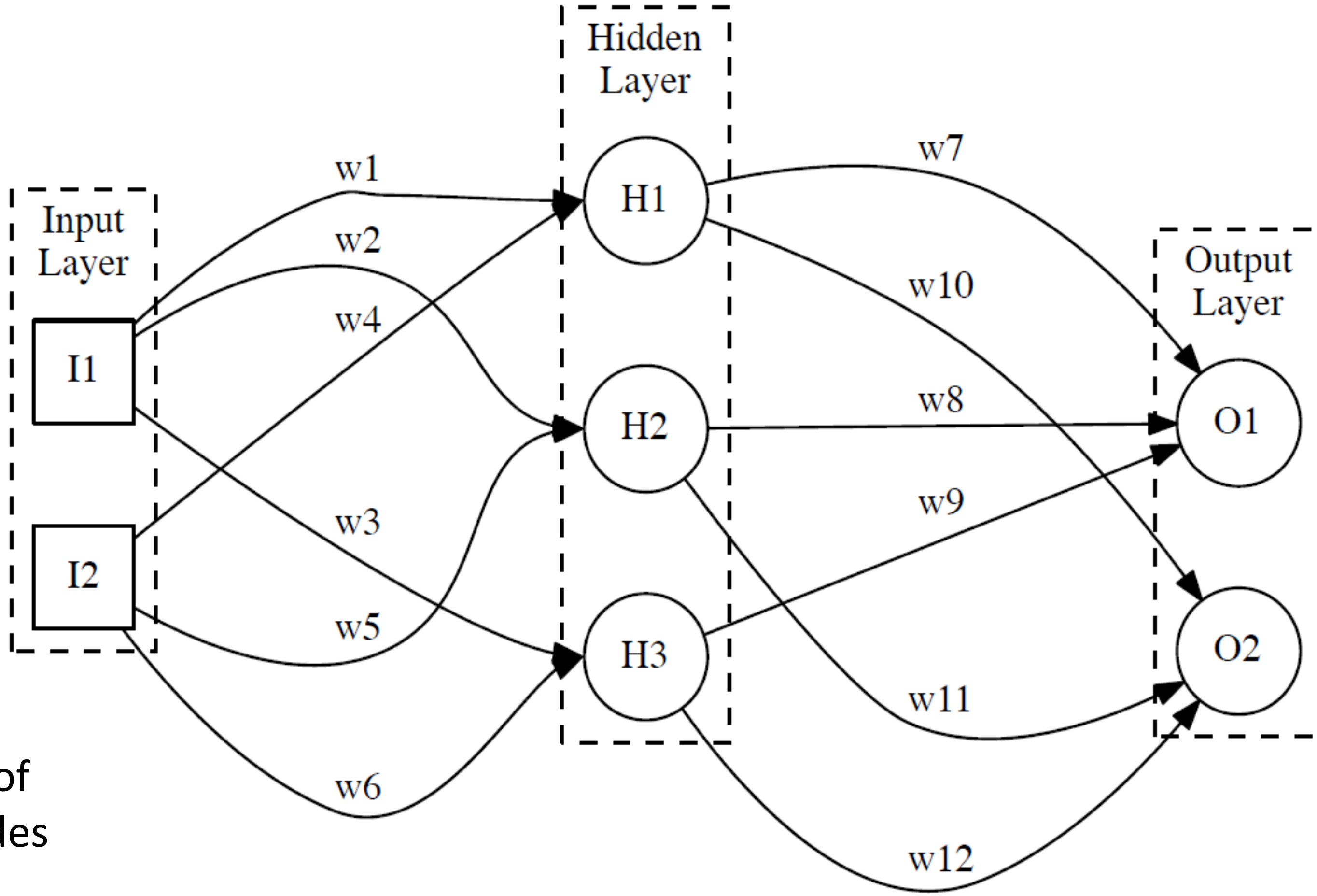
What is a Neural Network?



Where do the WEIGHTS come from?

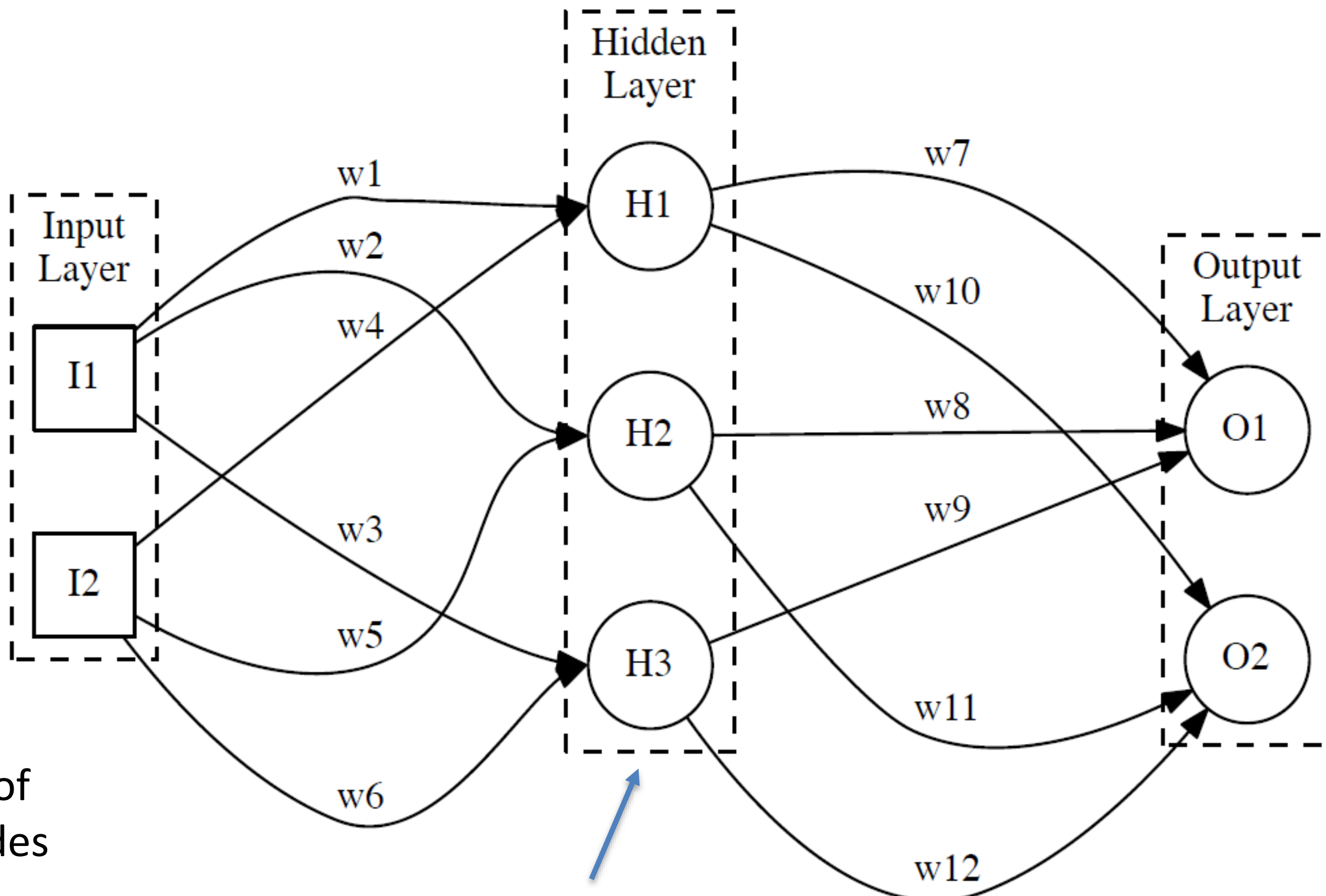


So we need:



A vector of input nodes

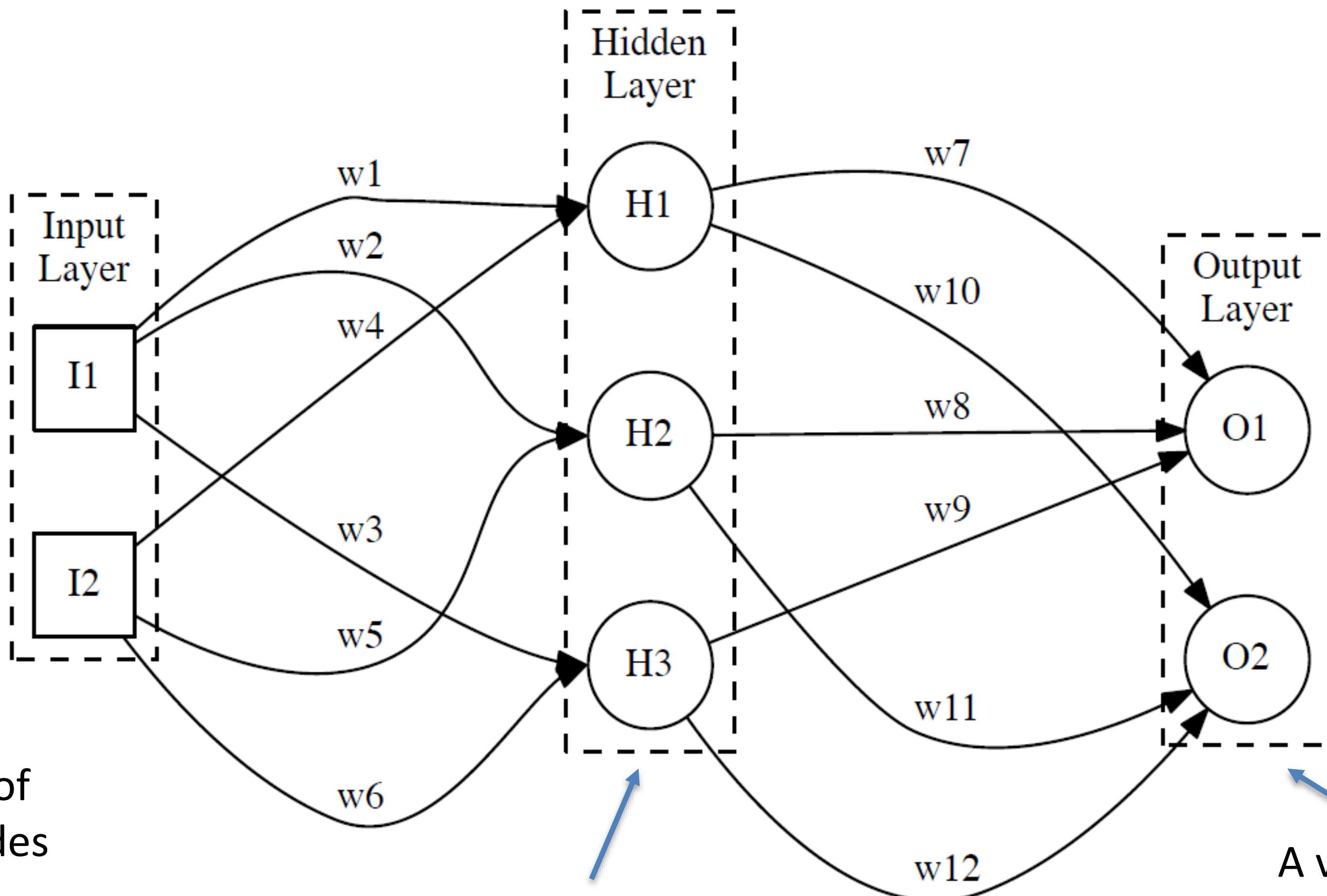
So we need:



A vector of input nodes

A vector of hidden nodes

So we need:



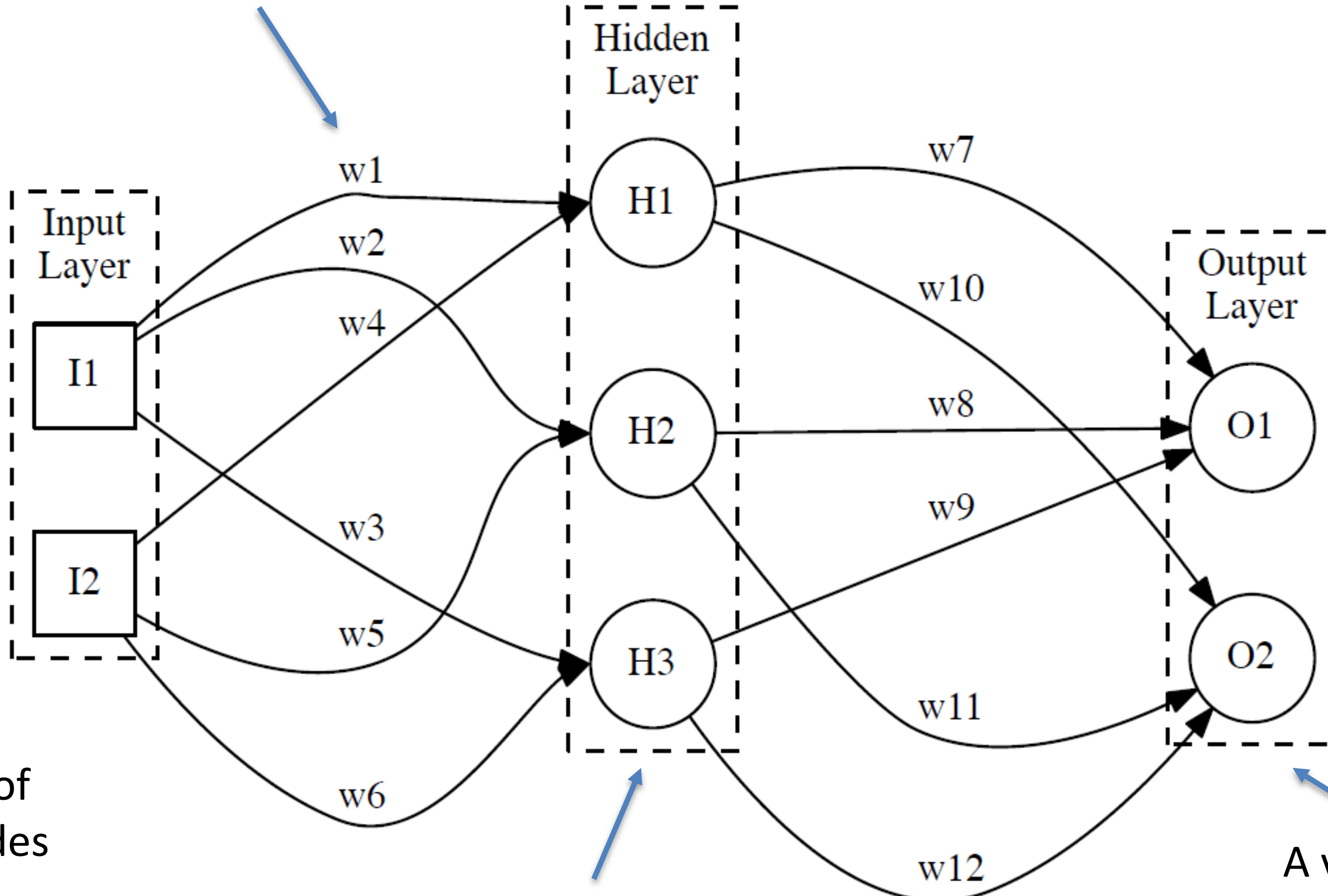
A vector of input nodes

A vector of hidden nodes

A vector of output nodes

So we need:

A matrix of weights connecting input nodes and hidden nodes



A vector of input nodes

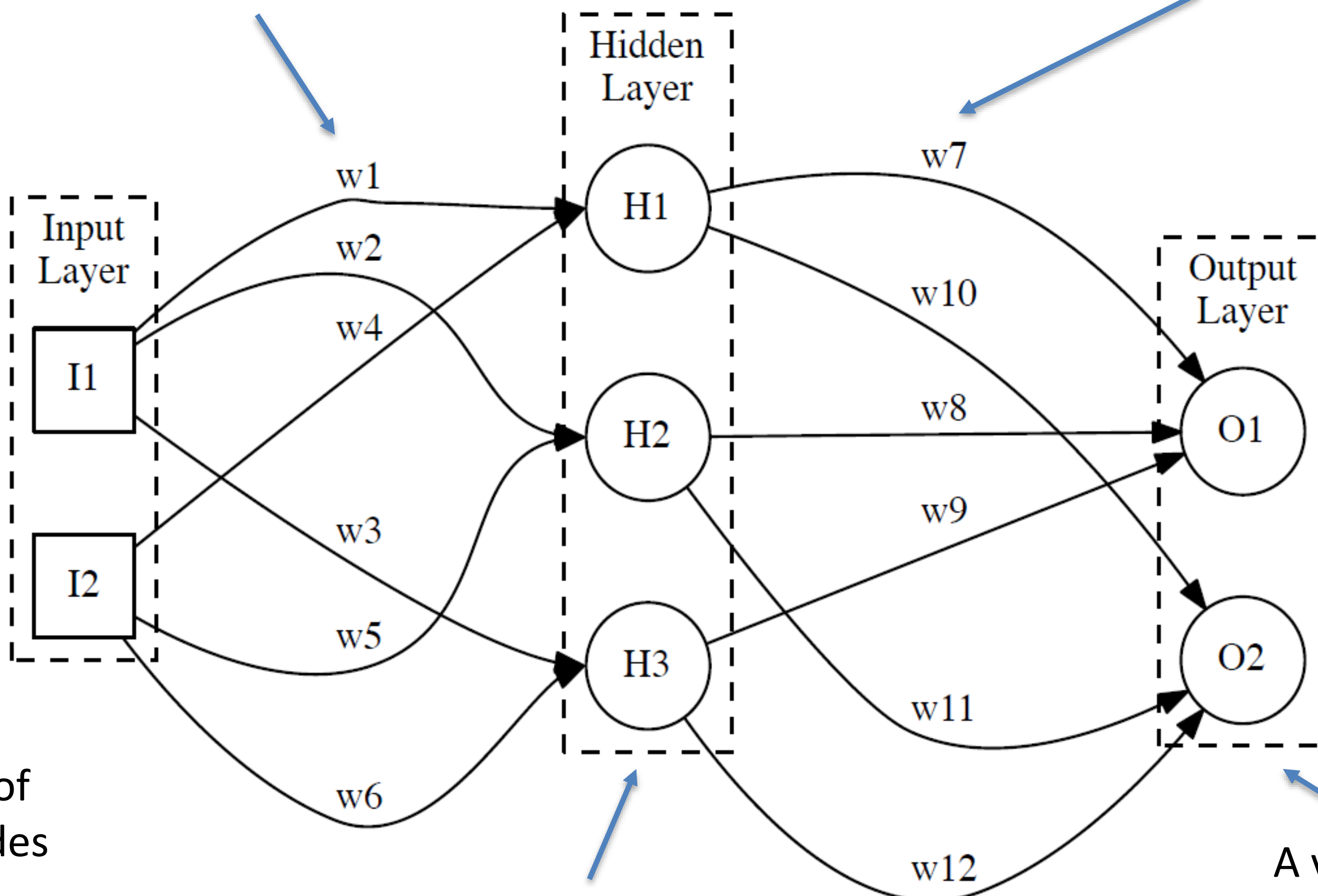
A vector of hidden nodes

A vector of output nodes

So we need:

A matrix of weights connecting input nodes and hidden nodes

A matrix of weights connecting hidden nodes and output nodes



A vector of input nodes

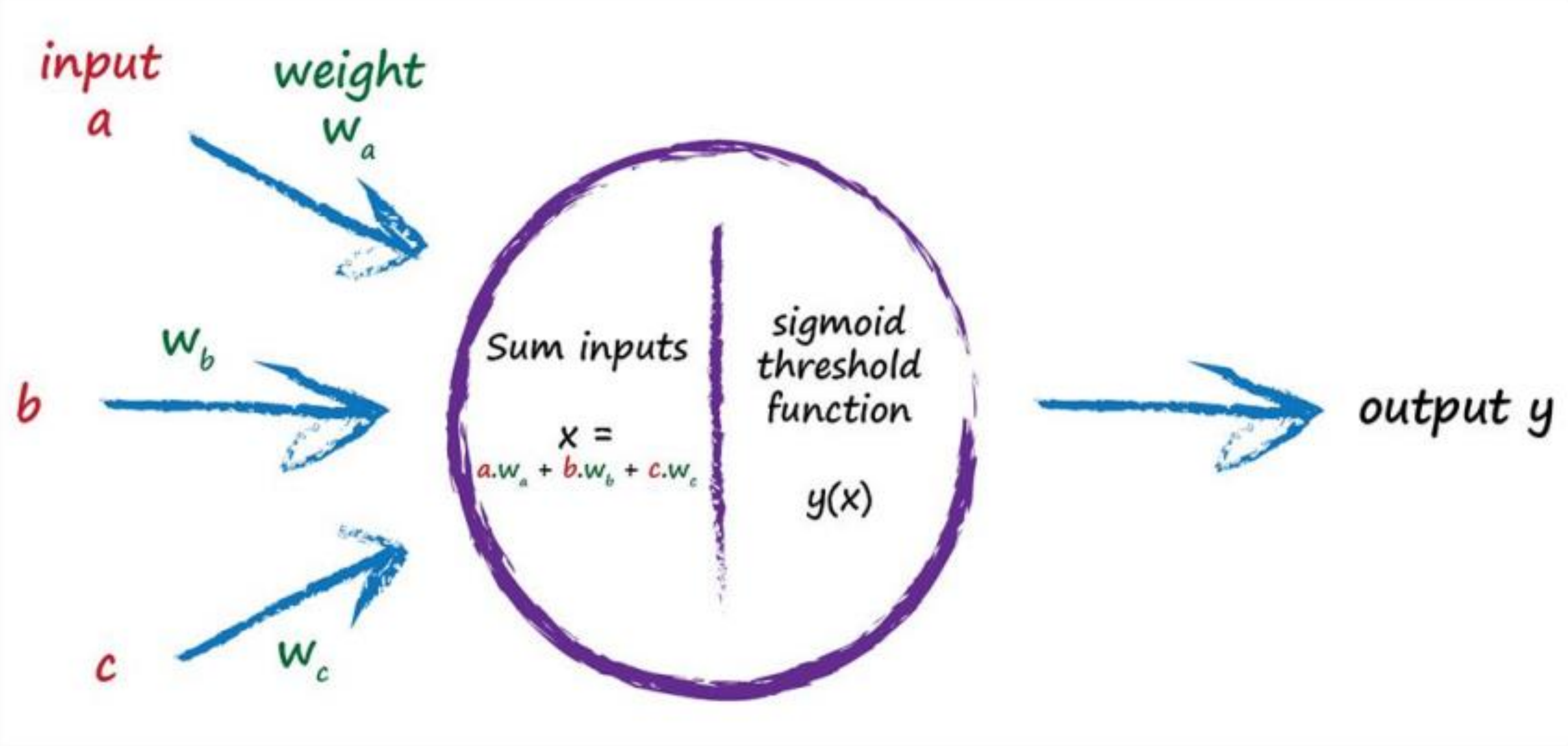
A vector of hidden nodes

A vector of output nodes

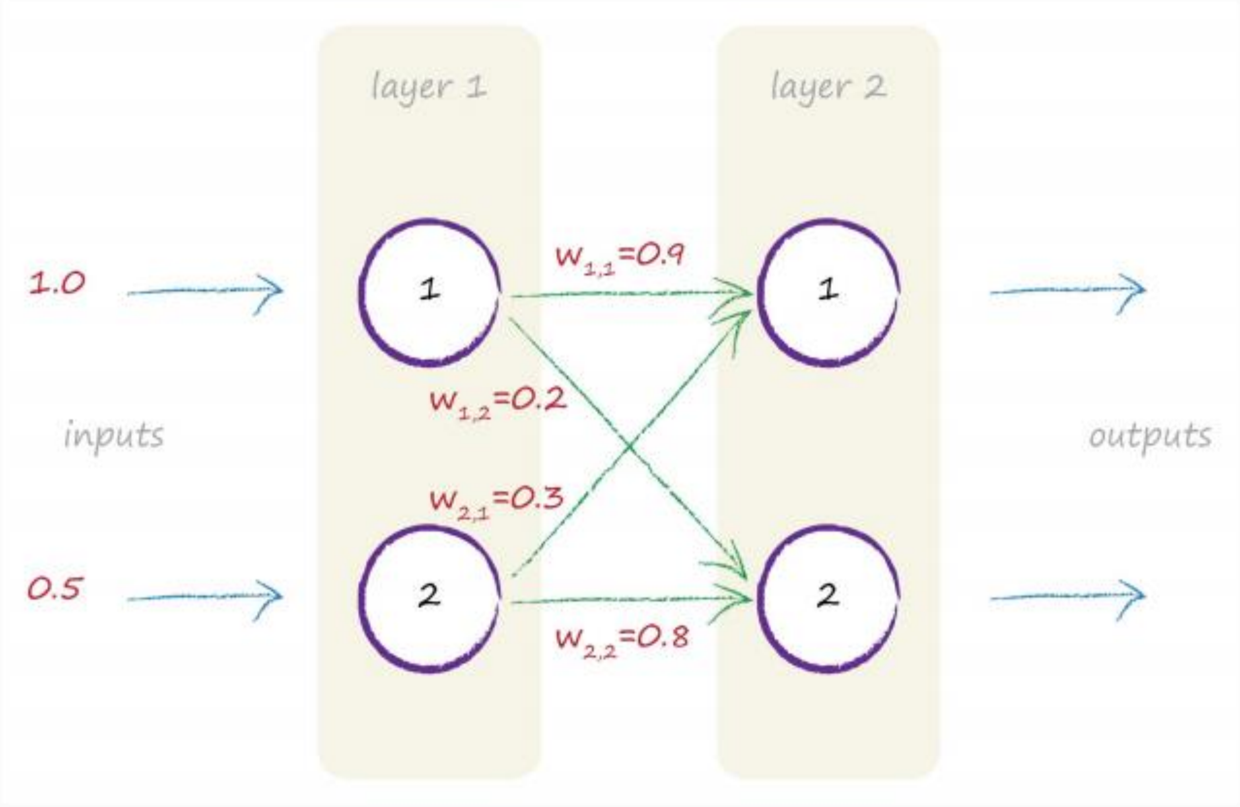
Training a NEURAL NETWORK

- We train a neural network by iteratively updating the weights
- We start by randomly assigning weights to each edge
- We then show the network examples of inputs and expected outputs and update the weights using BACKPROPAGATION so that the network outputs match the expected outputs
- We keep updating the weights until the network is working the way we want

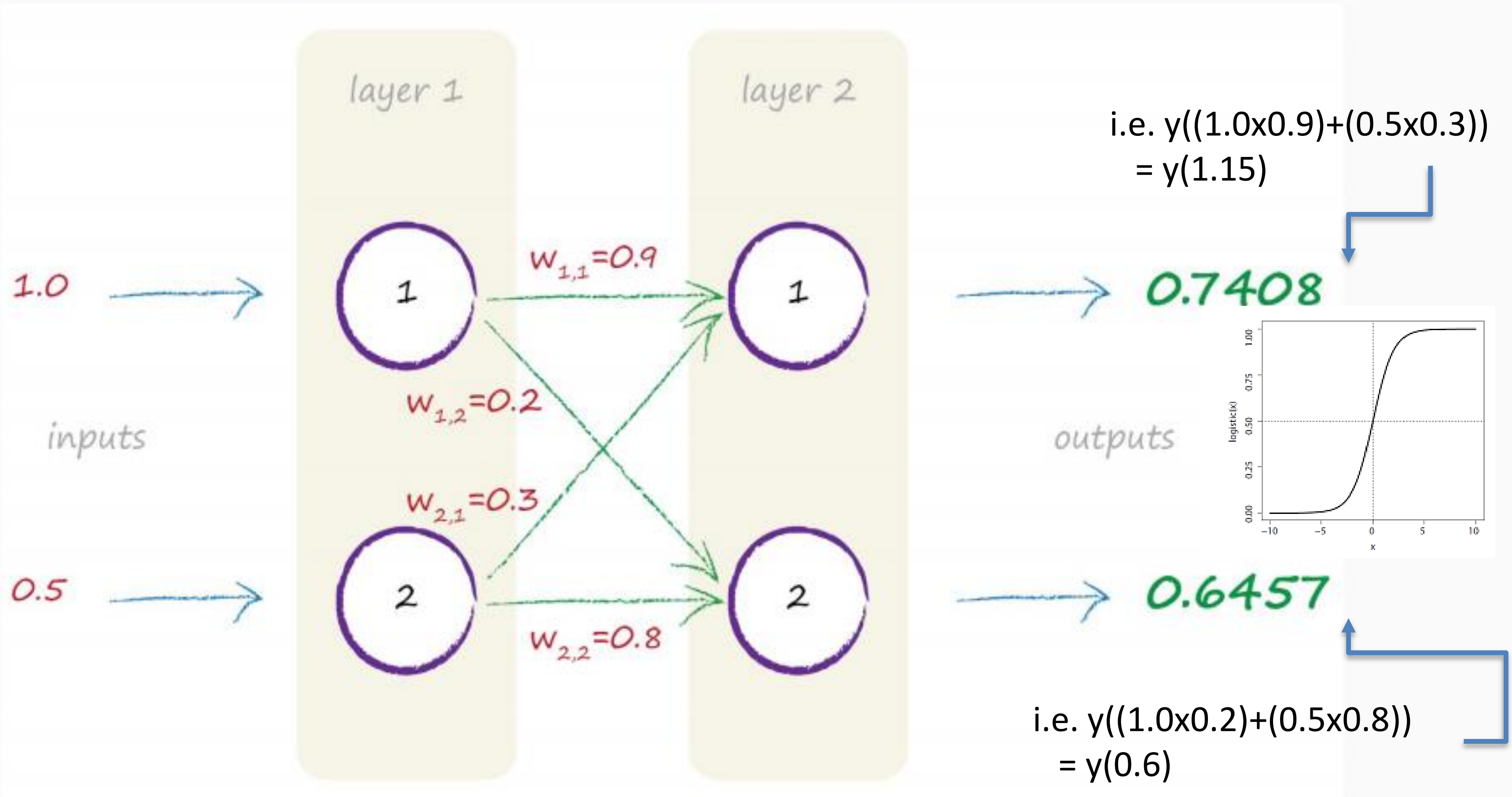
Feeding Signals Forward



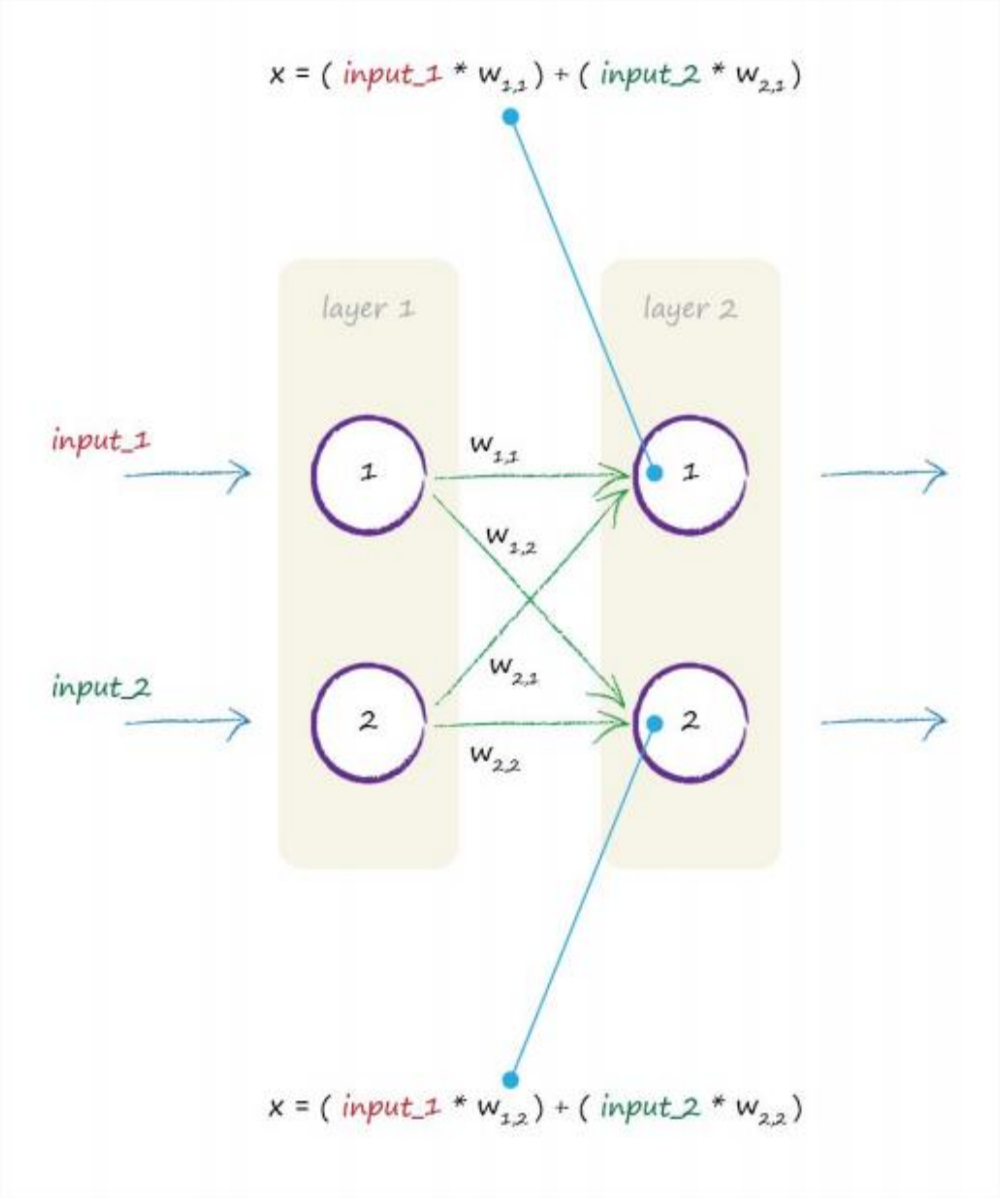
Feeding Signals Forward



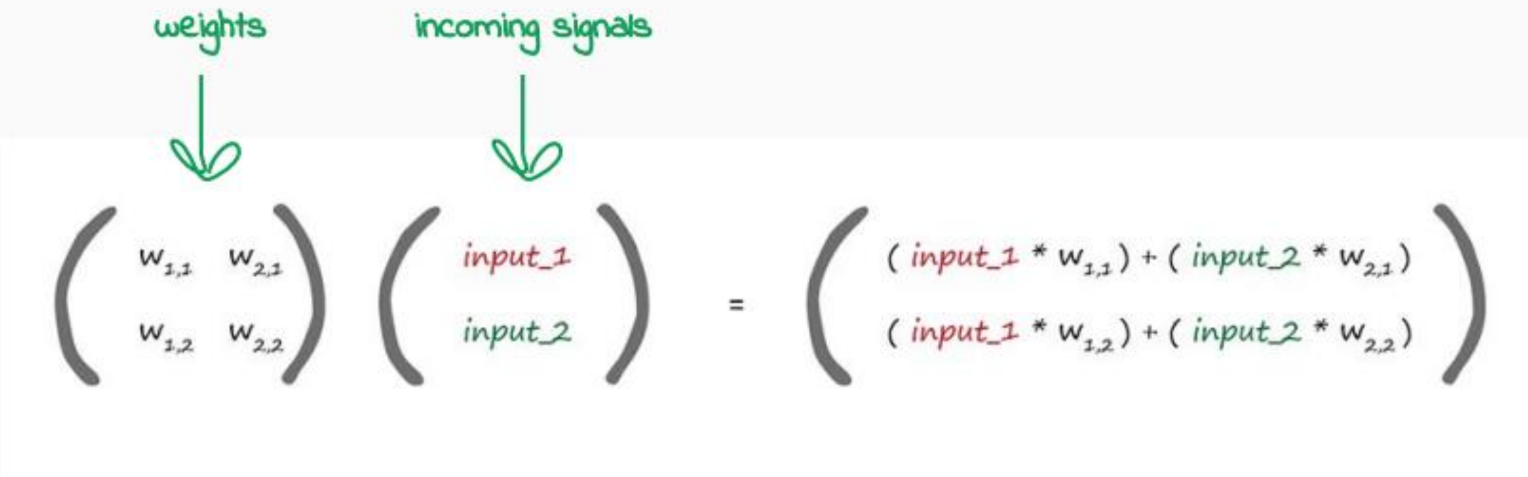
Feeding Signals Forward



Matrix Multiplication

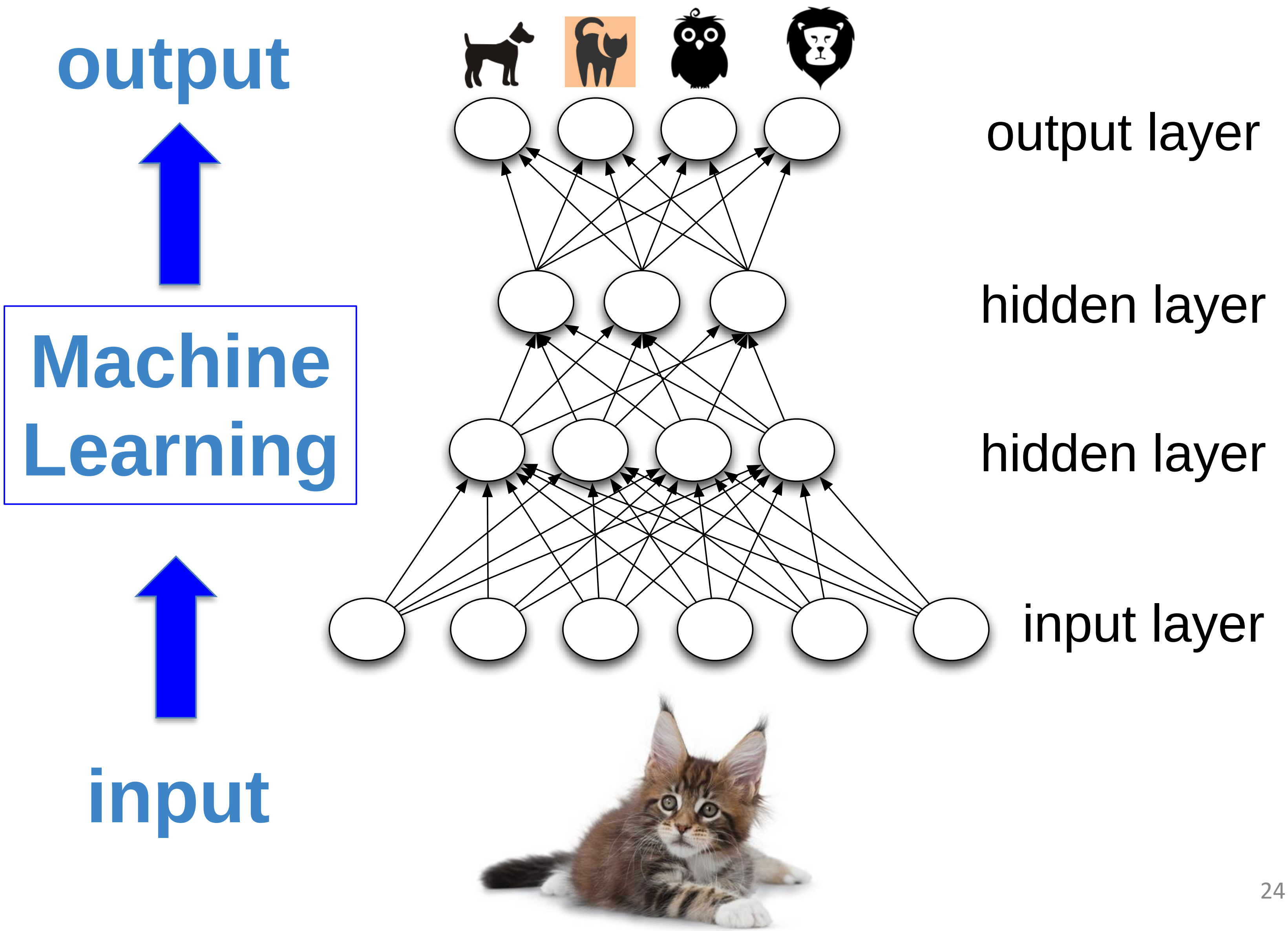


Matrix Multiplication

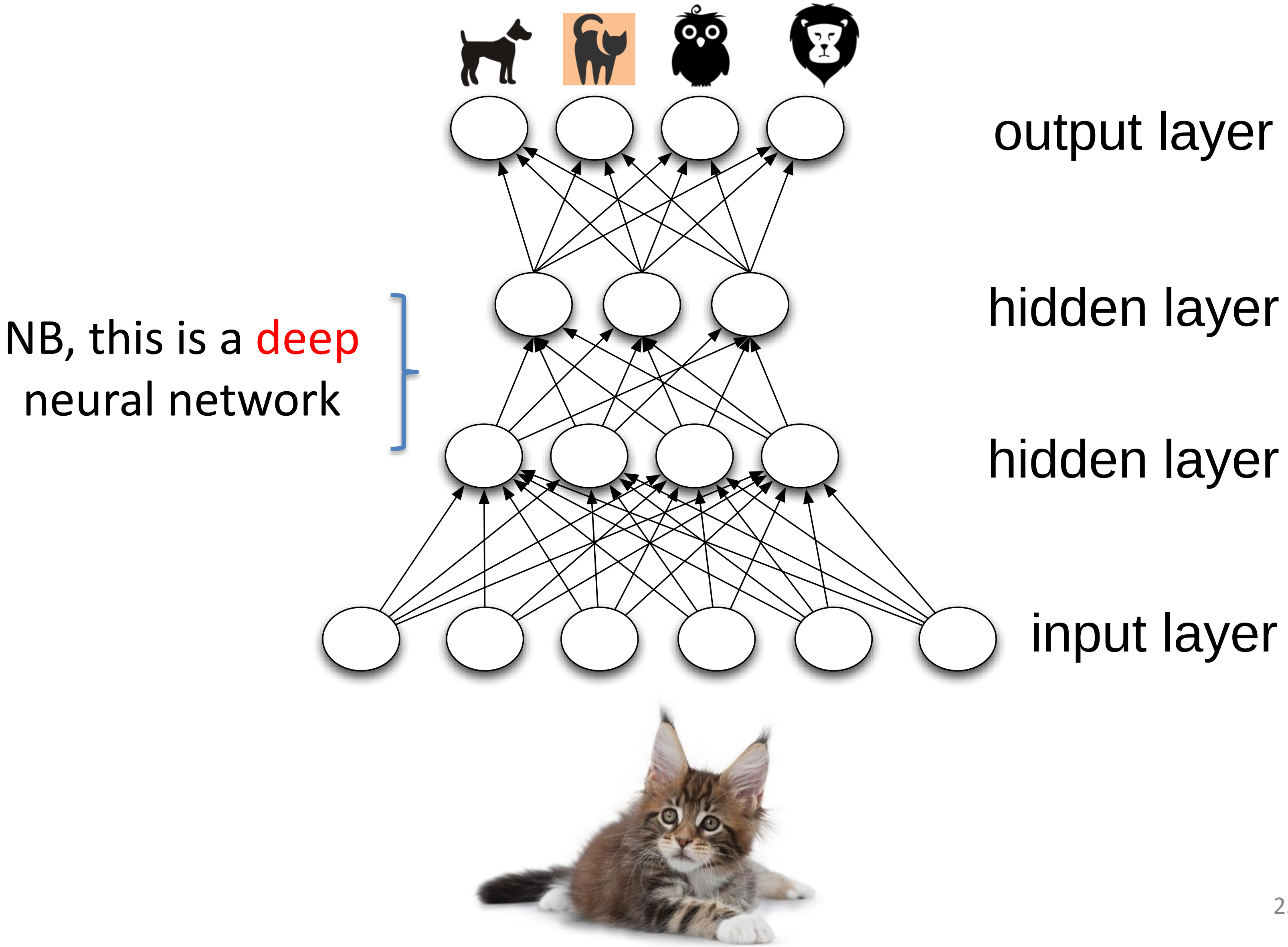


- The millions of feed-forward calculations can be expressed concisely as matrix multiplication, no matter what the configuration of the network is …
- Today’s programming languages (and tools like MatLab) can do matrix multiplication efficiently and quickly
- Beware of ‘too large’ matrices!

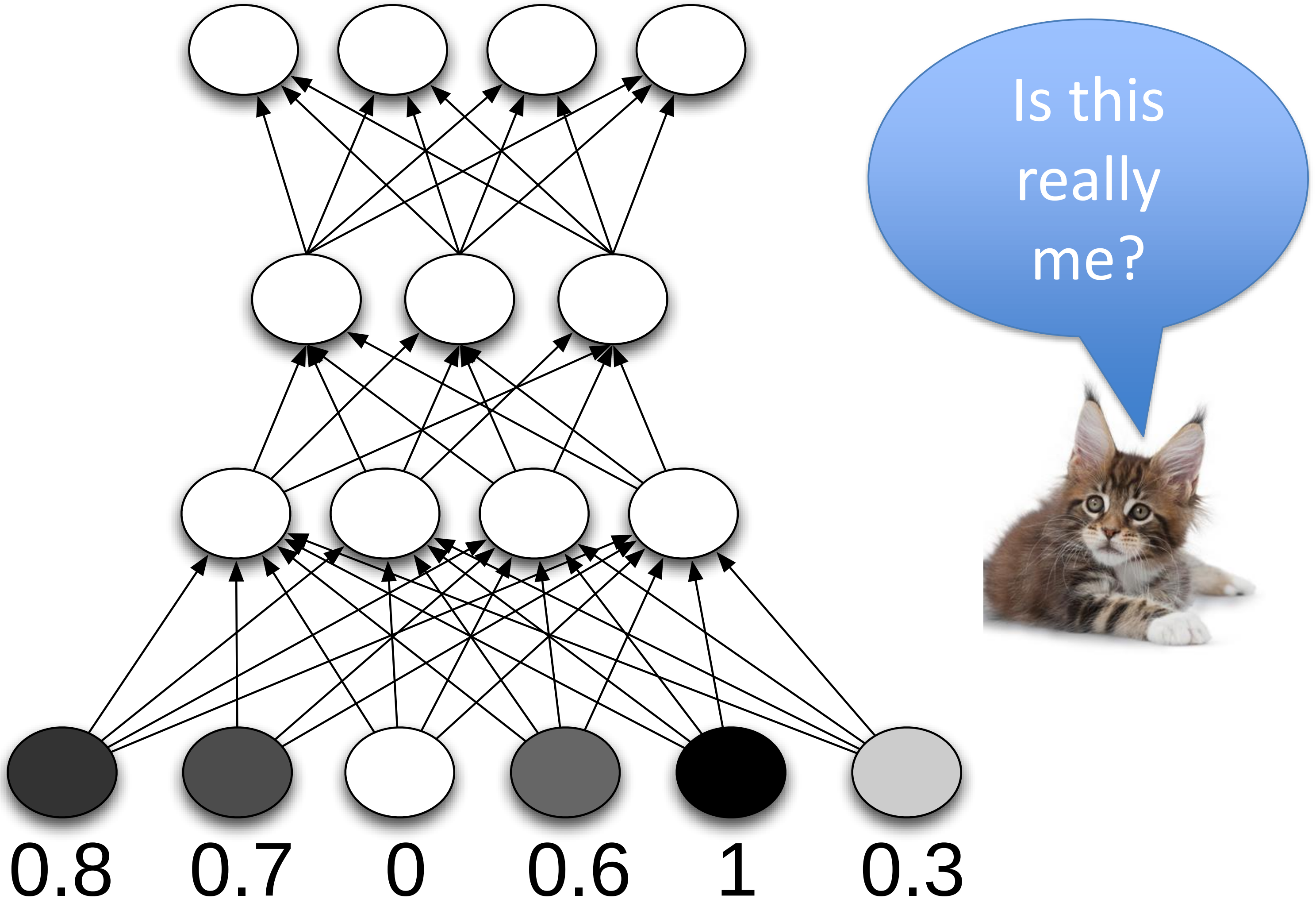
Feed-forward neural networks



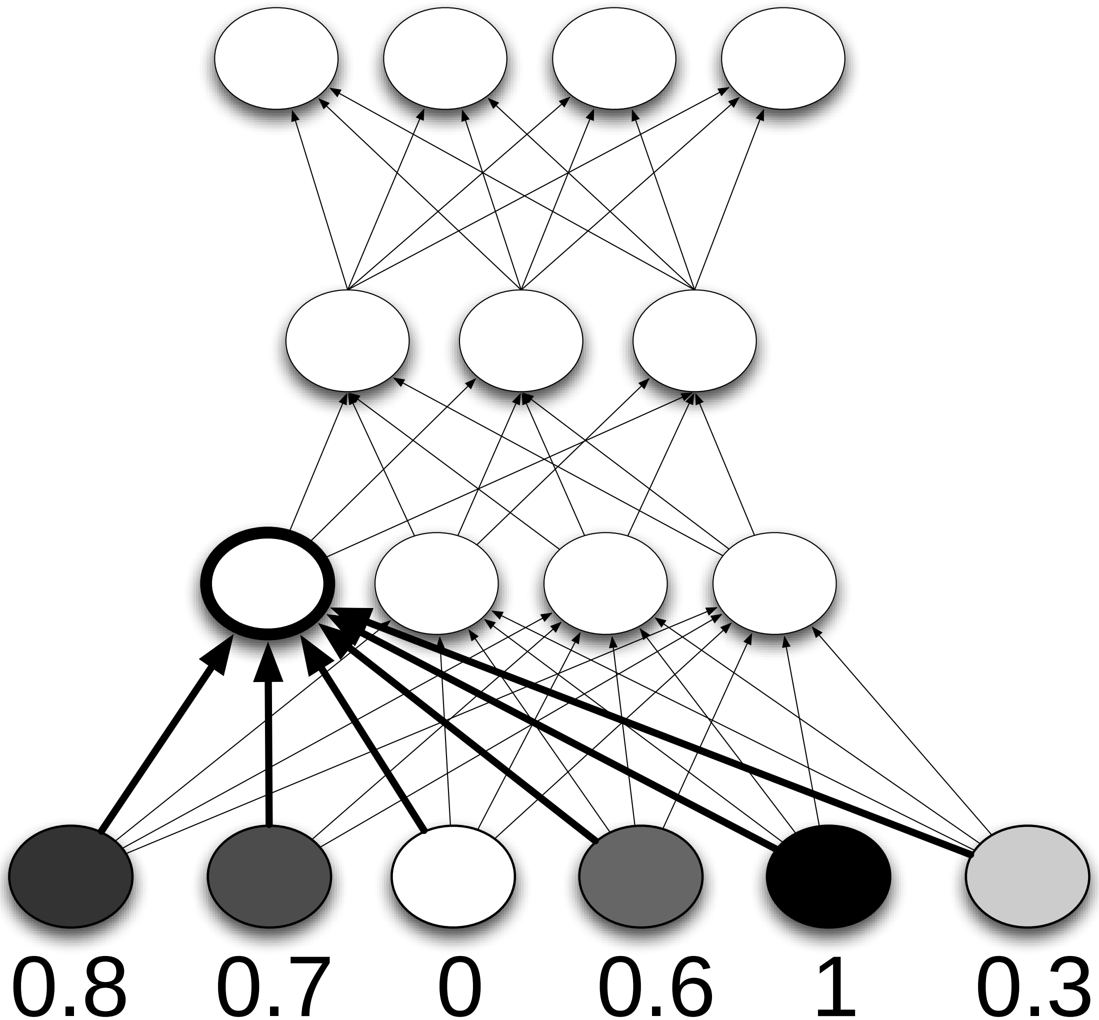
Feed-forward neural networks



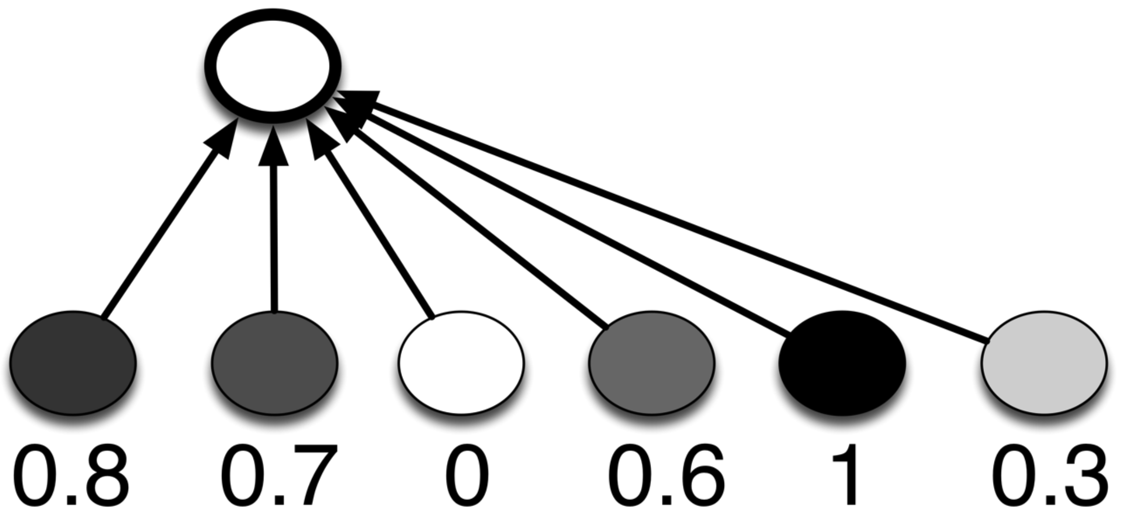
Feed-forward neural networks



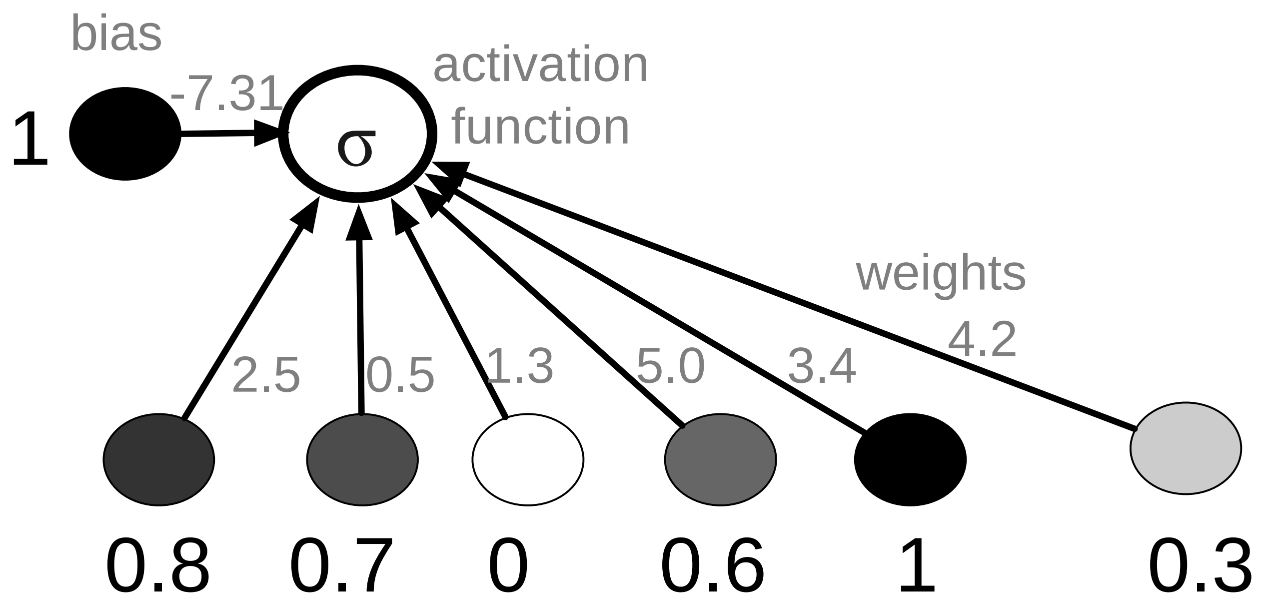
Feed-forward neural networks



Feed-forward neural networks

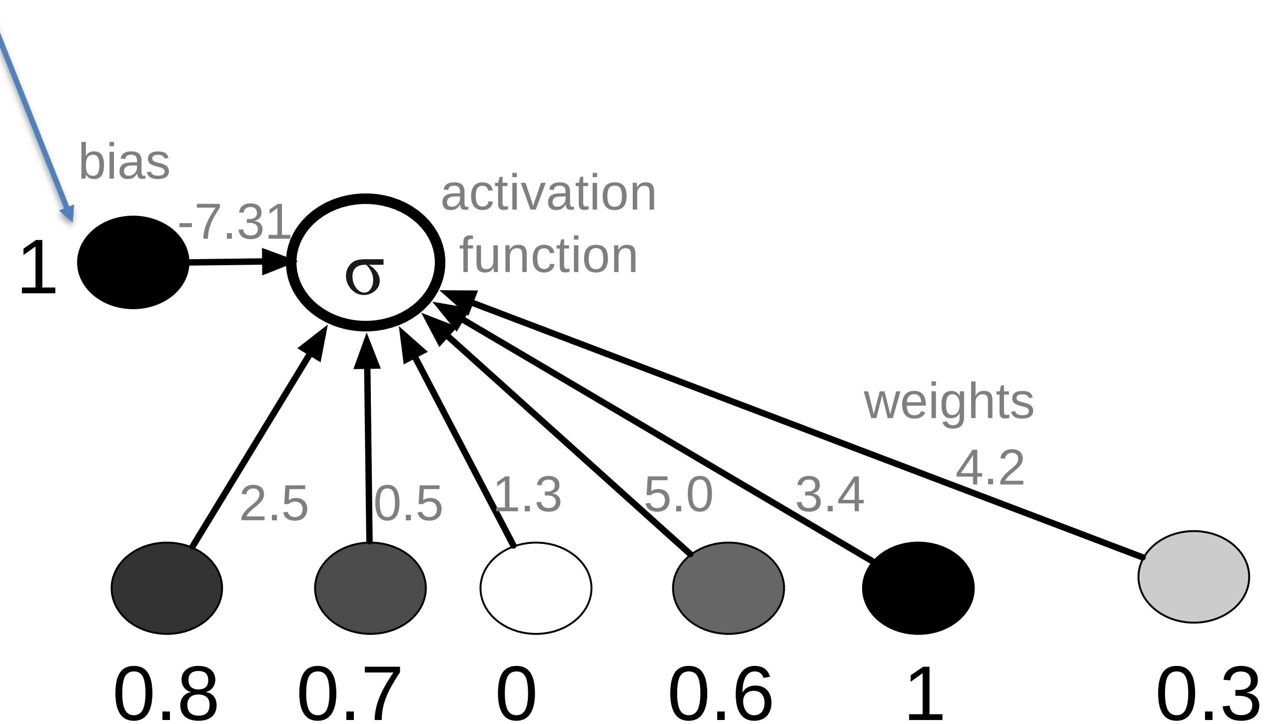


Feed-forward neural networks



Feed-forward neural networks

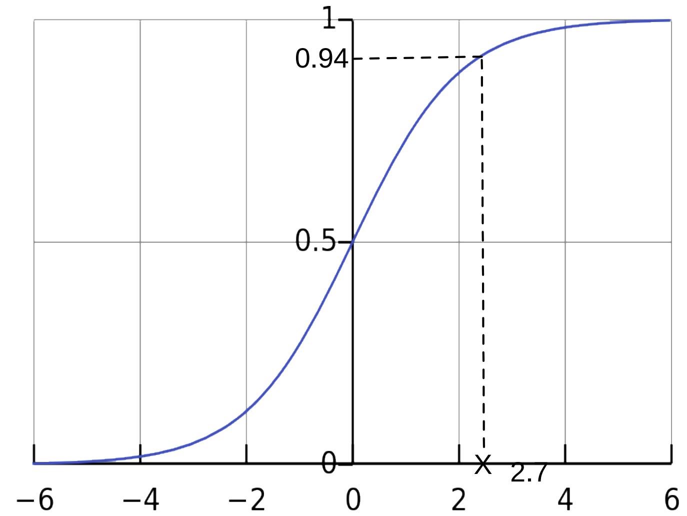
NB, something new, bias nodes.
Always have value 1; give the network something to work with in case all input values are 0 – avoids weighted sum being equal to 0

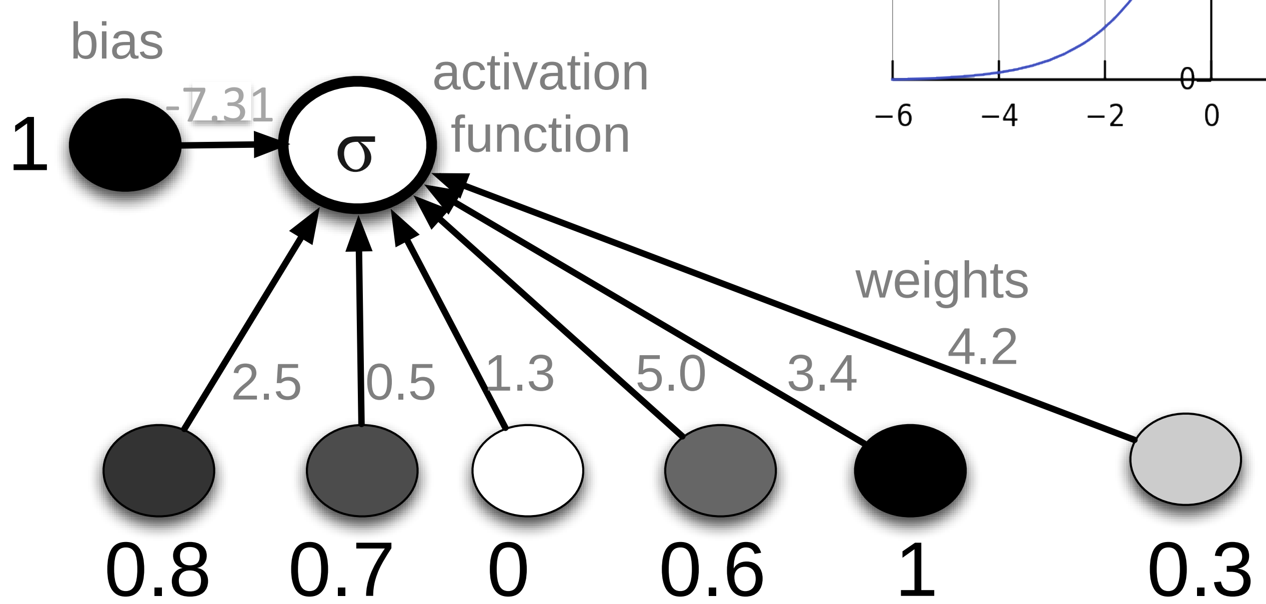


Feed-forward neural networks

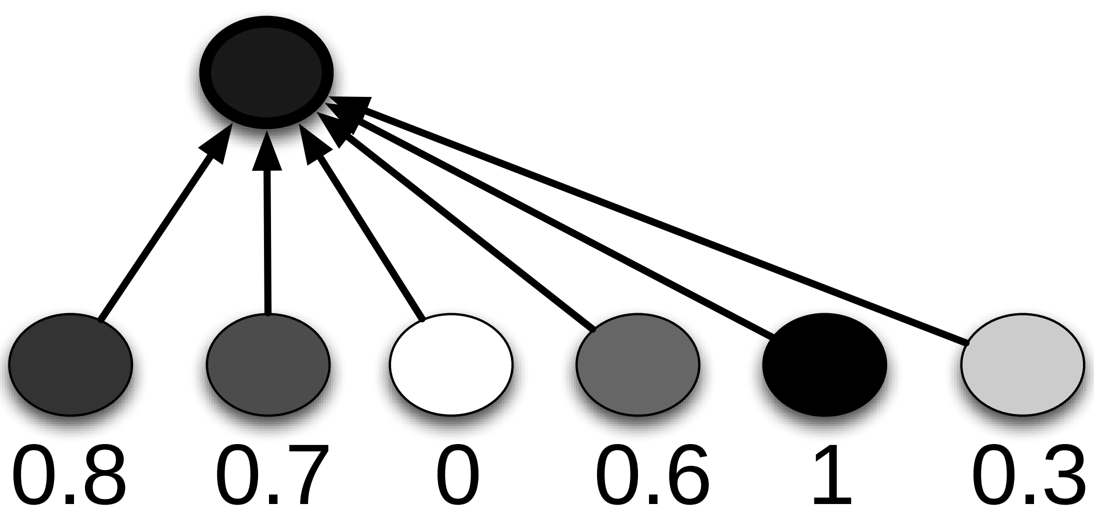
σ (input x weights)=
σ(2.7)= 0.94

sigmoid function

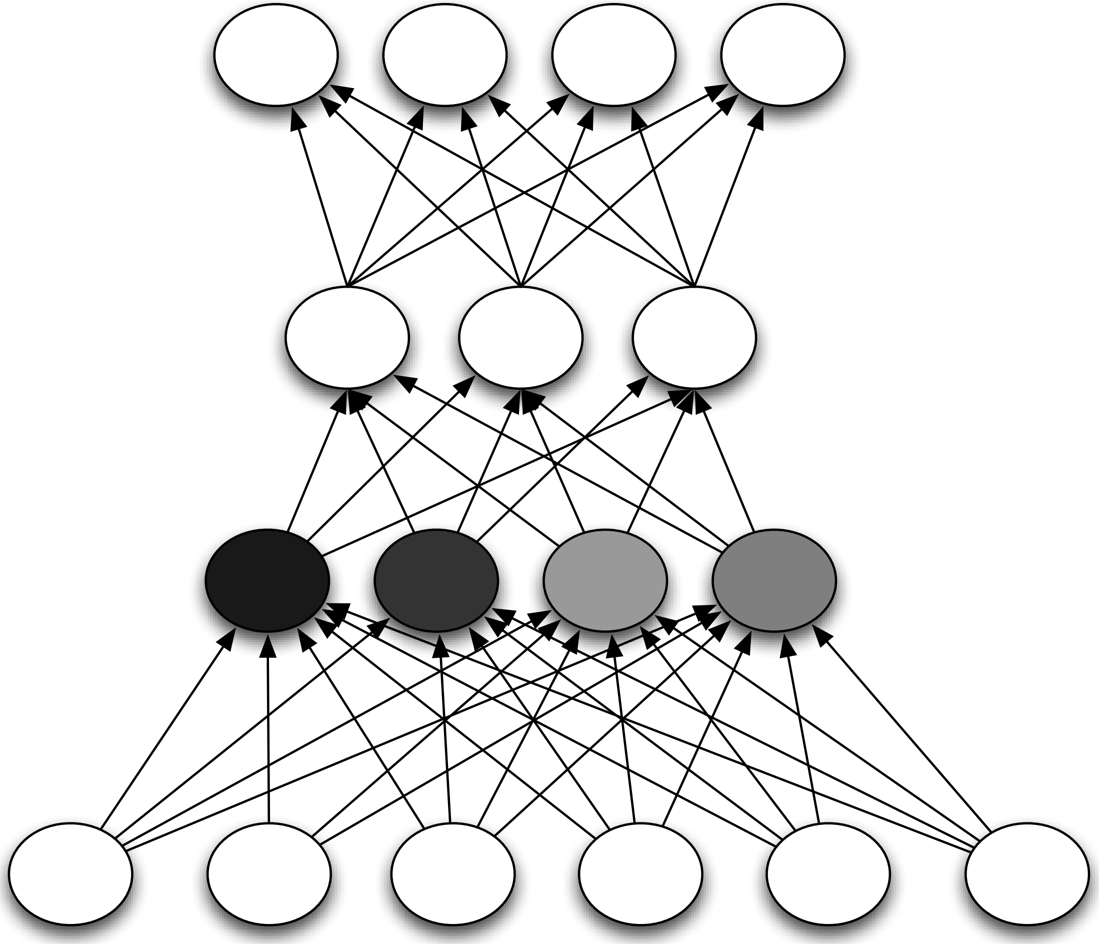




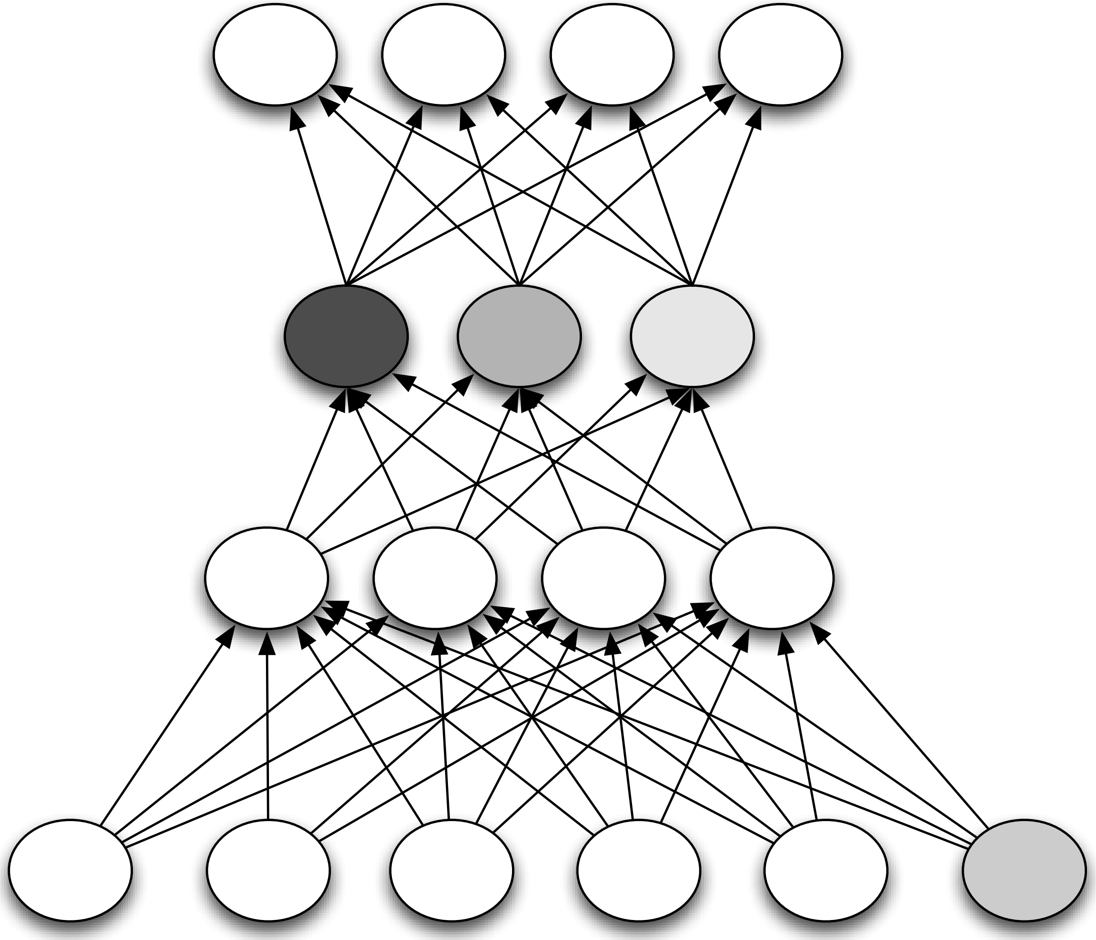
Feed-forward neural networks



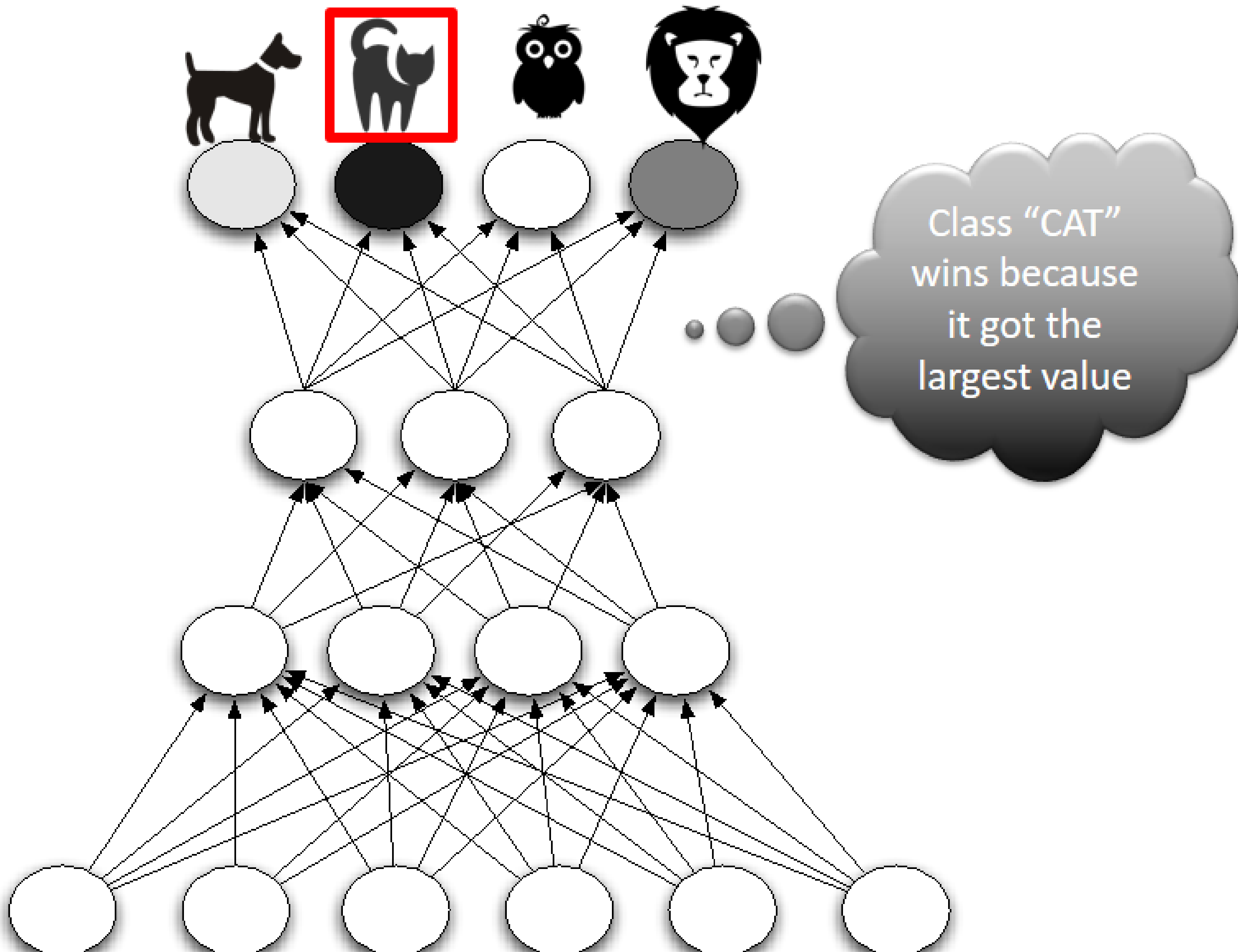
Feed-forward neural networks



Feed-forward neural networks



Feed-forward neural networks



FF-NNs: multilayer perceptron

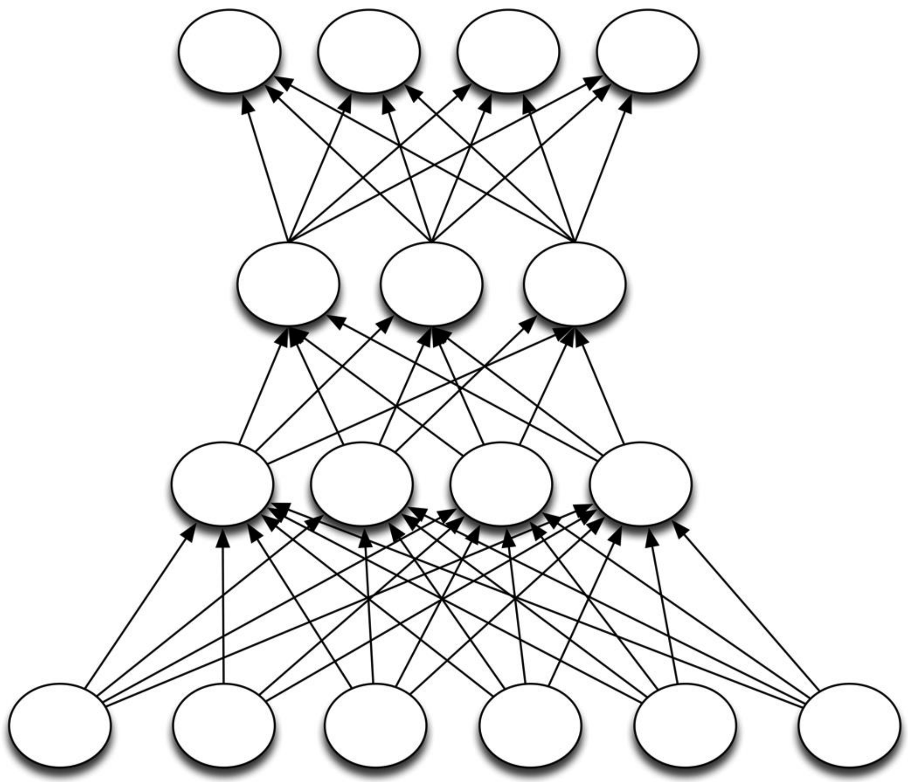
$y_{4\times 1}$

$W^{3}_{4\times 3}$

$W^{2}_{3\times 4}$ $b^{2}_{3\times 1}$

$W^{1}_{4\times 6}$ $b^{1}_{4\times 1}$

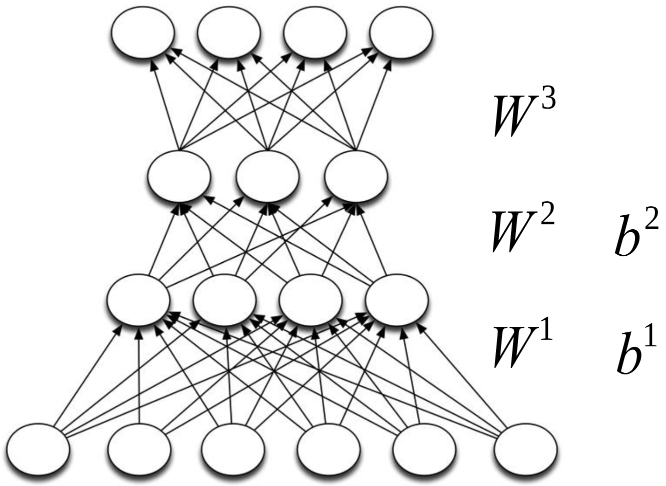
$x_{6\times 1}$



Weight matrices and bias vectors are the network parameters (bias nodes in the hidden layers are not shown in the picture)

How do NNs learn a task?

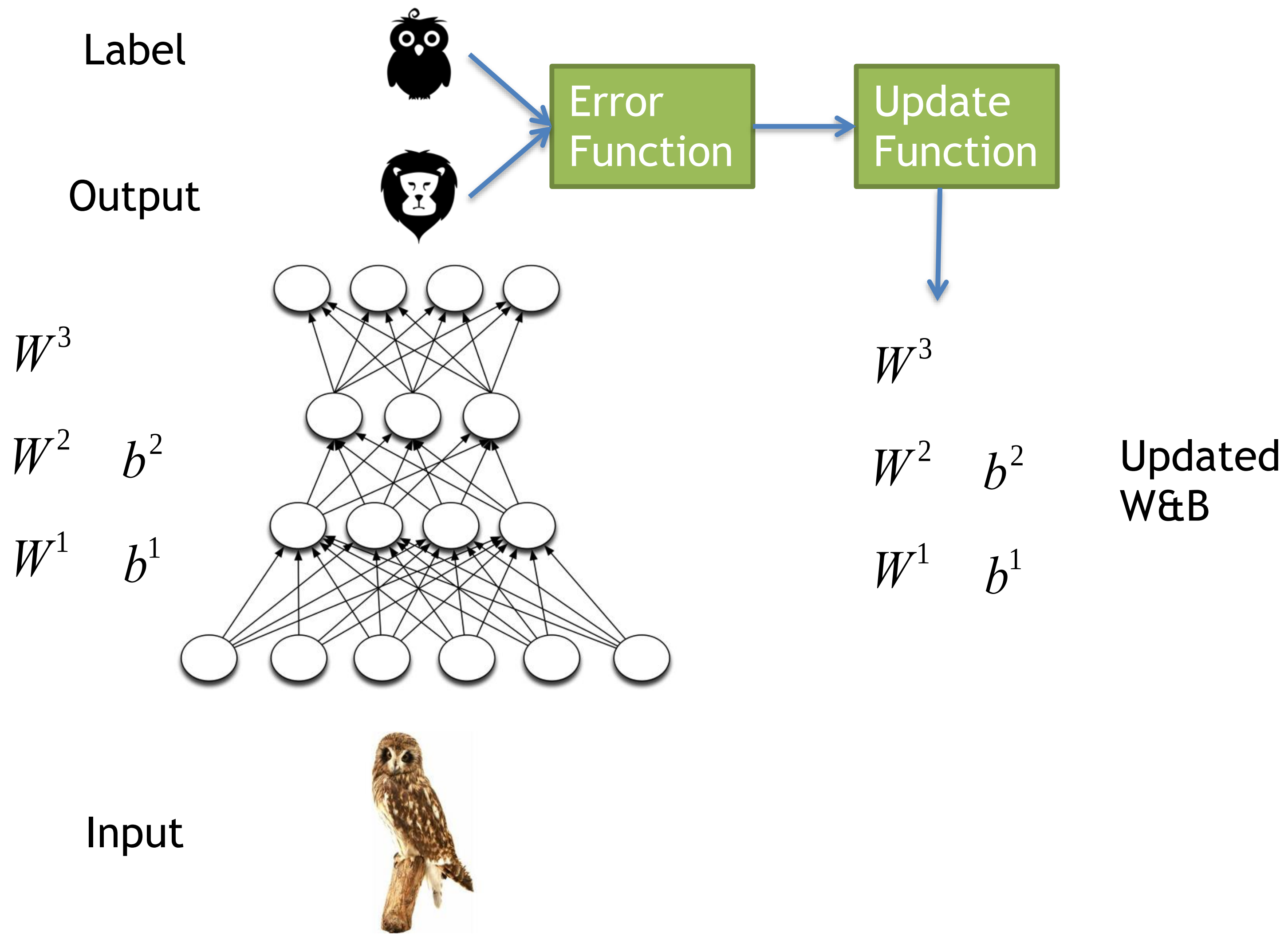
Once the architecture of the network is fixed its behaviour depends only on the values of its weights and biases



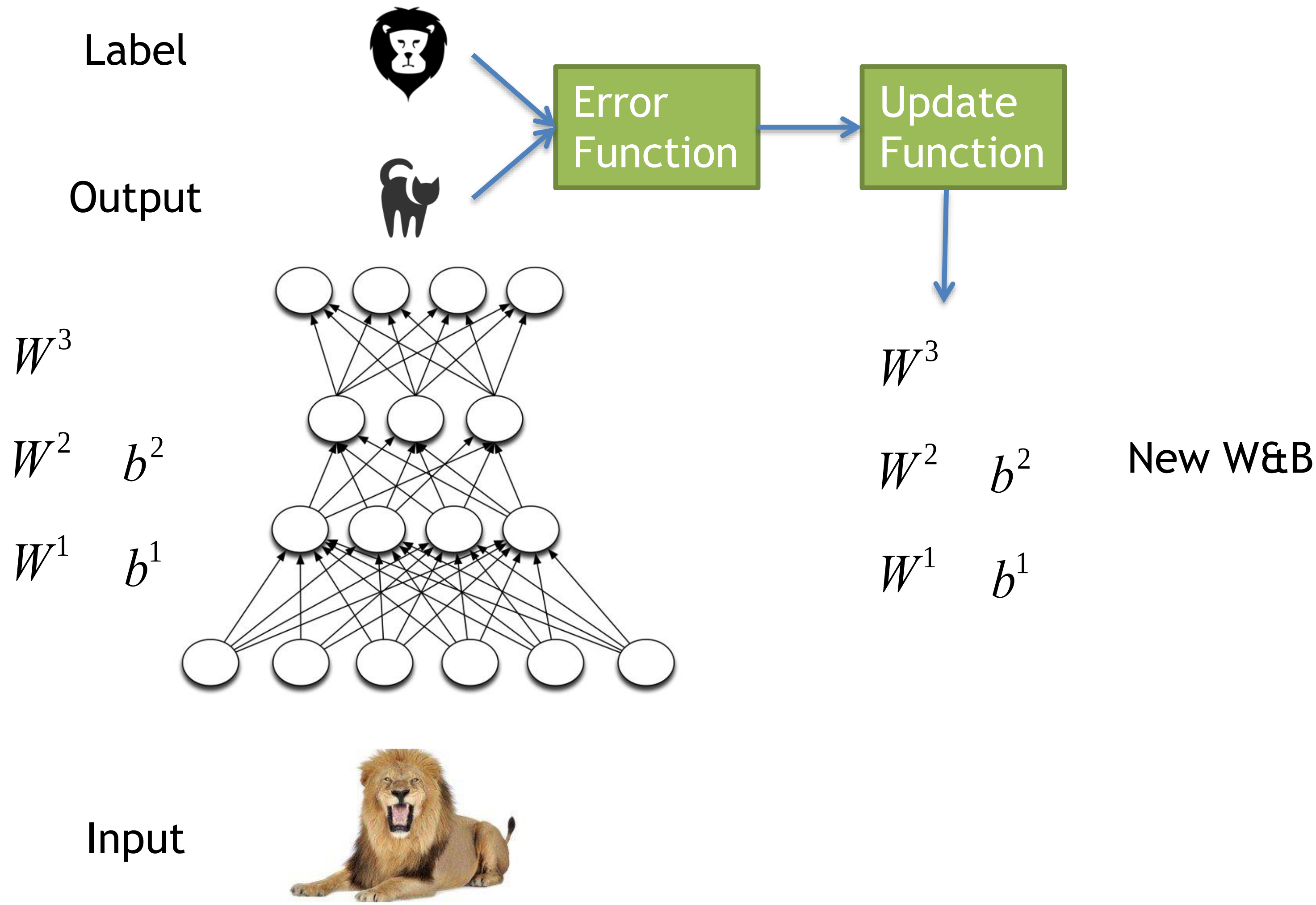
We start with random W&Bs and progressively adjust them with trial and error over a labelled data set

Labeled data set

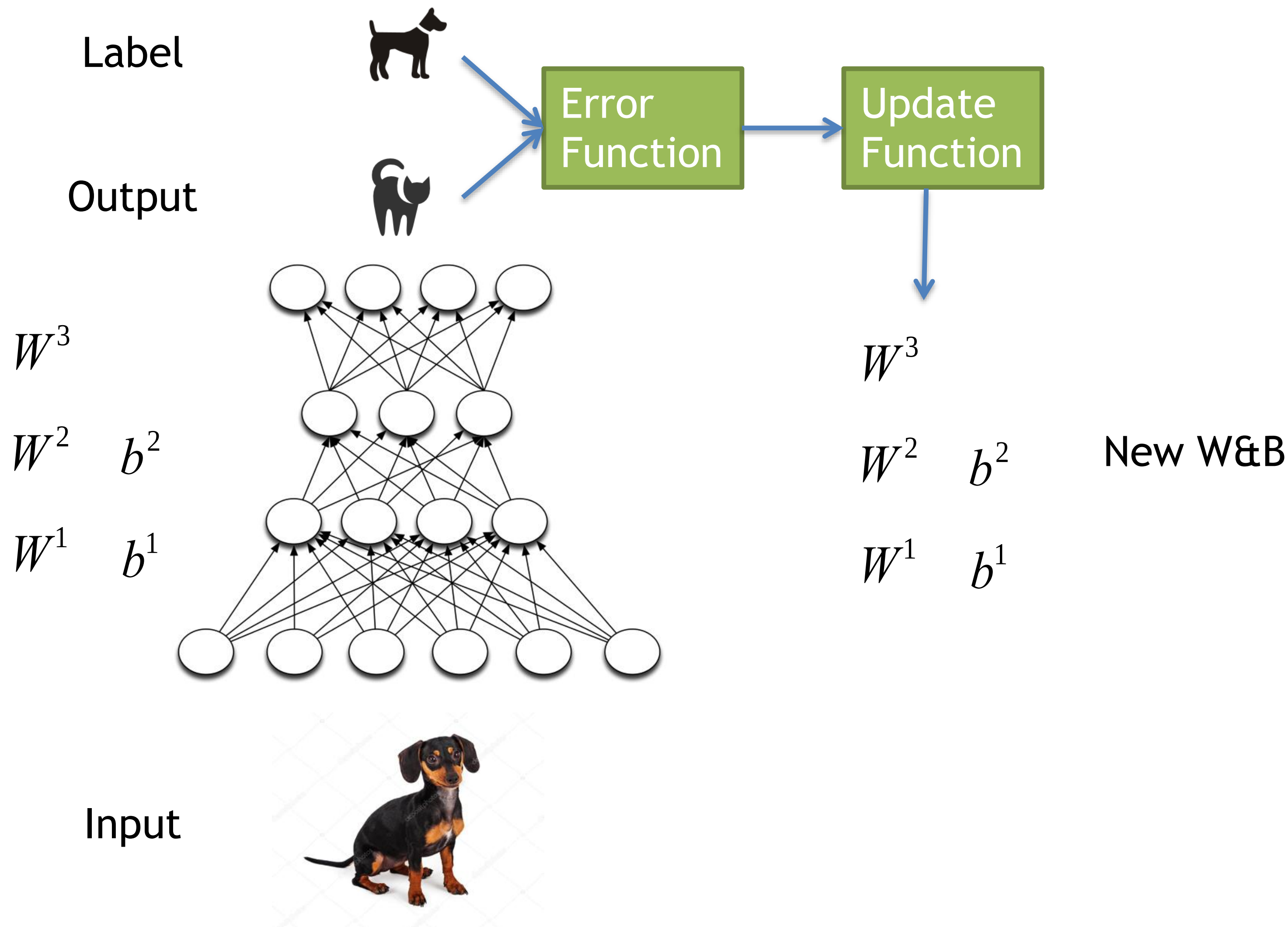
Learning by trial and error



Learning by trial and error



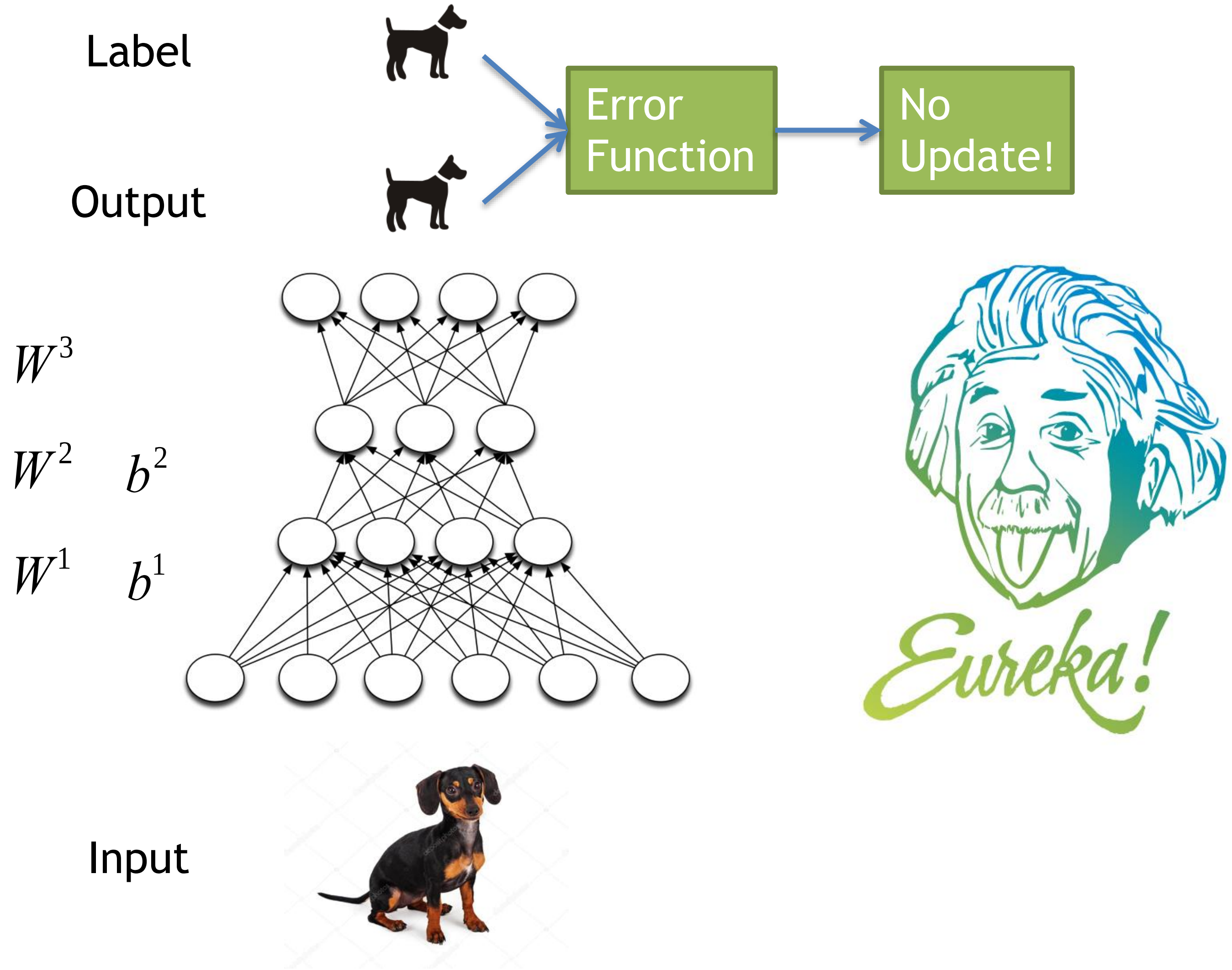
Learning by trial and error



Learning by trial and error

After many, many,…,many training sessions …

Learning by trial and error



What about the test set?

We got 75% accuracy!

Modelling with Uncertainty

1. If we don't exactly understand how something works, try a model with adjustable parameters
2. Use the error to refine the parameters

Using the error to improve the model

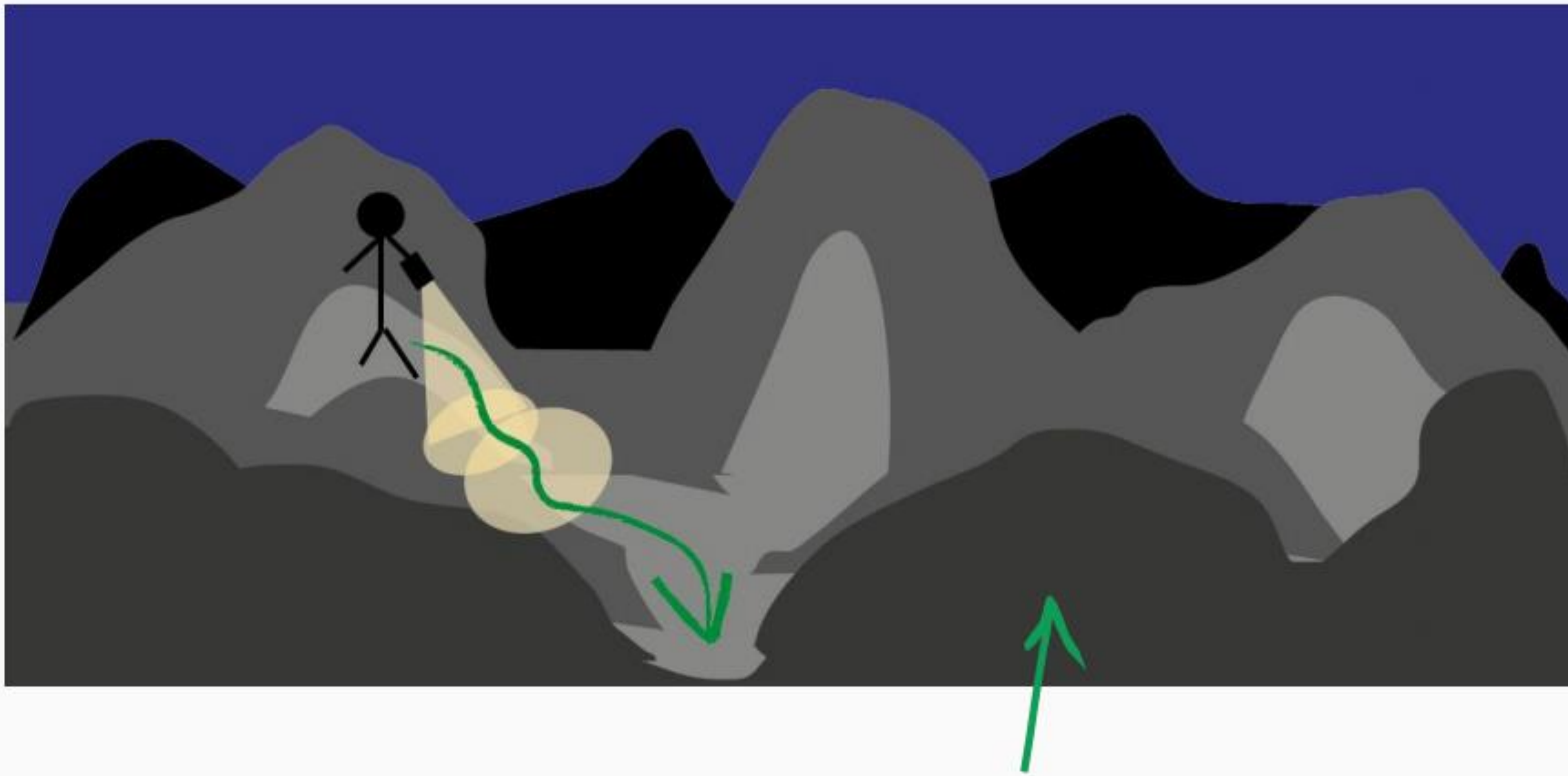
- Calculating the error at the output node is easy: the difference between the observed and the desired outputs
- It’s much less obvious what to do with network-internal errors: a heuristic is to split the error proportionally to the weights on the arcs
- Back-propagating the error can be expressed in terms of matrix multiplication too!

Yes, But How Do We Actually Update The Weights?

$$o_k = \frac{1}{1 + e^{-\sum_{j=1}^{3}\left(w_{j,k} \cdot \frac{1}{1+e^{-\sum_{i=1}^{3}(w_{i,j} \cdot x_i)}}\right)}}$$

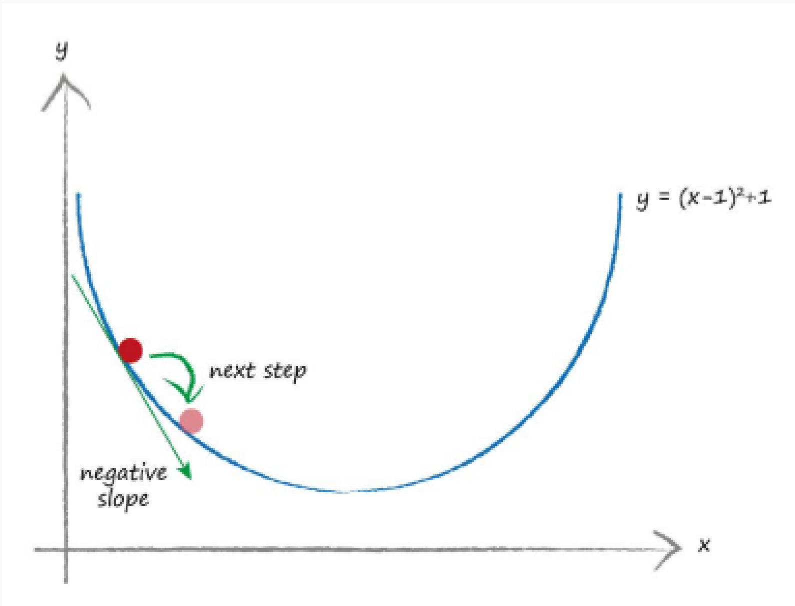
Aaarrrggghhh !!

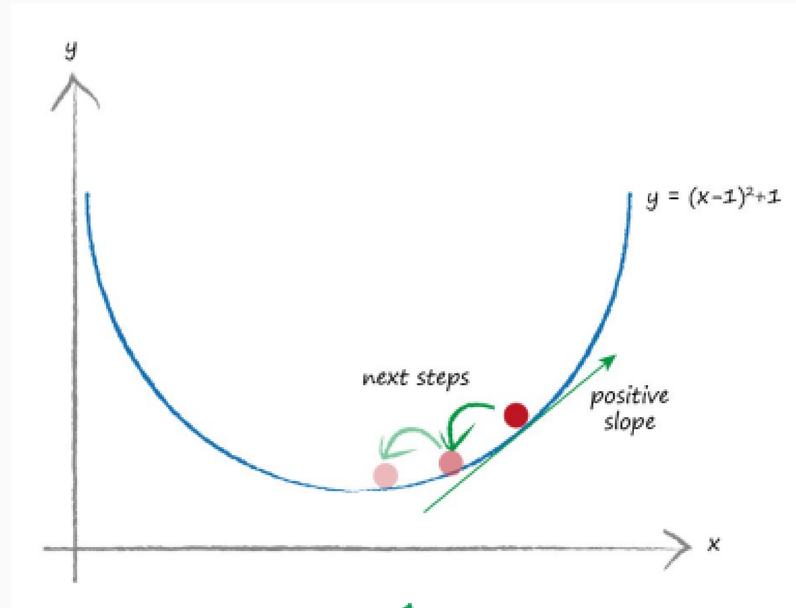
Perfect is the Enemy of Good



landscape is a complicated difficult mathematical function ..
... with all kinds of lumps, bumps, kinks ...

Gradient Descent





smaller gradient .. you're closer to the bottom ... take smaller steps?

How do we represent words?

One-hot vectors:

– dog = $(0,0,0,0,1,0,0,0,0,\ldots)^T$

– cat = $(0,0,0,0,0,0,0,1,0,\ldots)^T$

– eat = $(0,1,0,0,0,0,0,0,0,\ldots)^T$

– etc …

- multidimensional space, *huge* vectors, so limit vocabulary to (say) 20K tokens, OTHER for rest

Similar words behave similarly!

- *but the cute dog jumped*
- *but the cute cat jumped*

- *child hugged the cat tightly*
- *child hugged the dog tightly*

- *like to watch cat videos*
- *like to watch dog videos*

cf. Firth's "you shall know a word by the company it keeps"

Word Embeddings

Each word is represented by a vector of numbers that positions the word in a multi-dimensional space, e.g.:

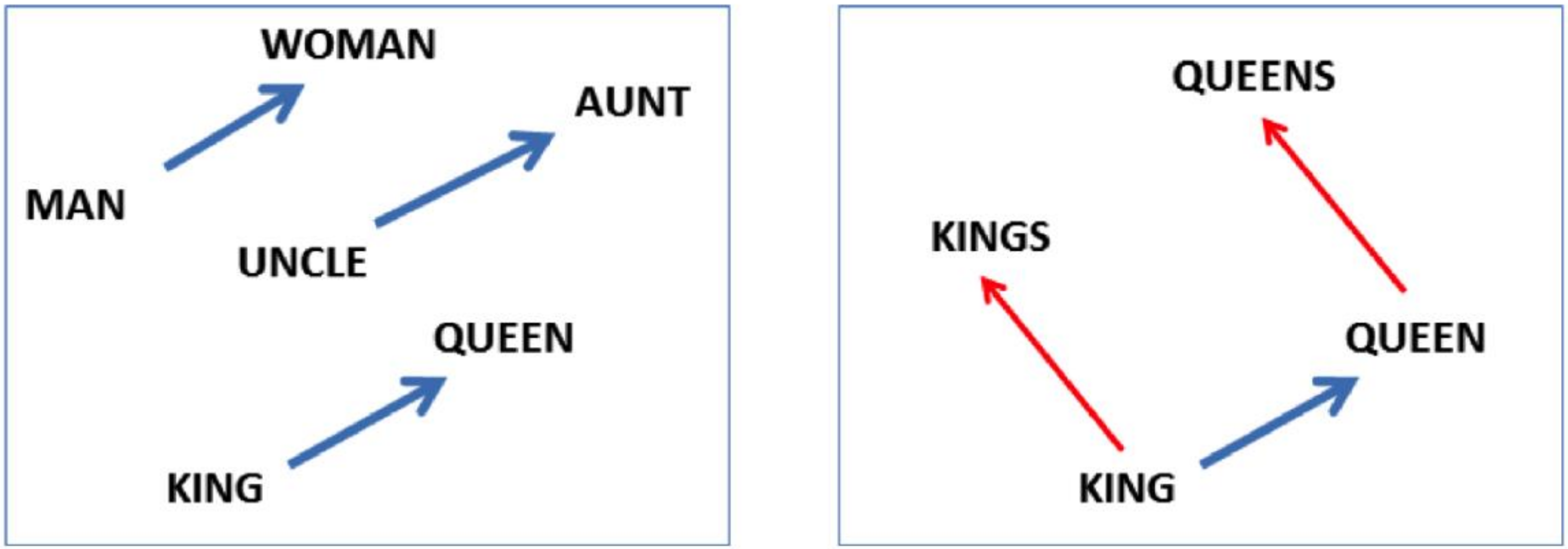
$$king = < 55, -10, 176, 27 >$$

$$man = < 10, 79, 150, 83 >$$

$$woman = < 15, 74, 159, 106 >$$

$$queen = < 60, -15, 185, 50 >$$

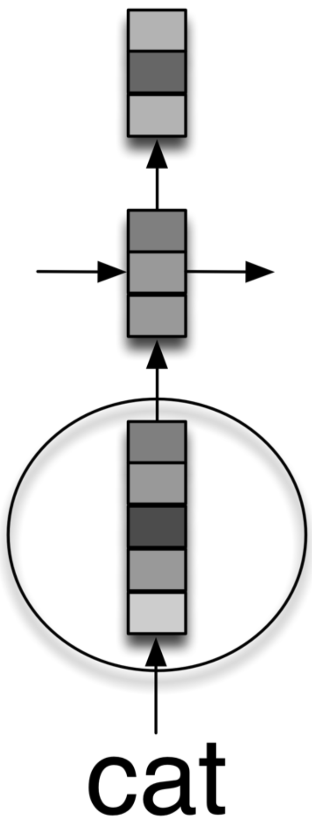
Word Embeddings

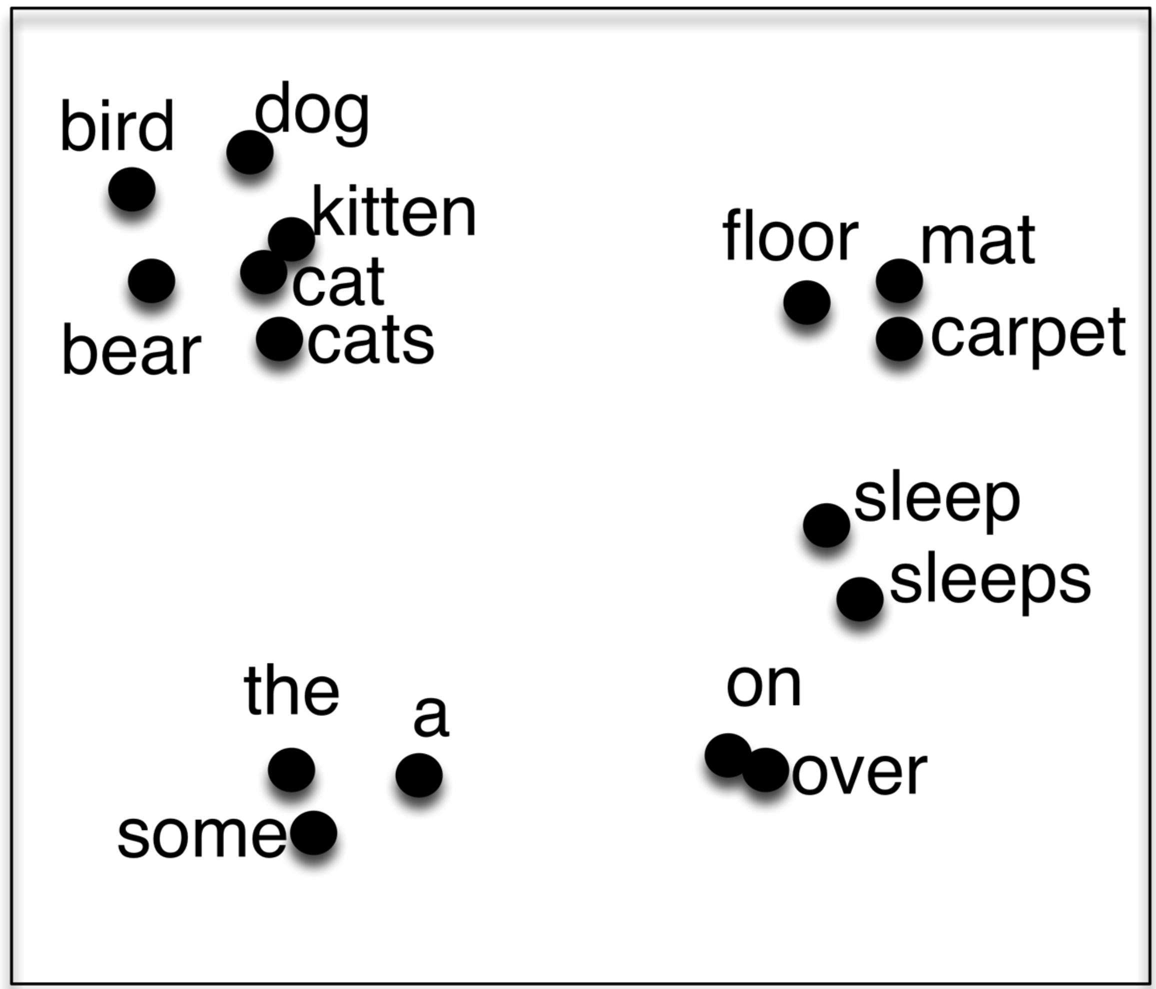


$$vec(King) - vec(Man) + vec(Woman) \approx vec(Queen)^2$$

[2] Linguistic Regularities in Continuous Space Word Representations (Mikolov et al., 2013)

Word embedding





Word Embedding using LM Prediction

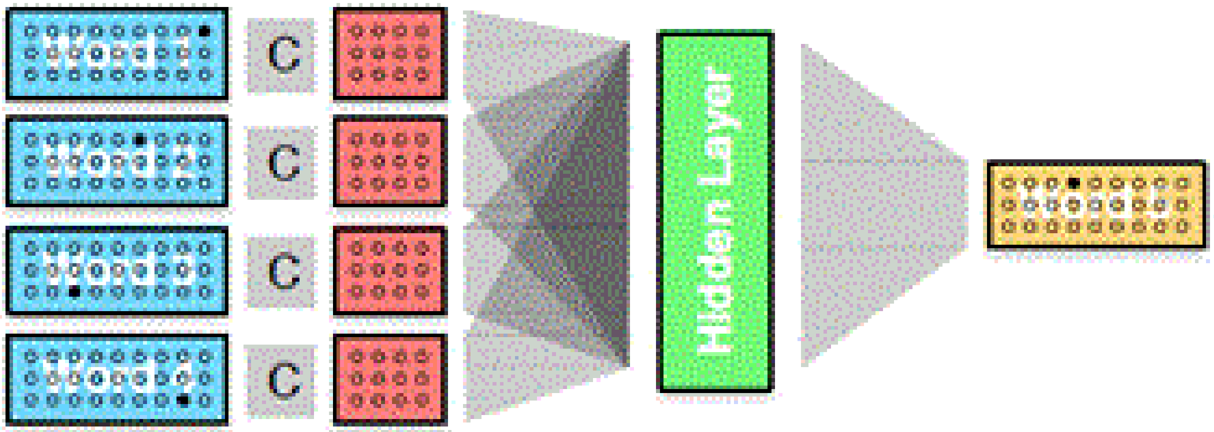


Figure 13.11: Full architecture of a feed-forward neural network language model. Context words $(w_{i-4}, w_{i-3}, w_{i-2}, w_{i-1})$ are represented in a one-hot vector, then projected into continuous space as word embeddings (using the same weight matrix C for all words). The predicted word is computed as a one-hot vector via a hidden layer.

- Enable generalization between words (*clustering*)
- Robust prediction in unseen context (*back-off*)

Are feed-forward NLMs restricted?

- FFLMs can use (much) longer contexts than traditional statistical back-off models. e.g. up to 30-gram models are being used!

Are feed-forward NLMs restricted?

- FFLMs can use (much) longer contexts than traditional statistical back-off models. e.g. up to 30-gram models are being used!
- But we are still using a fixed content word window (5-gram in the previous example) …

Are feed-forward NLMs restricted?

- FFLMs can use (much) longer contexts than traditional statistical back-off models. e.g. up to 30-gram models are being used!
- But we are still using a fixed content word window (5-gram in the previous example) …
- Recurrent NLMs can condition on *any* length context sequences by reusing the hidden layer when predicting w_n as additional input to predict word w_{n-1}

Recurrent Neural Networks

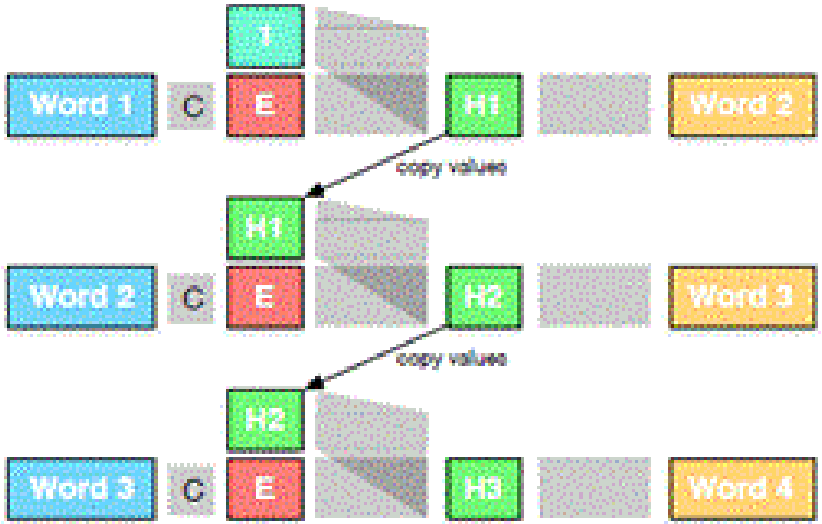
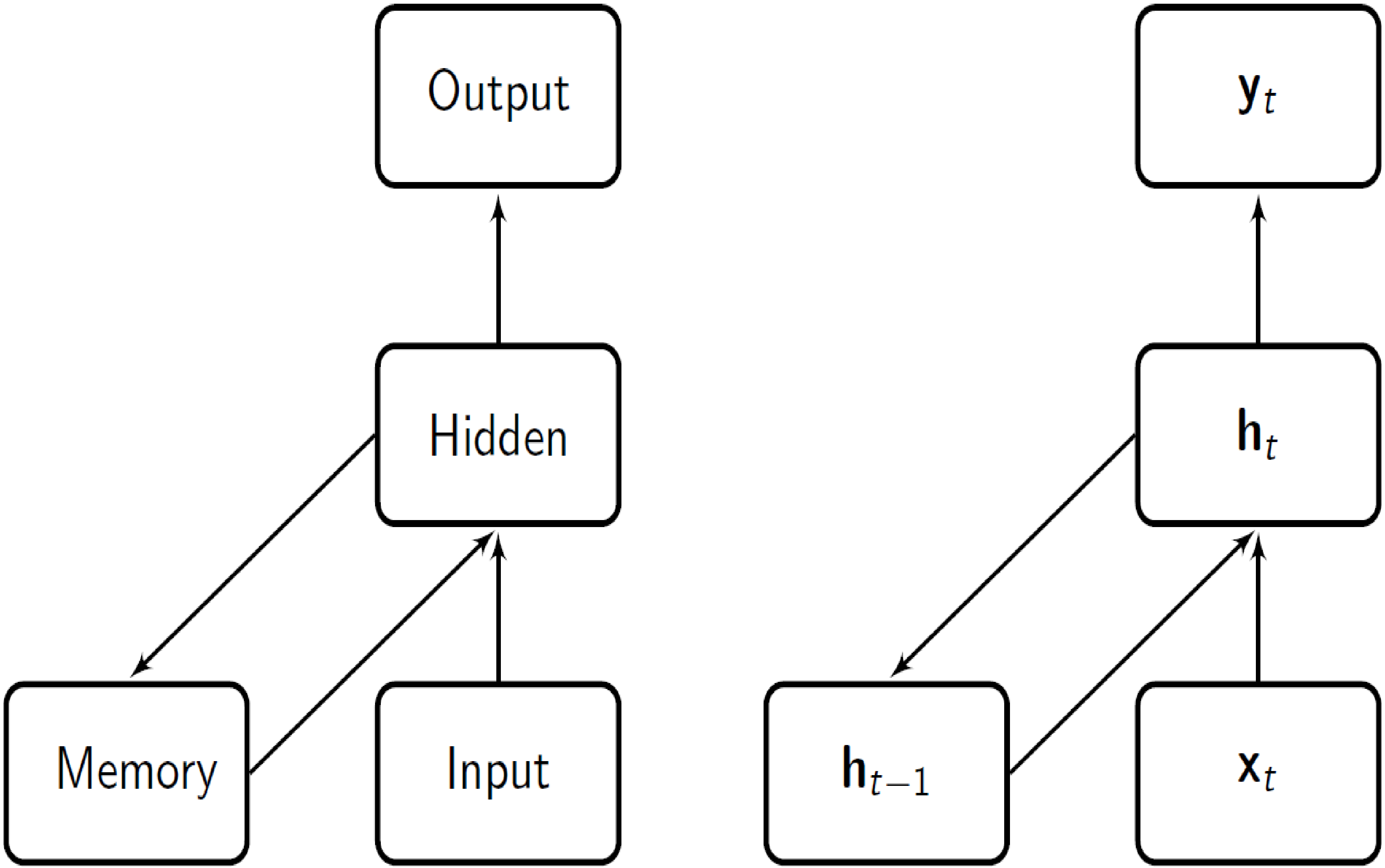


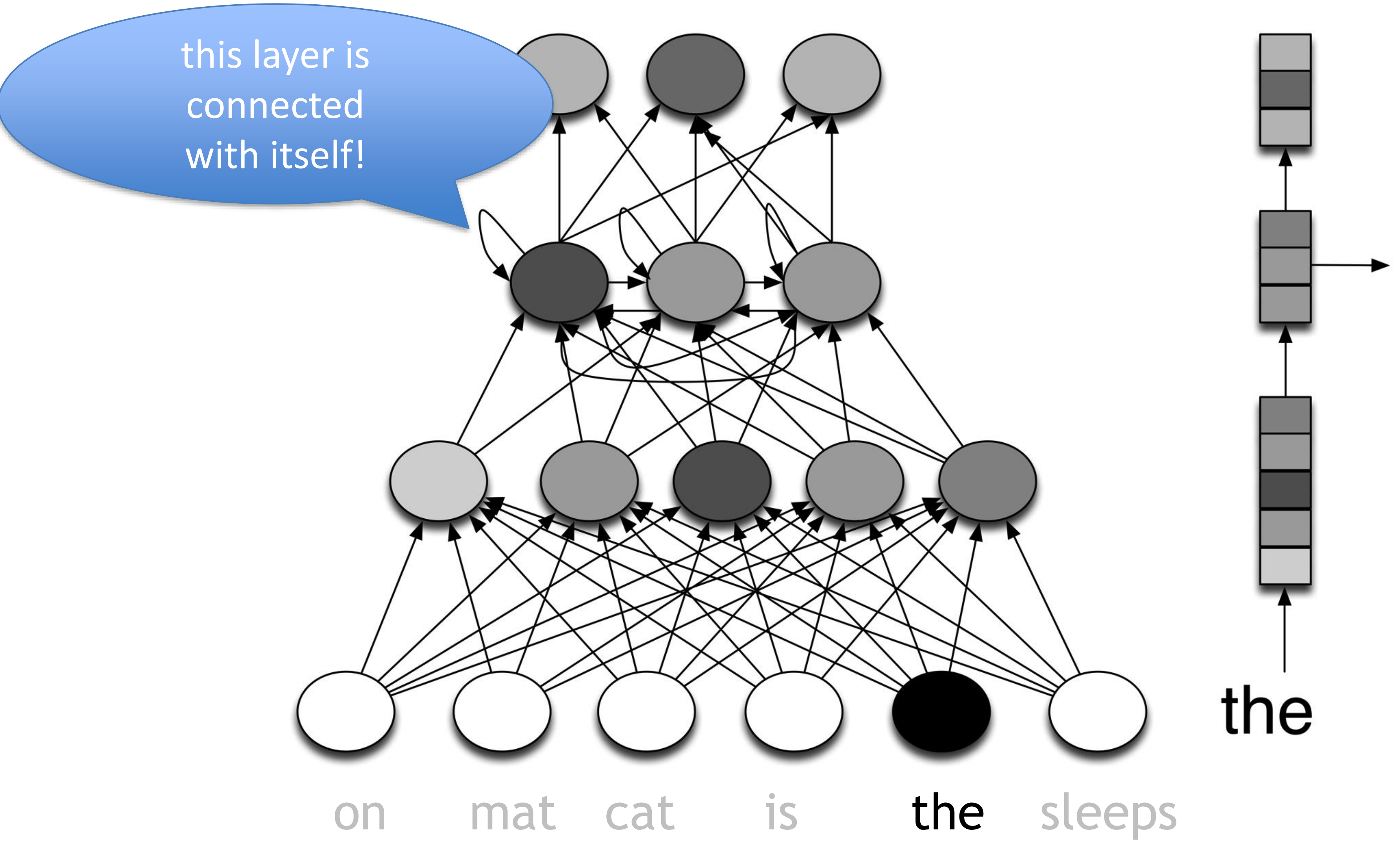
Figure 13.13: Recurrent neural language models: After predicting Word 2 in the context of following Word 1, we re-use this hidden layer (alongside the correct Word 2) to predict Word 3. Again, the hidden layer of this prediction is re-used for the prediction of Word 4.

- RNNs useful for processing sequential input (like language)
- Using an RNN we process our sequential data one input at a time
- In an RNN the outputs of some of the neurons for one input are fed back into the network as part of the next input
- The ‘copy value’ encodes the previous context – part of a sentence in MT – in the sequence
- At the end, we have the full history of the sentence!

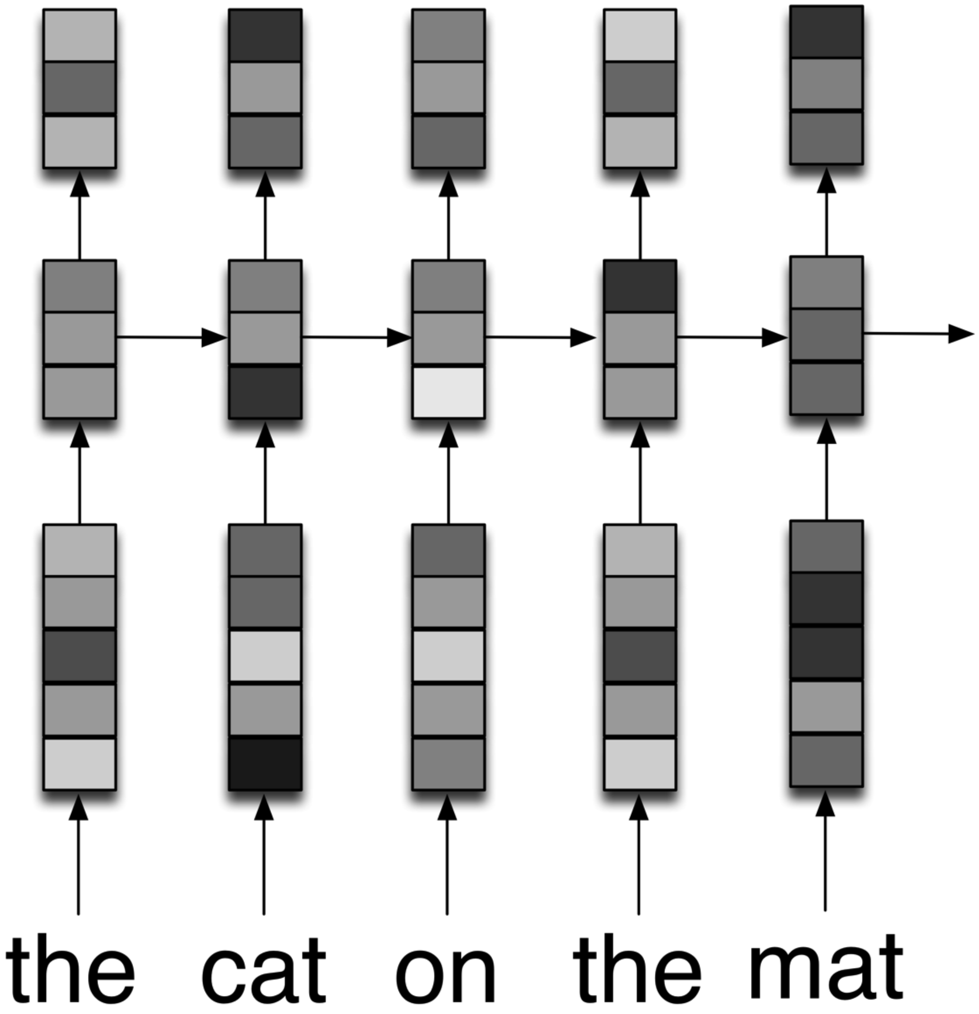
Recurrent Neural Networks



Recurrent neural networks



Time Unfolded RNN



Overview

Engaging Content
Engaging People

- Basic building blocks: functions & neurons
- Neural Networks
  - Feed-Forward Neural Networks
  - Inputs, Outputs, Weights, Errors
  - Word Embeddings
  - Recurrent Neural Networks
- Neural Machine Translation: Architecture
  - Encoders
  - Decoders (language models)
  - Attention
- Neural Machine Translation: Improvements
- Future Work in NMT
- Concluding Remarks

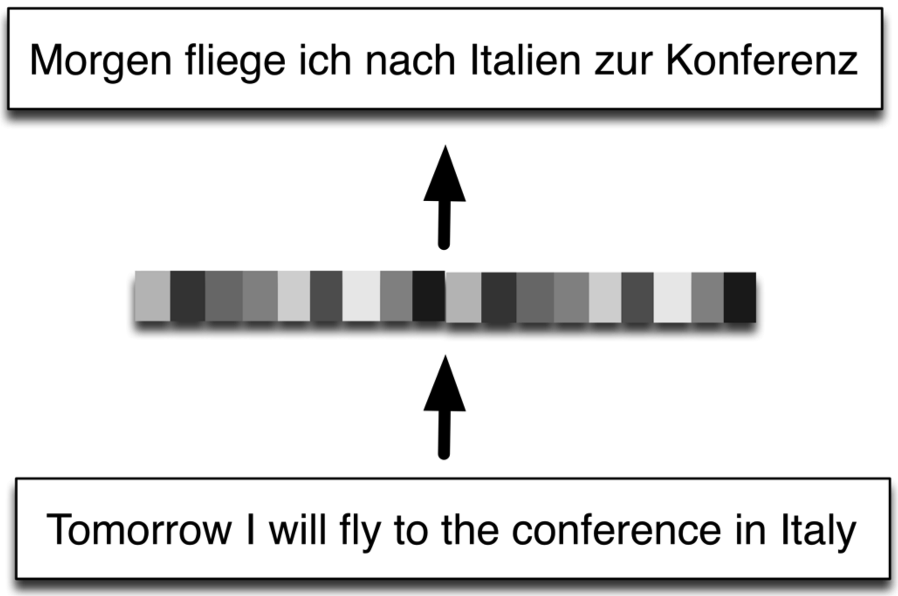
Machine Translation

English

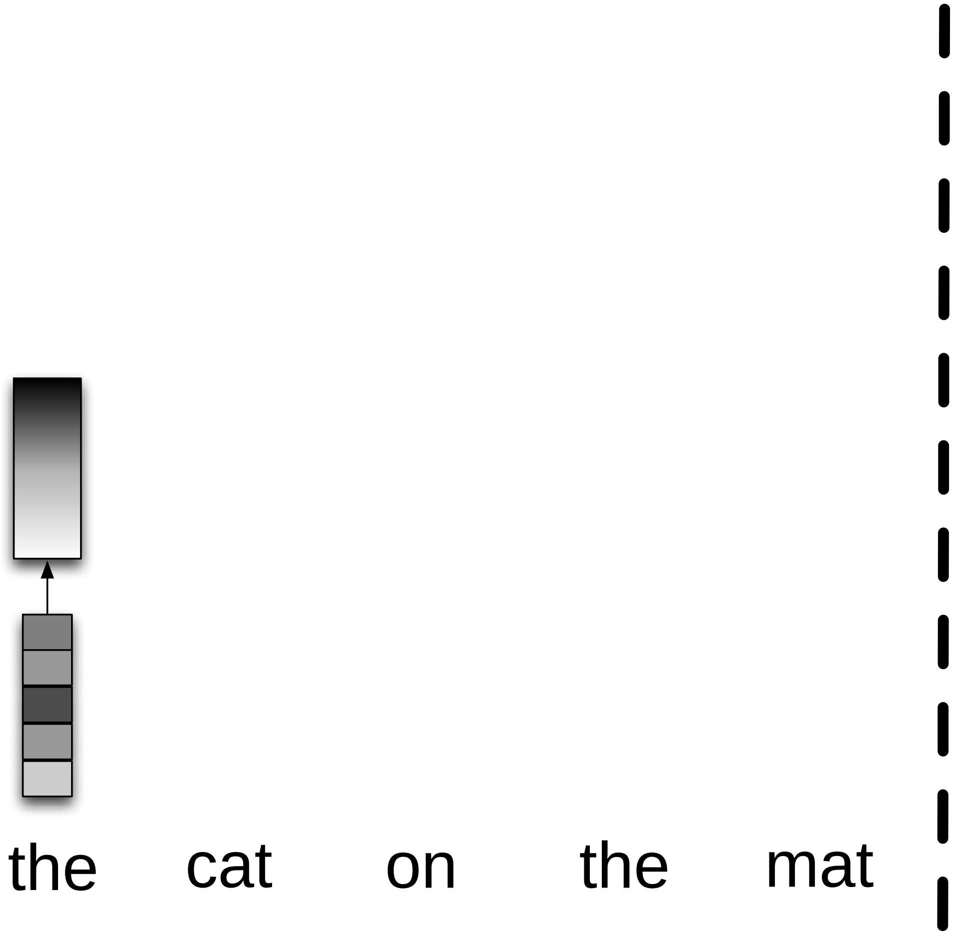
German

Source → **Machine Learning** → **Target**

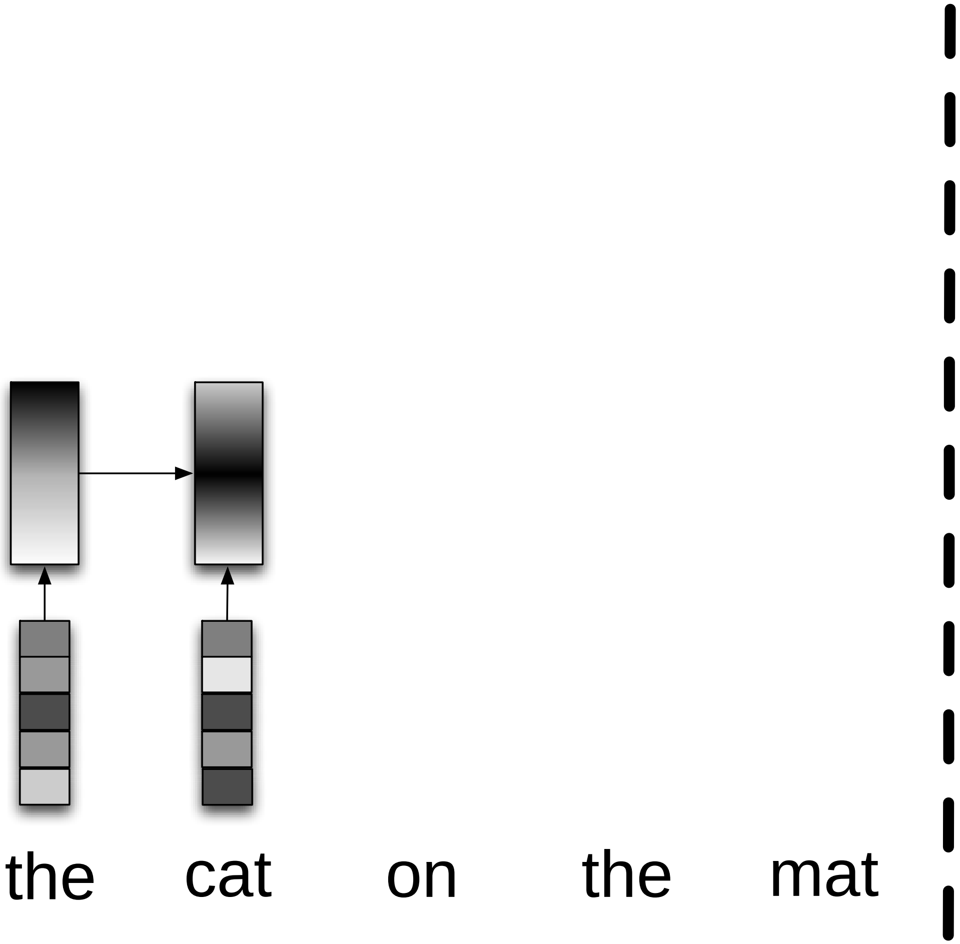
Phrase-based vs. Neural MT



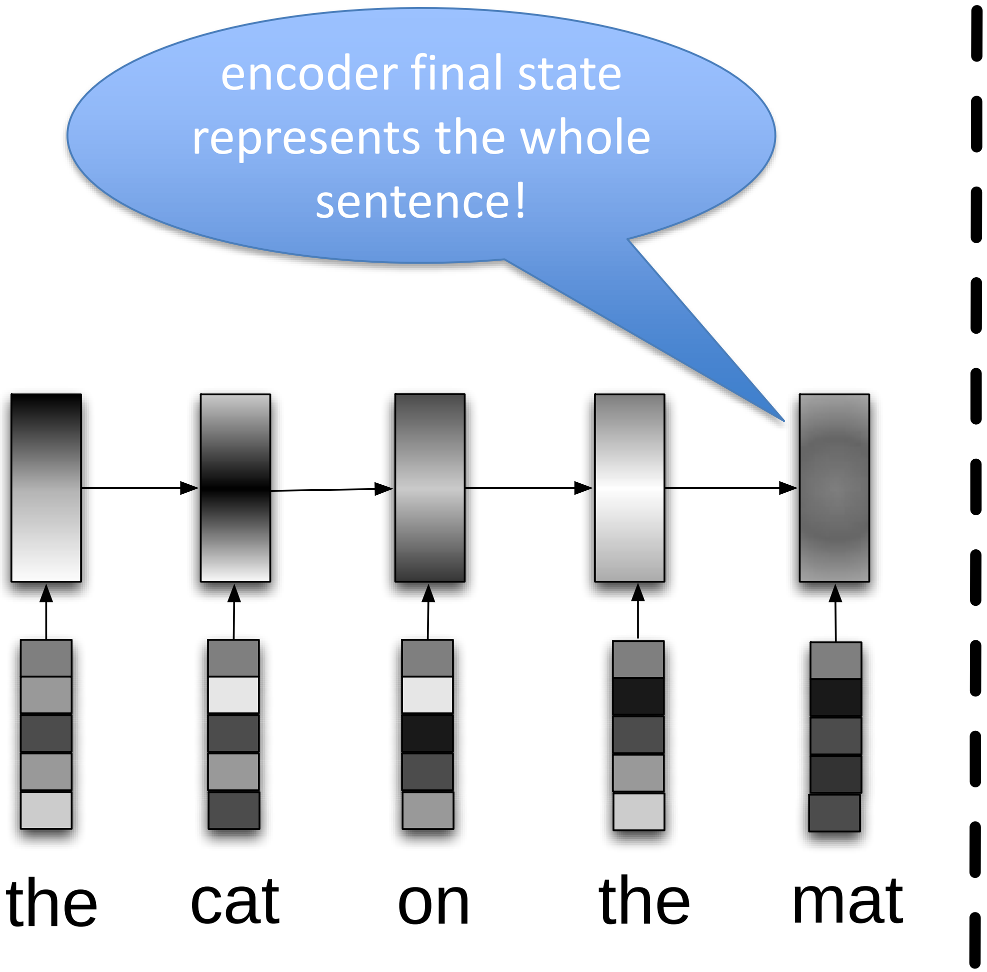
NMT: encoder-decoder



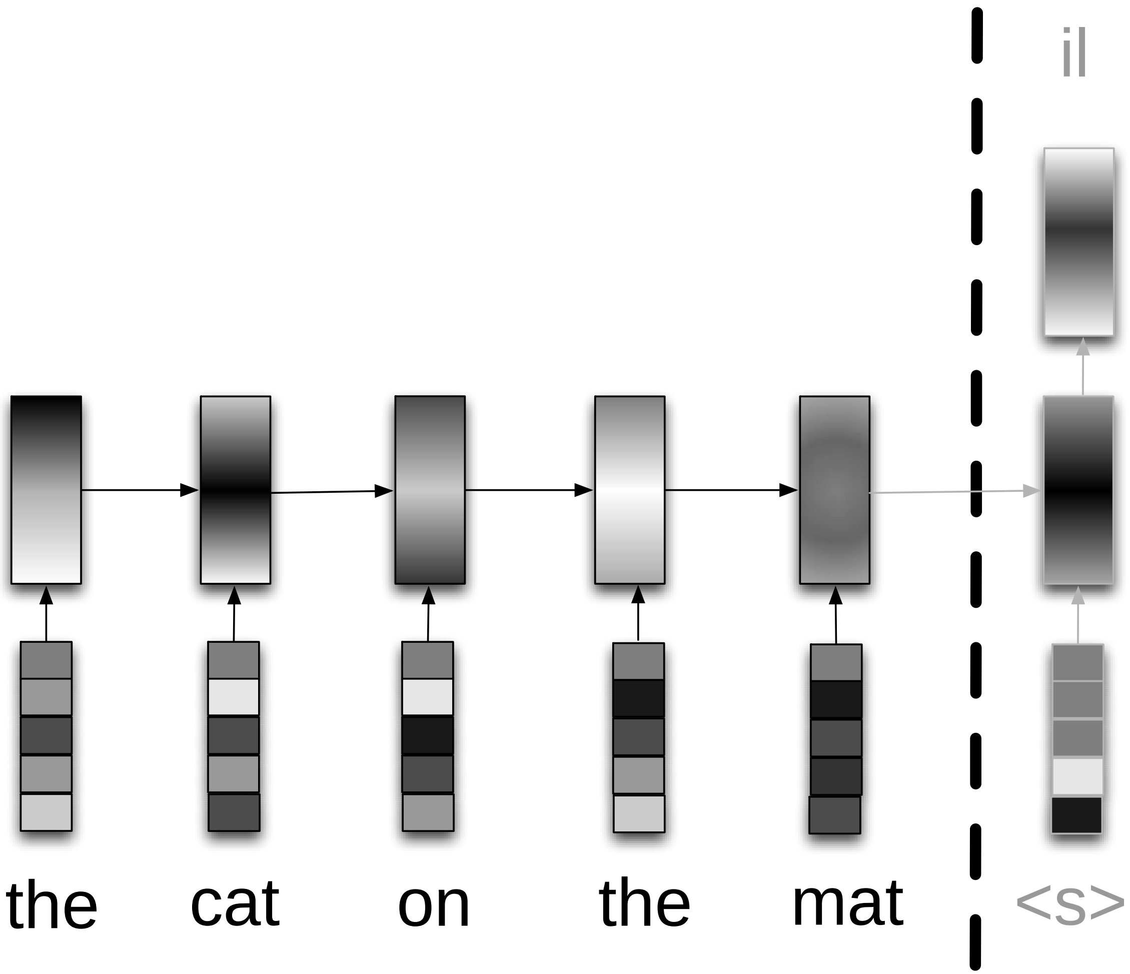
NMT: encoder-decoder



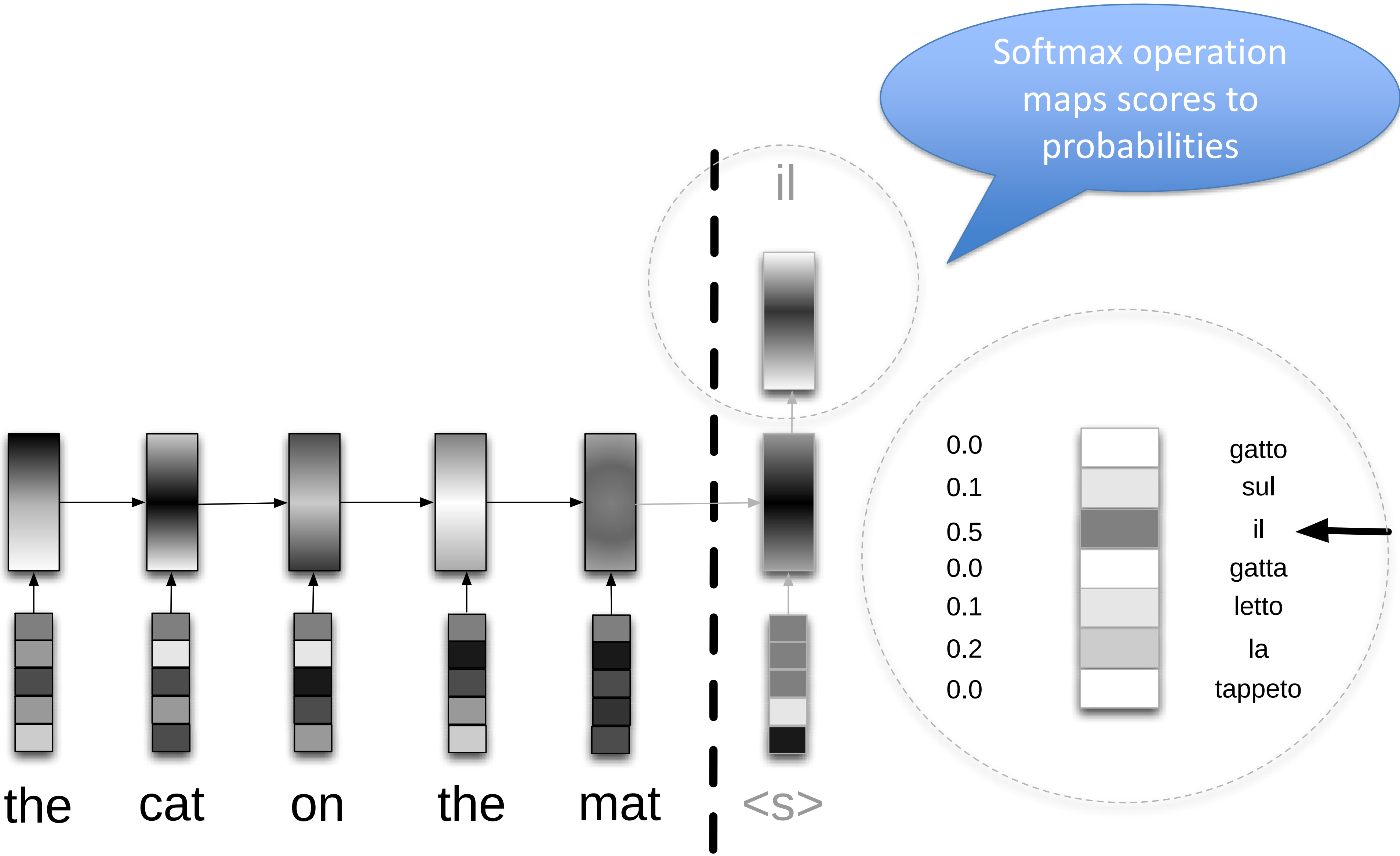
NMT: encoder-decoder



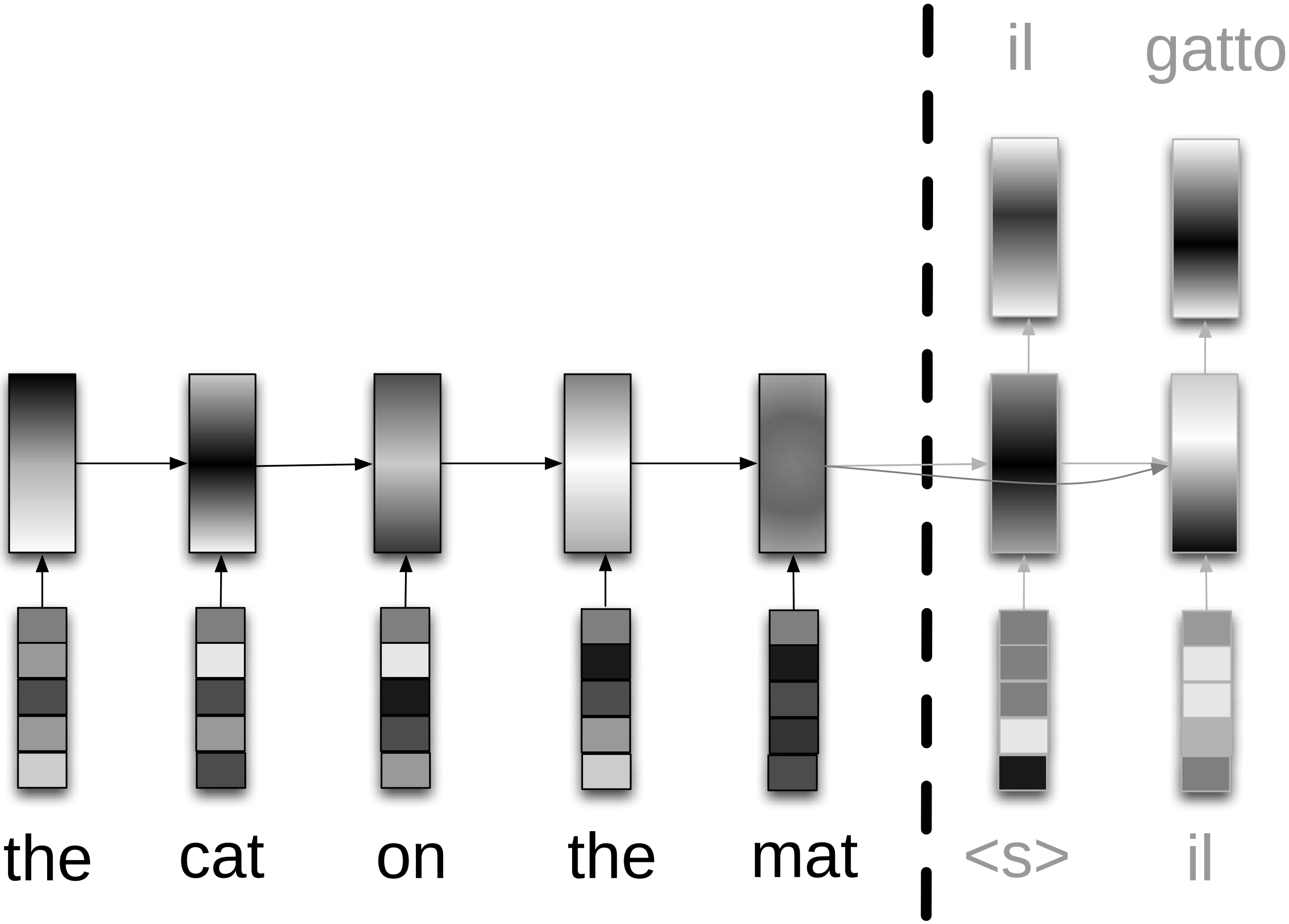
NMT: encoder-decoder



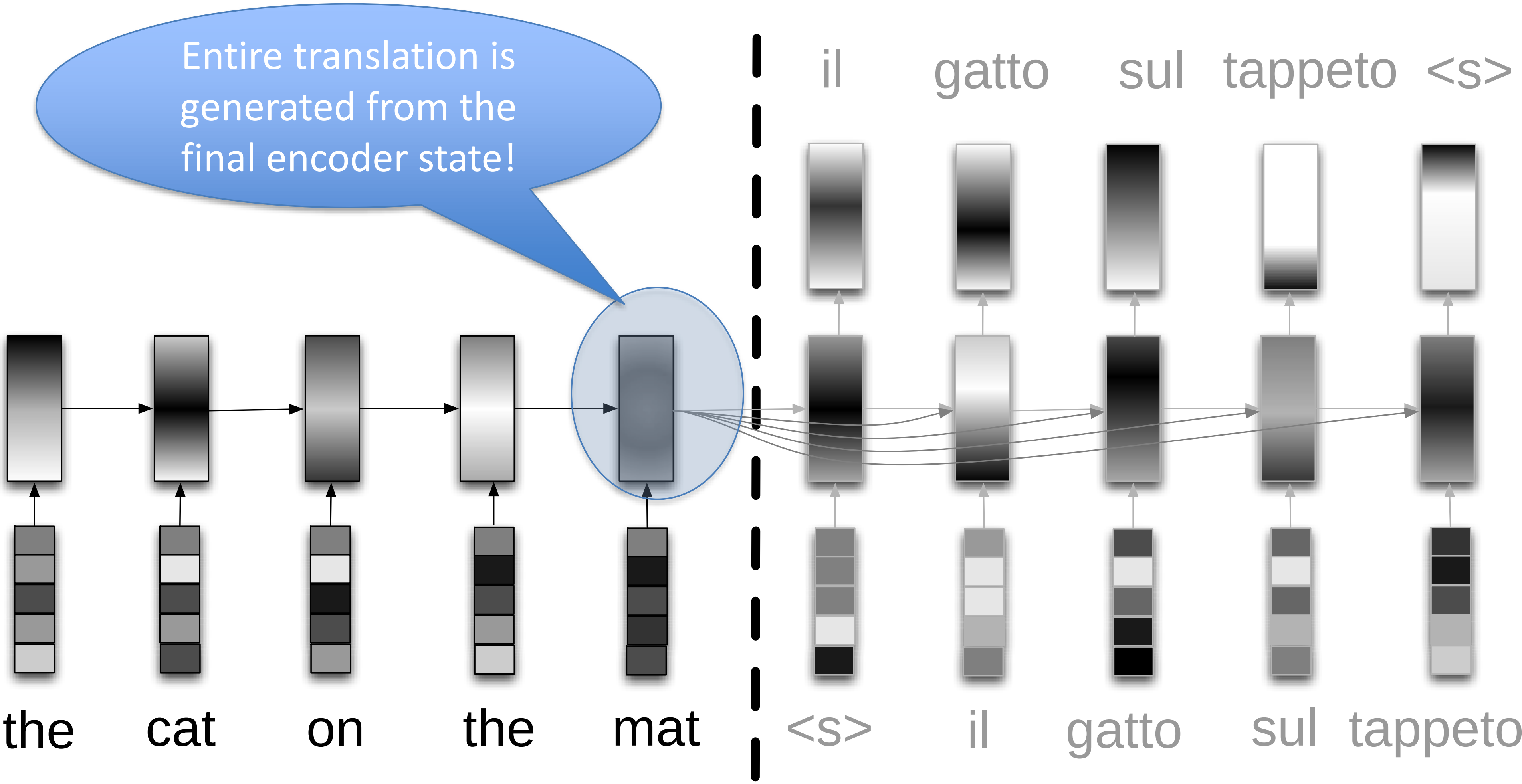
NMT: encoder-decoder



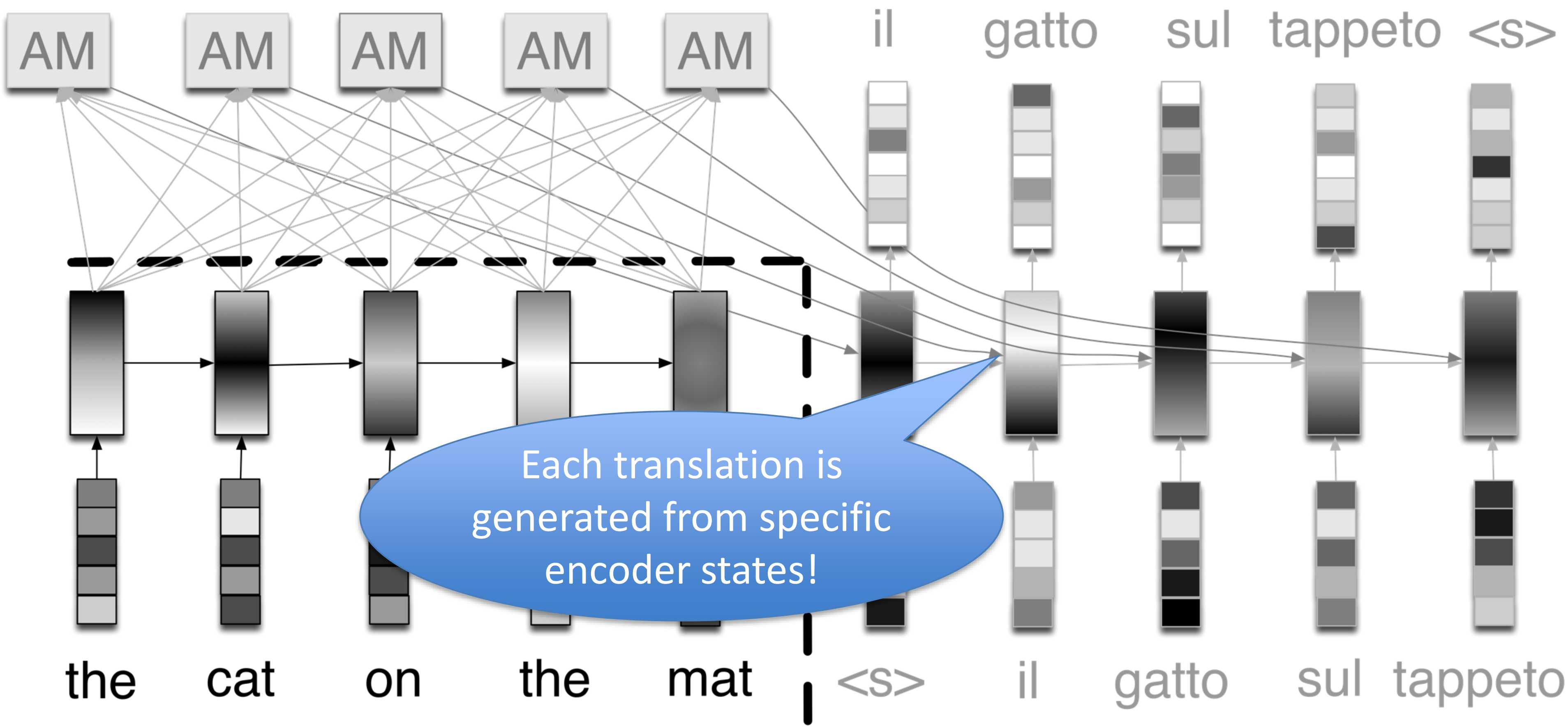
NMT: encoder-decoder



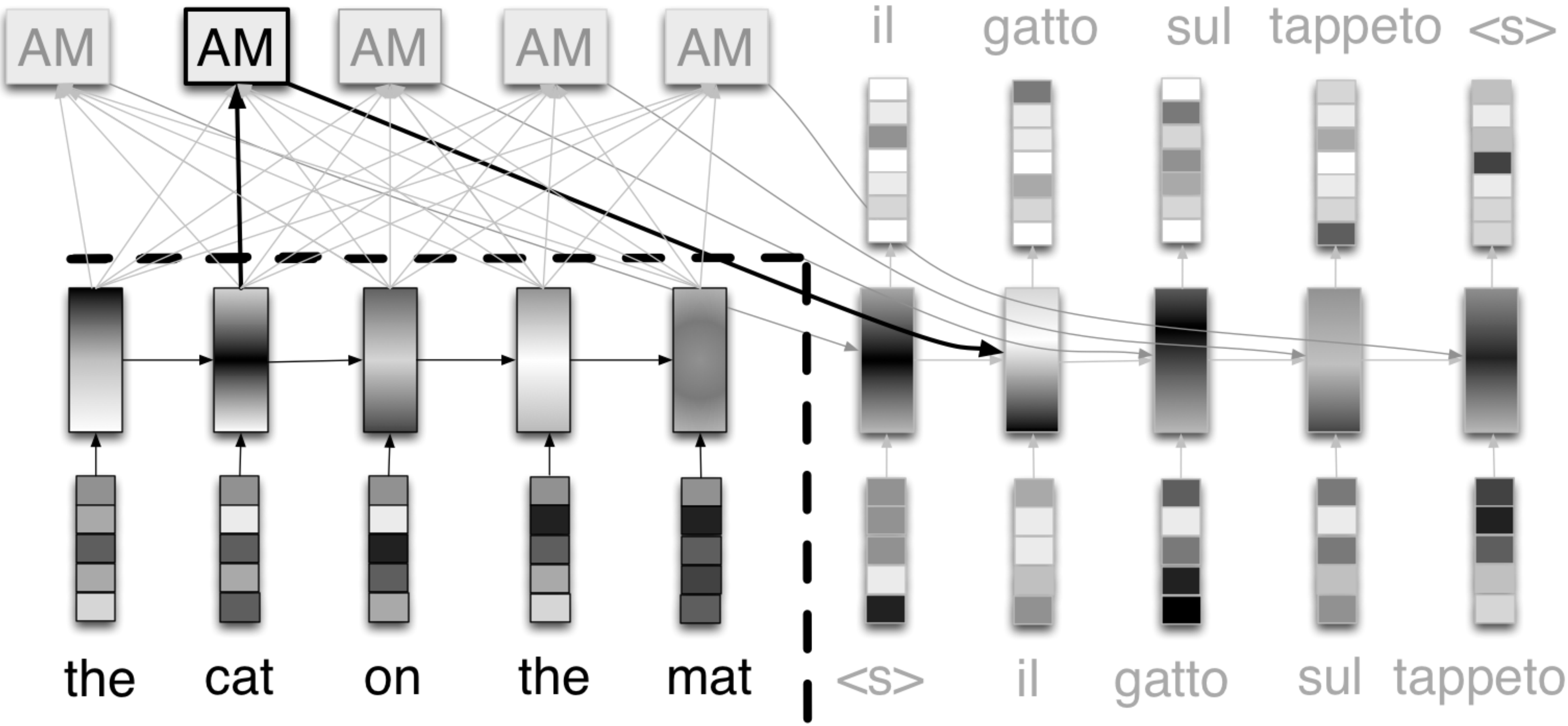
NMT: encoder-decoder



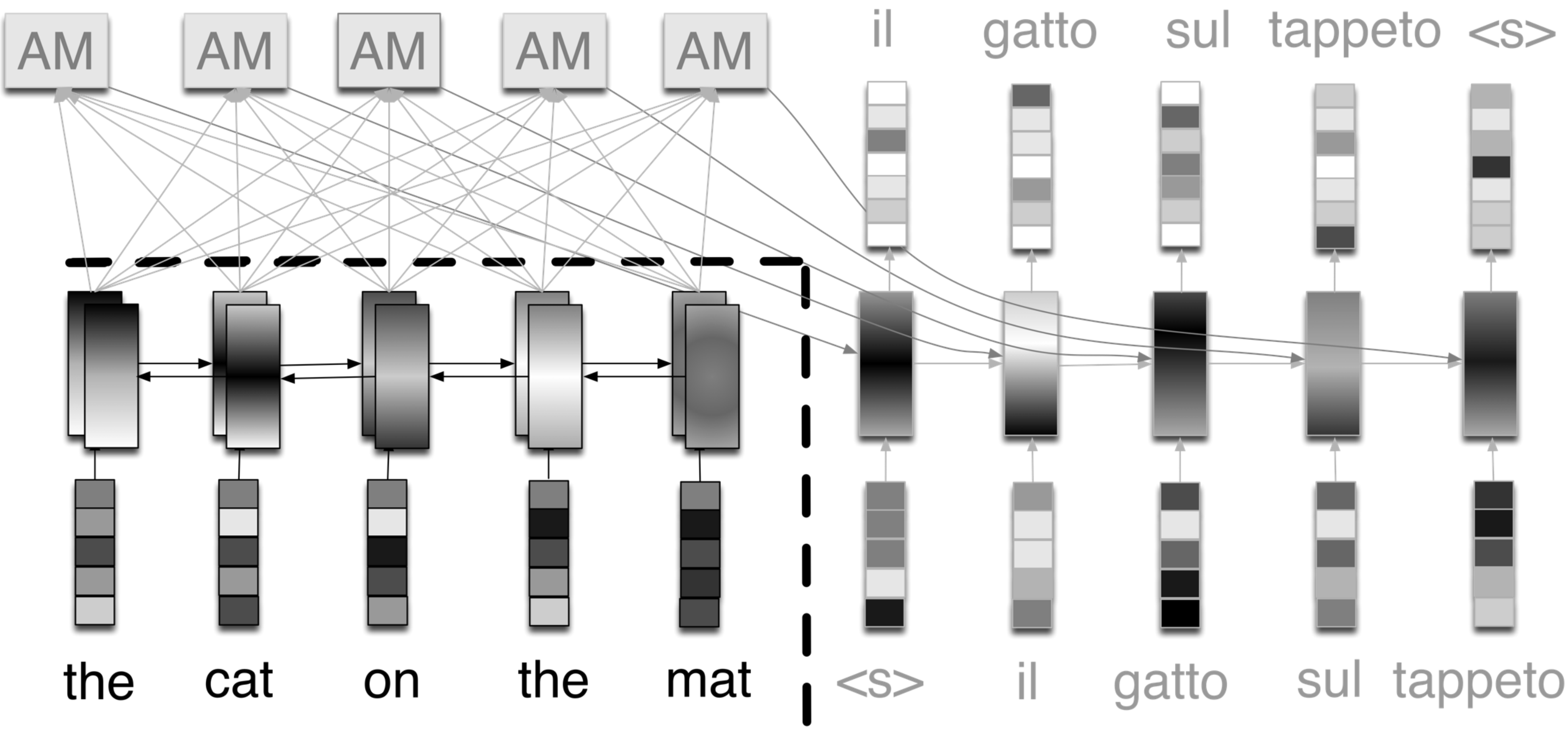
NMT with attention model



NMT with attention model

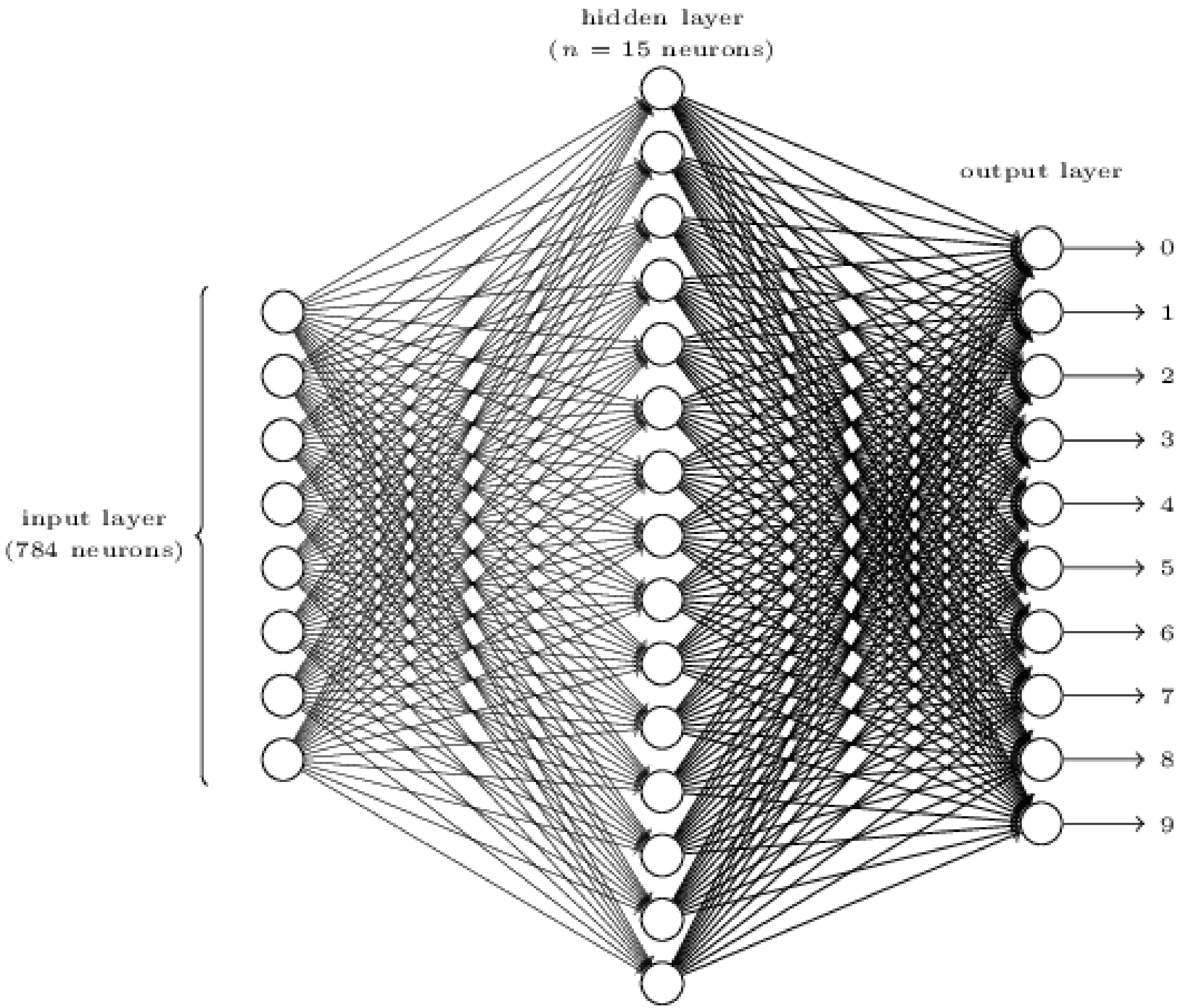


NMT with attention model

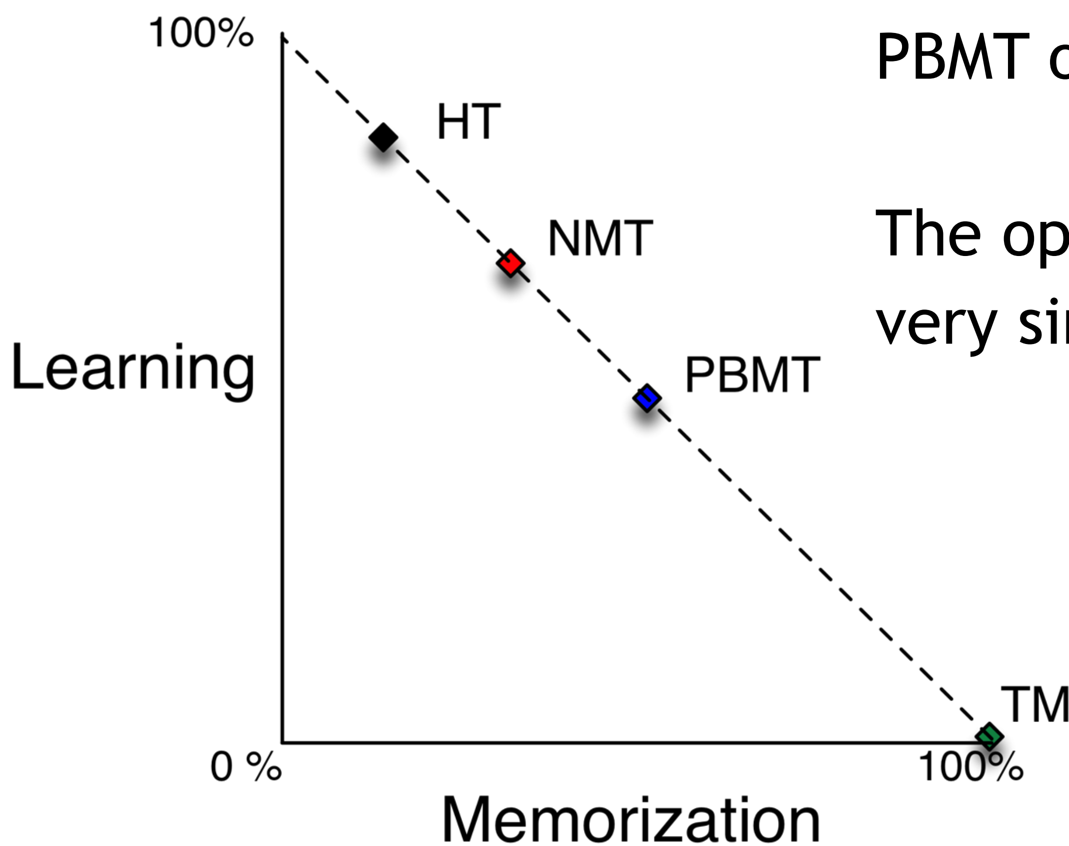


Bidirectional RNN (Bahdanau et al., 2015)

A neural net for handwriting recognition



PBMT and NMT learn differently



NMT tends to be better than PBMT on **new** sentences.

The opposite happens on inputs very similar to training examples.

Thanks to Marcello Federico for this observation

PBMT and NMT learn differently

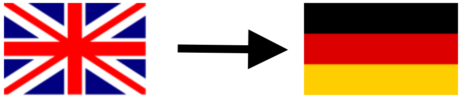
NMT have a long history – longer than PBMT – but started to outperform PBMT in Autumn of 2015

- Which are the strengths of NMT?
- On which linguistic phenomena?

Overview

- Basic building blocks: functions & neurons
- Neural Networks
  - Feed-Forward Neural Networks
  - Inputs, Outputs, Weights, Errors
  - Word Embeddings
  - Recurrent Neural Networks
- Neural Machine Translation: Architecture
  - Encoders
  - Decoders (language models)
  - Attention
- Neural Machine Translation: Improvements
- Future Work in NMT
- Concluding Remarks

IWSLT 2015 MT Task



Evaluation Data

tst 2015 HE SET
12 TED Talks
- 600 src sentences
- ~10K src words

PBSY (Huck & Birch, '15)	PBSY Post-Edit
HPB (Jehl et al., '15)	HPB Post-Edit
SPB (Ha et al.,'15)	SPB Post-Edit
NMT (Luong & Manning, '15)	NMT Post-Edit

Evaluation Metrics

TER: traces the edits done by post-editors

PBSY (Huck & Birch, 2015)

mTER

PBSY Post-Edit
HPB Post-Edit
SPB Post-Edit
NMT Post-Edit

Reliable and informative since post-editors' variability is controlled → ***Analyses of overall quality***

cf. Bentivogli et al. (EMNLP, 2016)

Evaluation Metrics

TER: traces the edits done by post-editors

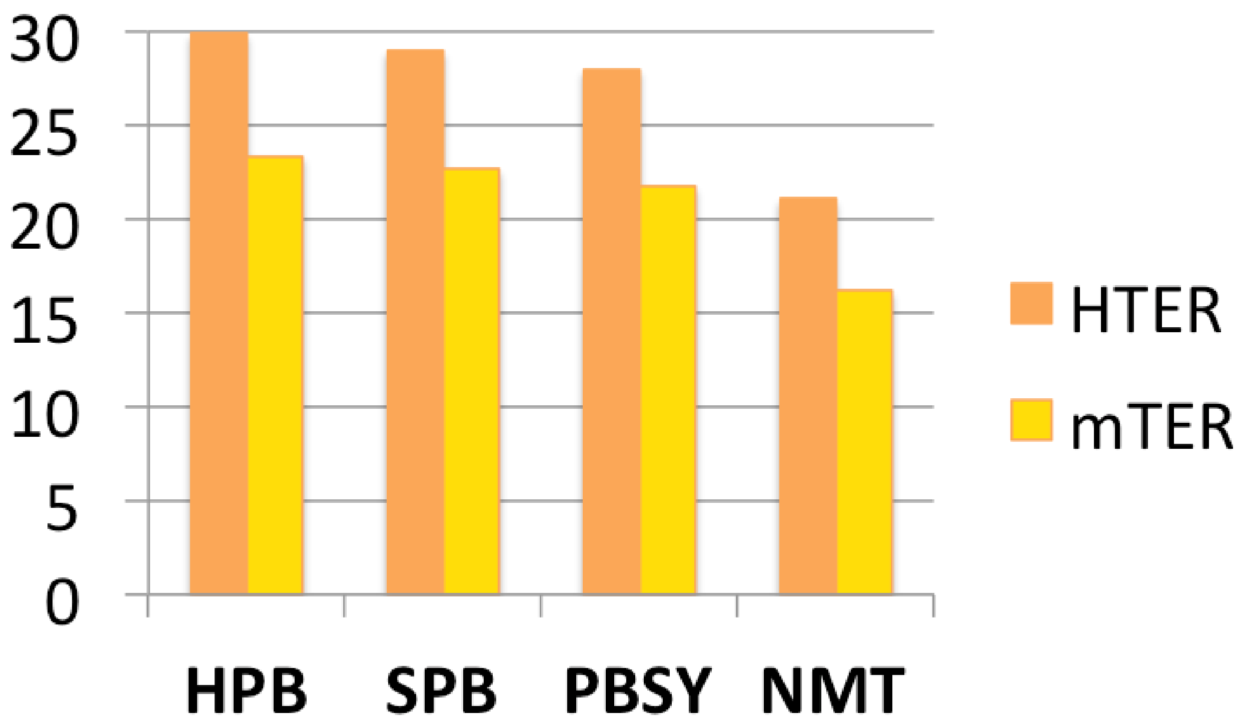
PBSY (Huck & Birch, 2015)

PBSY Post-Edit

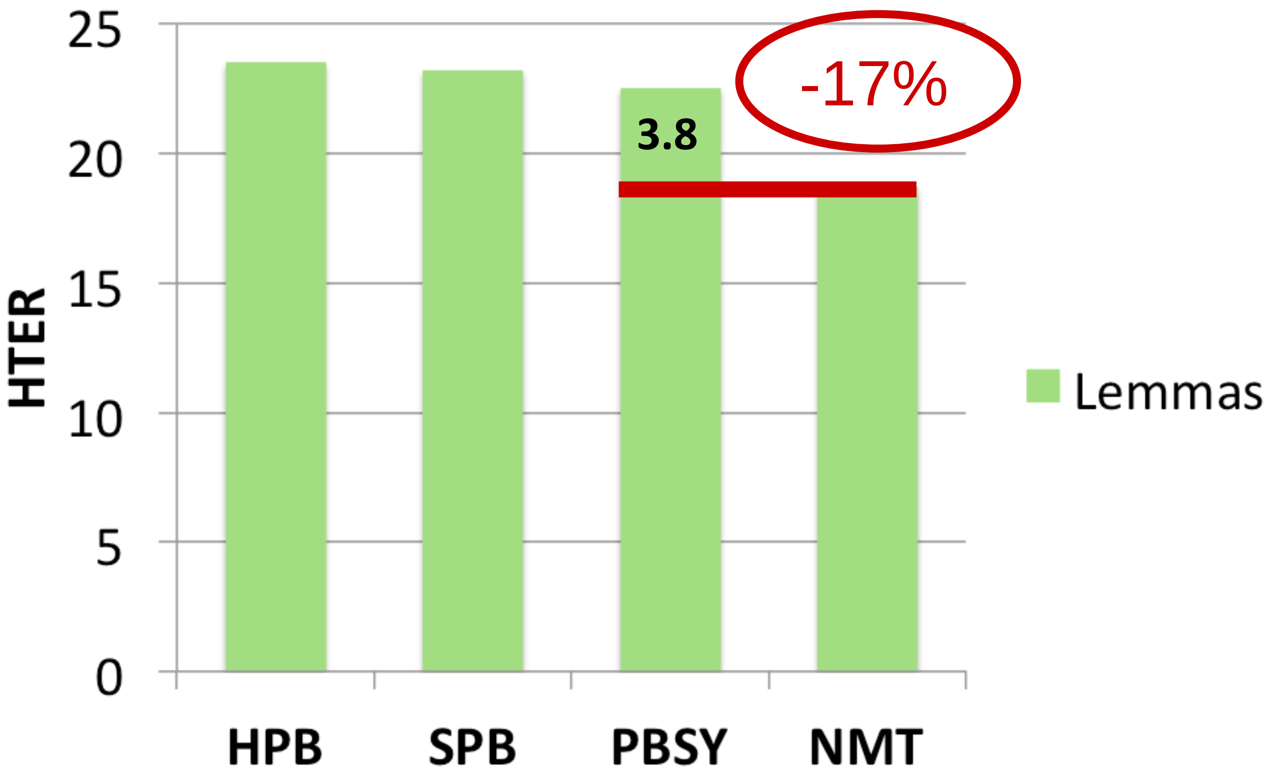
HTER

Focuses on what a human implicitly annotated as a translation error → ***Analyses of errors***

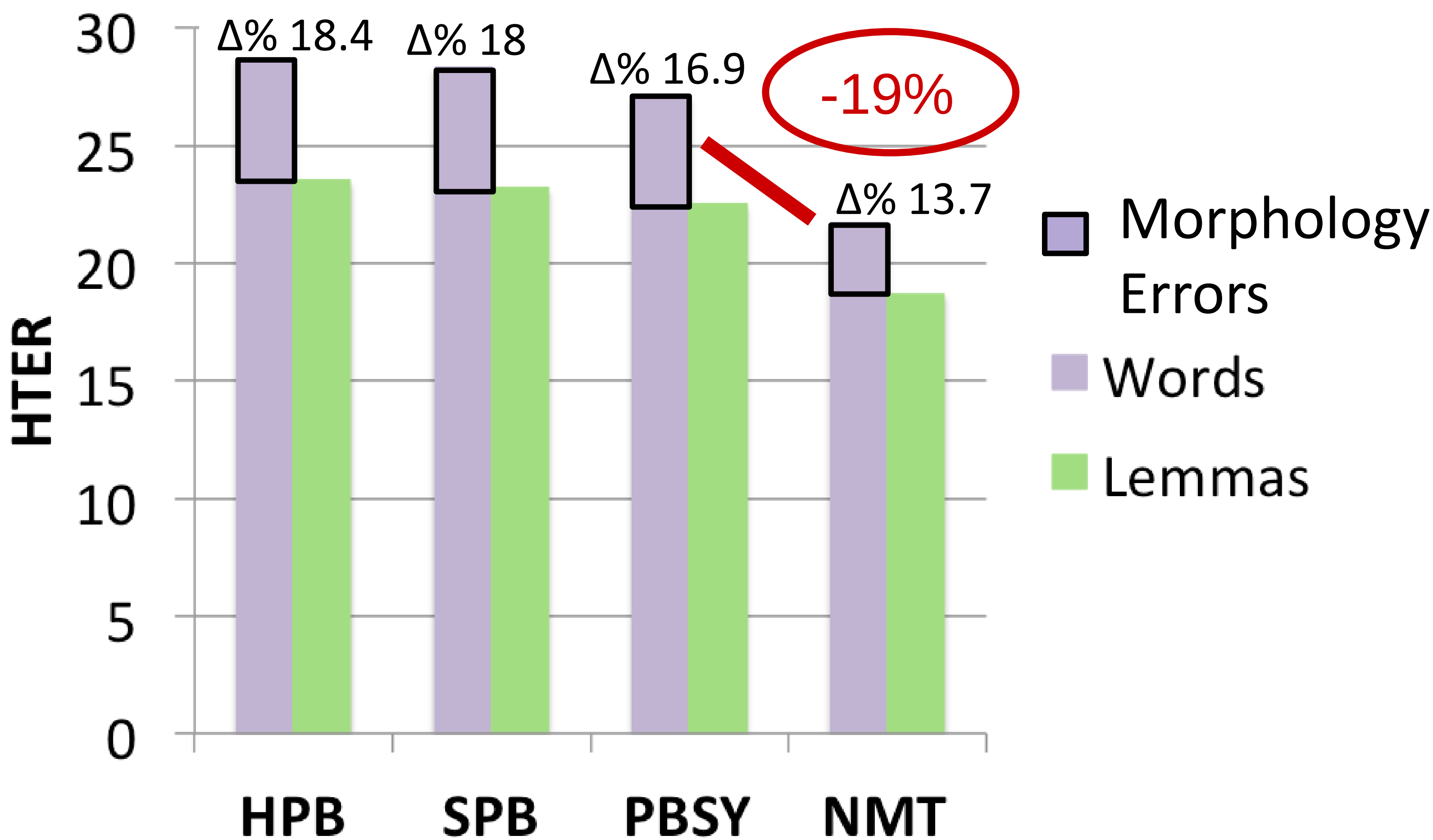
Overall Quality



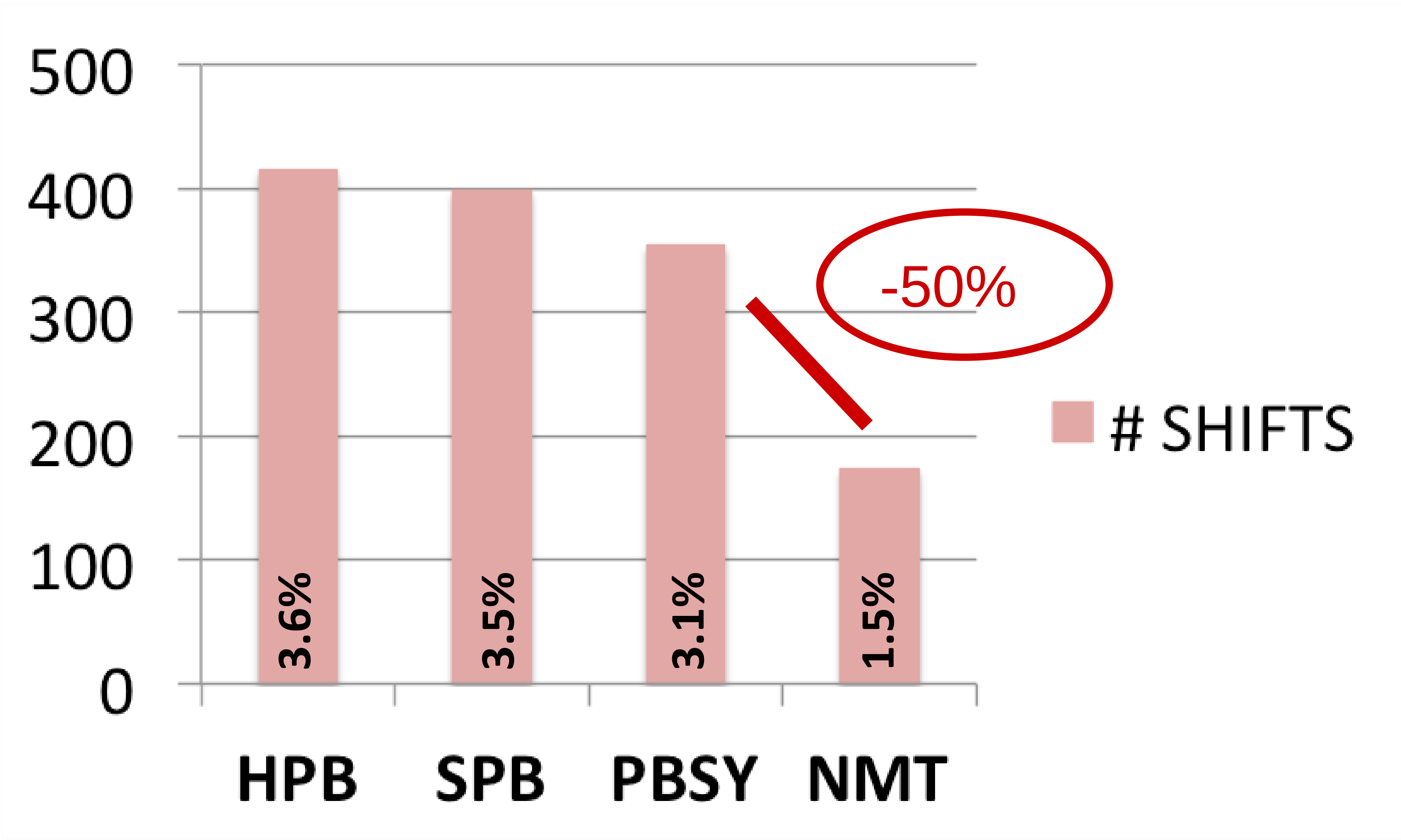
Lexical Errors



Morphology Errors



Reordering Errors



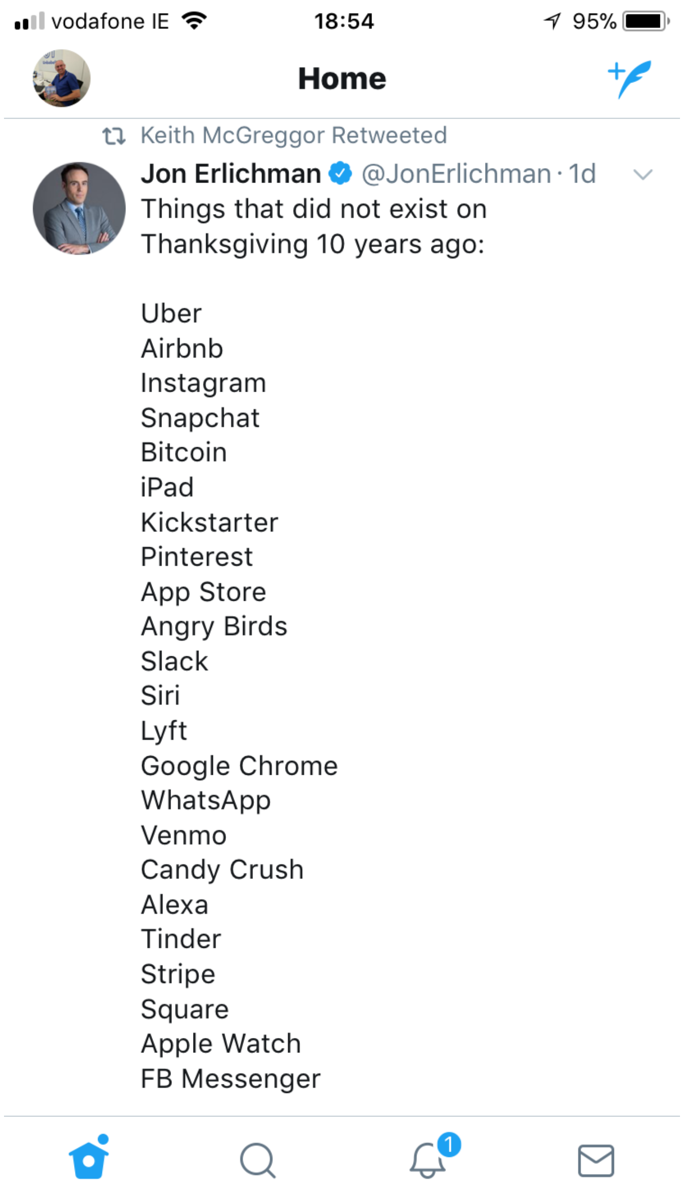
Evaluation Summary

- Generic text features:
  - Input length: degrades faster with long sentences
  - Performs better with lexically rich text
- Linguistic errors:
  - Lexical: -17%, Morphology: -19%, Word Order: -50%
- Word order error types
  - Verb reordering -70%
  - More subtle translation decisions (order of semantic arguments, focus of negation) remain a challenge.

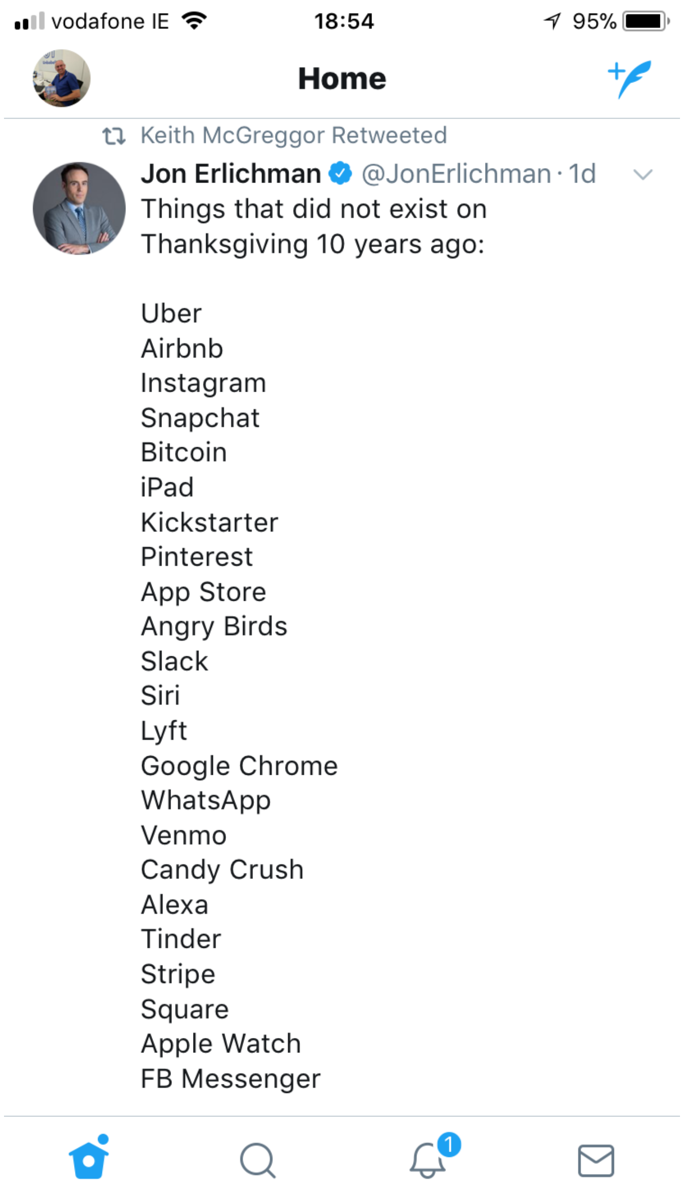
Overview

- Basic building blocks: functions & neurons
- Neural Networks
  - Feed-Forward Neural Networks
  - Inputs, Outputs, Weights, Errors
  - Word Embeddings
  - Recurrent Neural Networks
- Neural Machine Translation: Architecture
  - Encoders
  - Decoders (language models)
  - Attention
- Neural Machine Translation: Improvements
- Future Work in NMT
- Concluding Remarks

Hard to Predict!

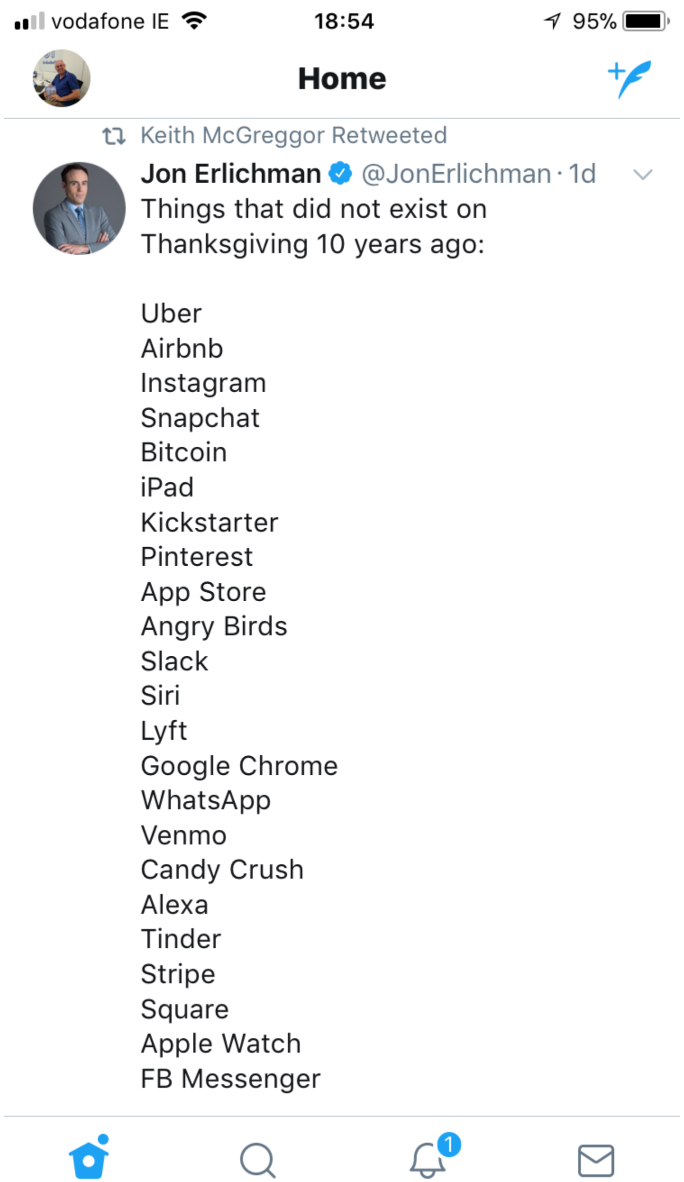


Hard to Predict!



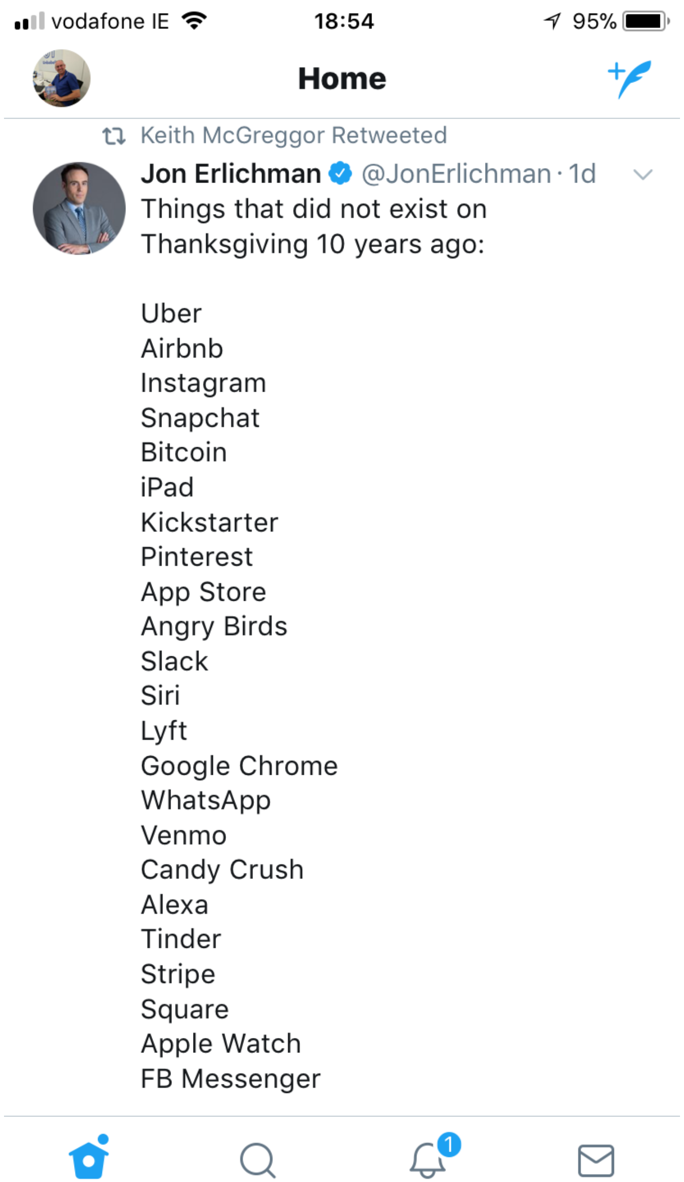
- NMT wasn't *at all* mainstream even four years ago, so it's very hard to predict where we'll be in (say) 3-10 years time …

Hard to Predict!



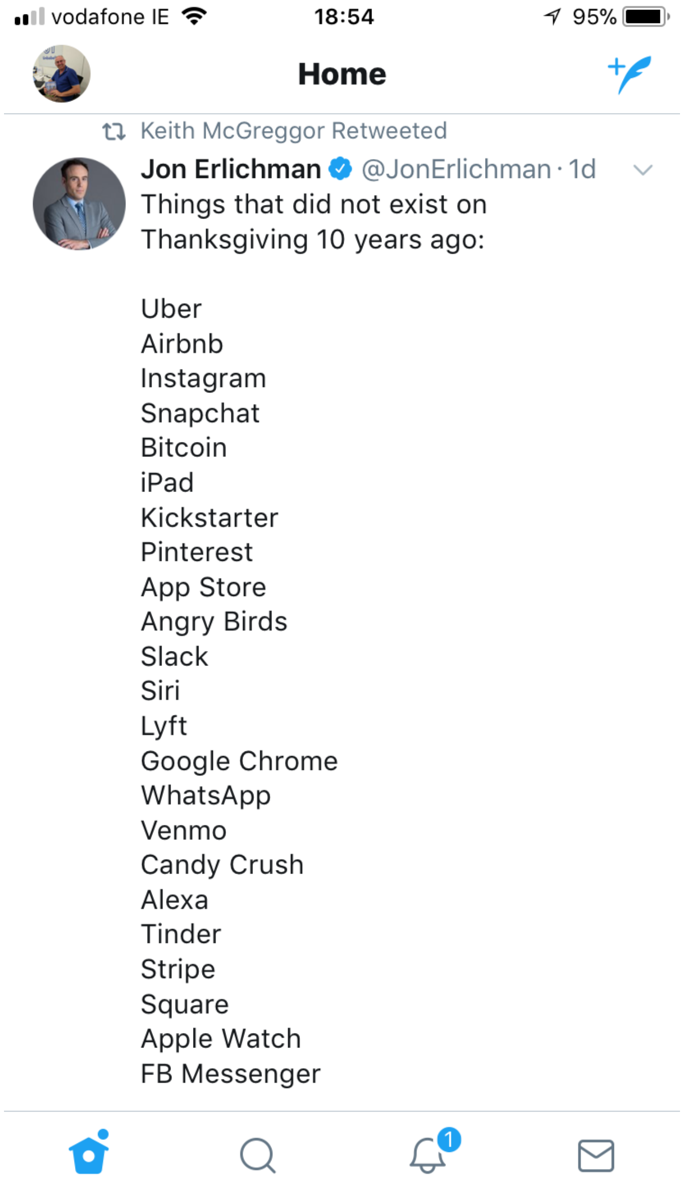
- NMT wasn't *at all* mainstream even four years ago, so it's very hard to predict where we'll be in (say) 3-10 years time …
- Despite claims to the contrary, problems of multilinguality aren’t going away …

Hard to Predict!



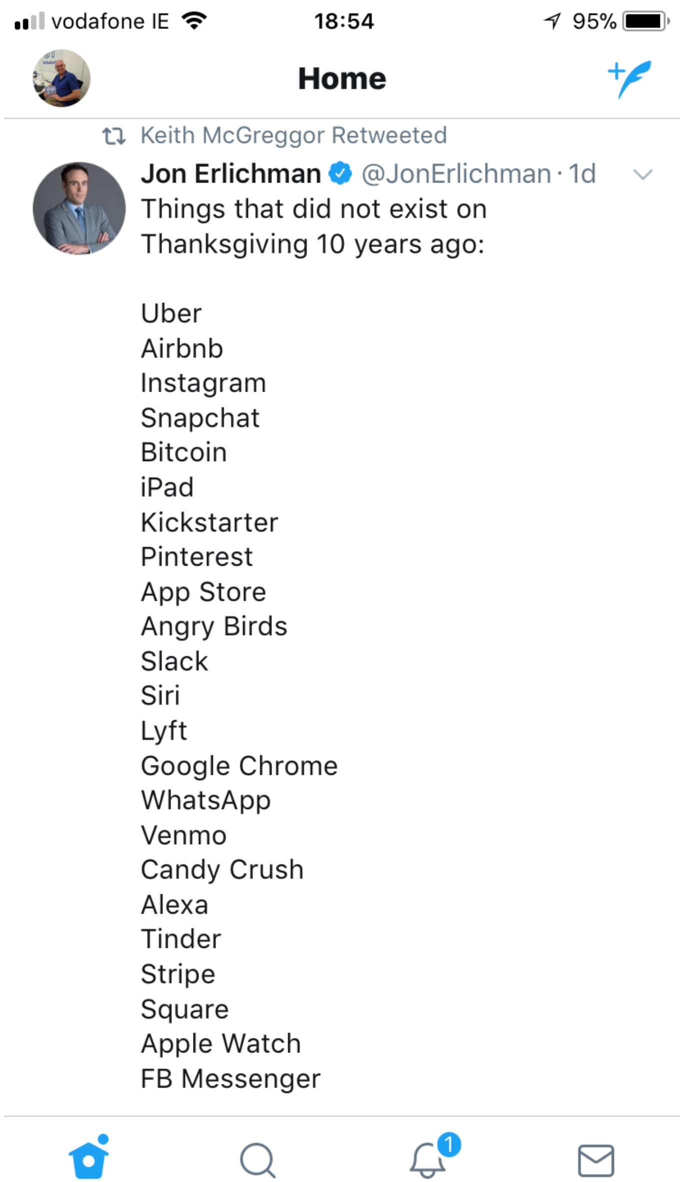
- NMT wasn't *at all* mainstream even four years ago, so it's very hard to predict where we'll be in (say) 3-10 years time …
- Despite claims to the contrary, problems of multilinguality aren't going away …
- Human Factors topics will always be current and topical

Hard to Predict!



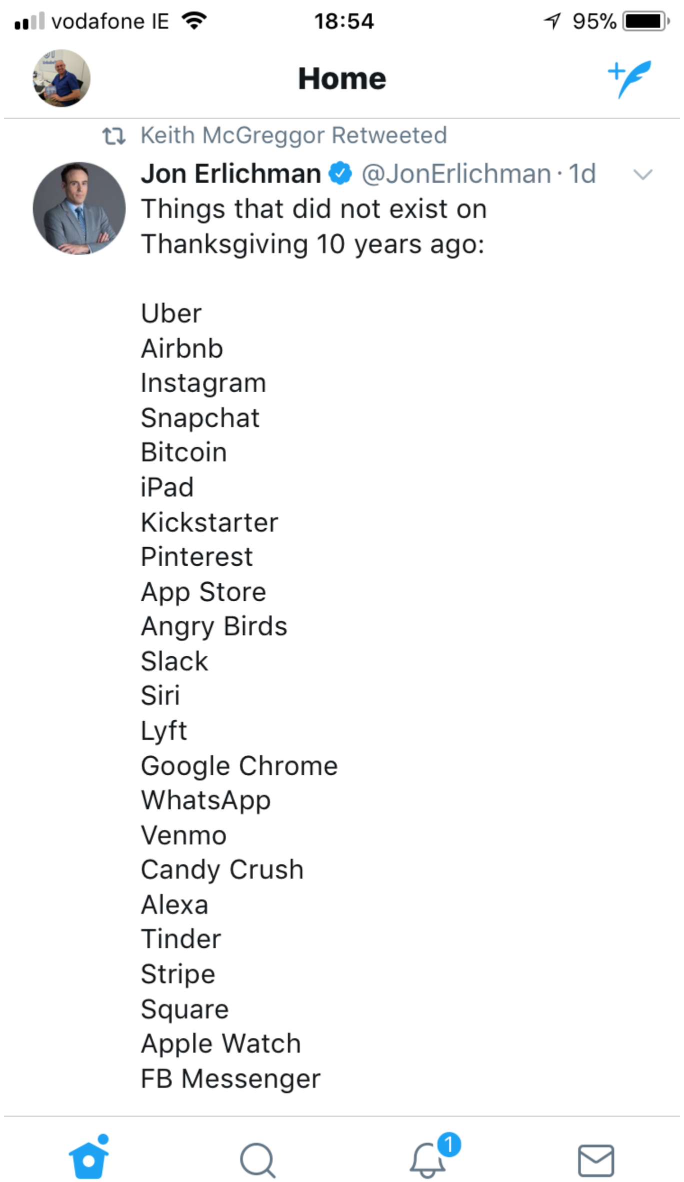
- NMT wasn't *at all* mainstream even four years ago, so it's very hard to predict where we'll be in (say) 3-10 years time …
- Despite claims to the contrary, problems of multilinguality aren't going away …
- Human Factors topics will always be current and topical
- Evaluation remains an unsolved topic

Hard to Predict!



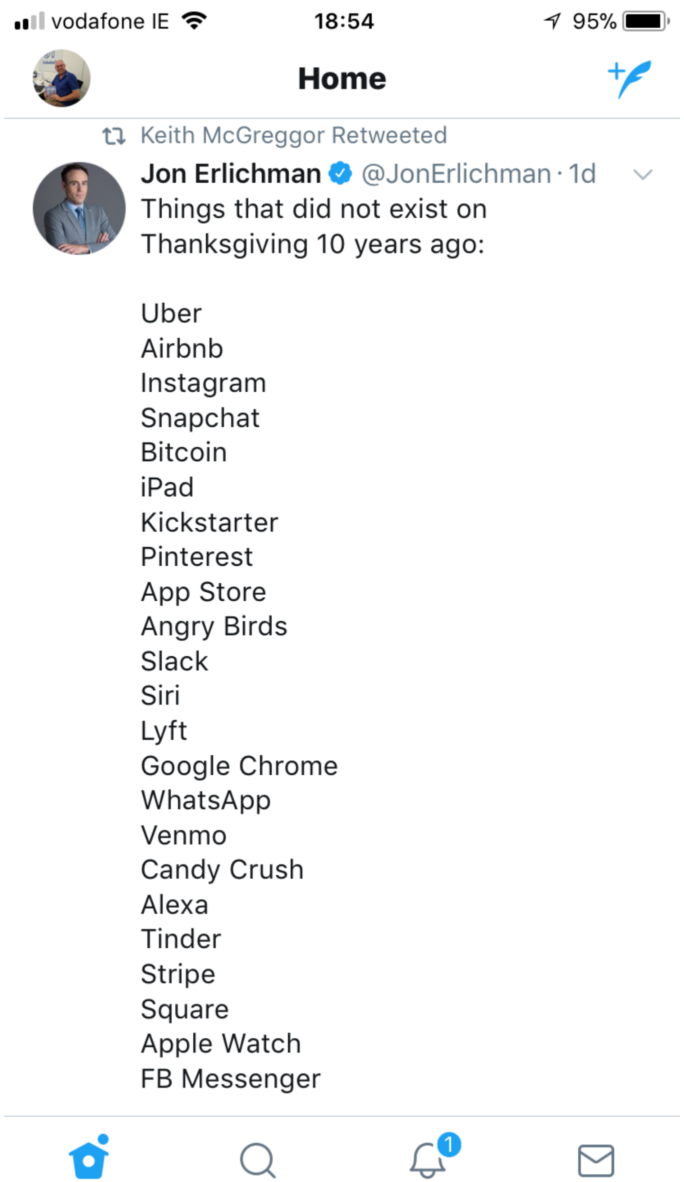
- NMT wasn't *at all* mainstream even four years ago, so it's very hard to predict where we'll be in (say) 3-10 years time …
- Despite claims to the contrary, problems of multilinguality aren’t going away …
- Human Factors topics will always be current and topical
- Evaluation remains an unsolved topic
- Data Selection is arguably a more important topic in NMT than it's been heretofore in SMT

Hard to Predict!



- NMT wasn't *at all* mainstream even four years ago, so it's very hard to predict where we'll be in (say) 3-10 years time …
- Despite claims to the contrary, problems of multilinguality aren't going away …
- Human Factors topics will always be current and topical
- Evaluation remains an unsolved topic
- Data Selection is arguably a more important topic in NMT than it's been heretofore in SMT
- User-Generated Content will continue to increase

Hard to Predict!



- NMT wasn't *at all* mainstream even four years ago, so it's very hard to predict where we'll be in (say) 3-10 years time …
- Despite claims to the contrary, problems of multilinguality aren't going away …
- Human Factors topics will always be current and topical
- Evaluation remains an unsolved topic
- Data Selection is arguably a more important topic in NMT than it's been heretofore in SMT
- User-Generated Content will continue to increase
- Engine training times need to come down, so parallelization might be an important topic

Symbiotic Translation

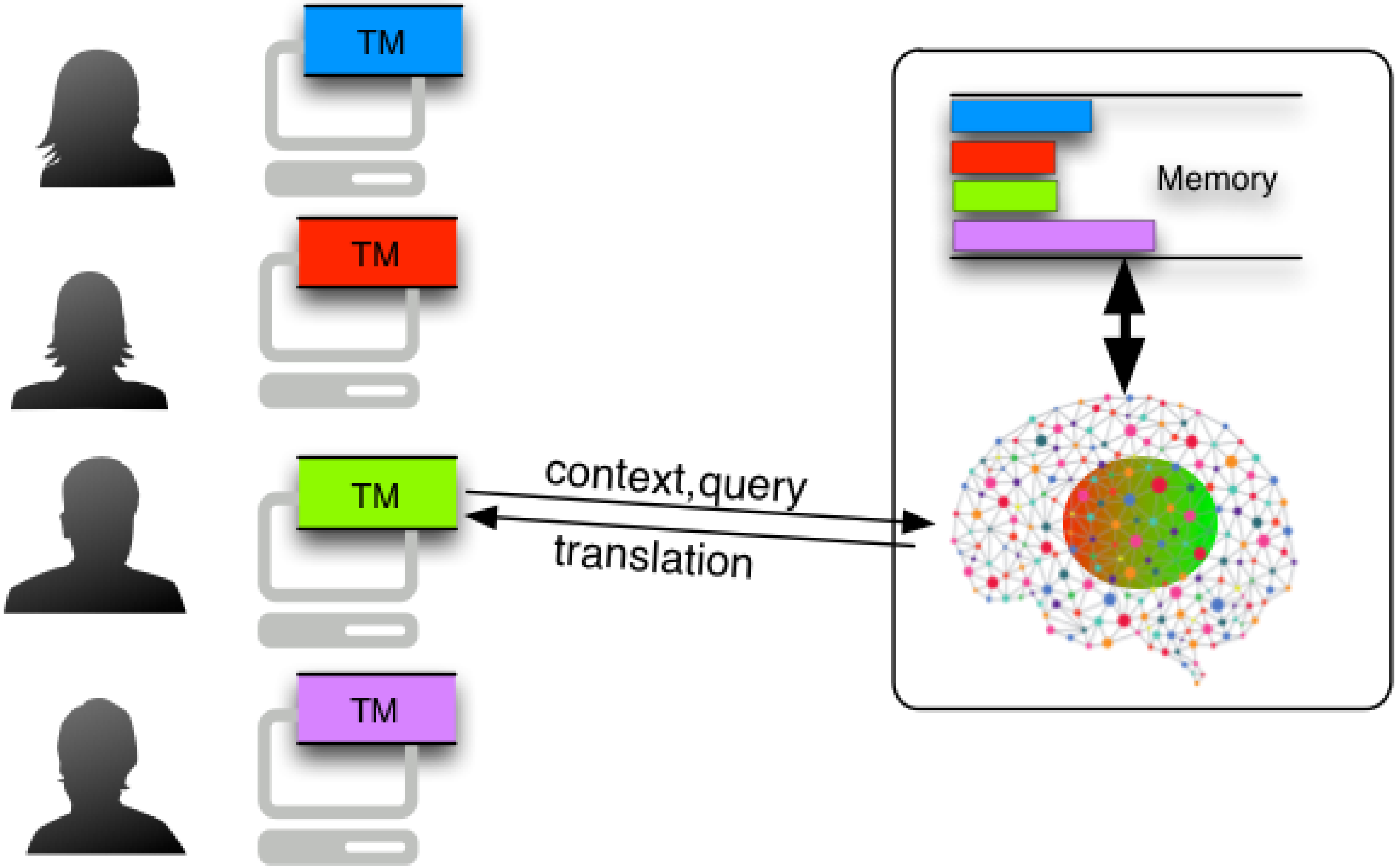
MT seamlessly

- adapts to user data
- learns from post-editing

User enjoys

- enhanced productivity
- better user experience

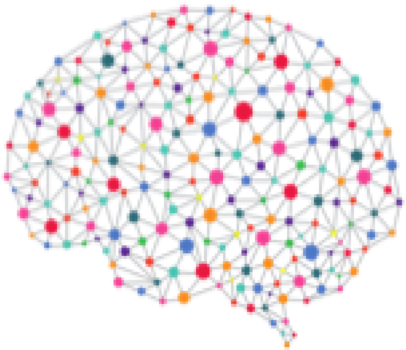
Adaptive Neural MT



Neural Automatic Post-Editing

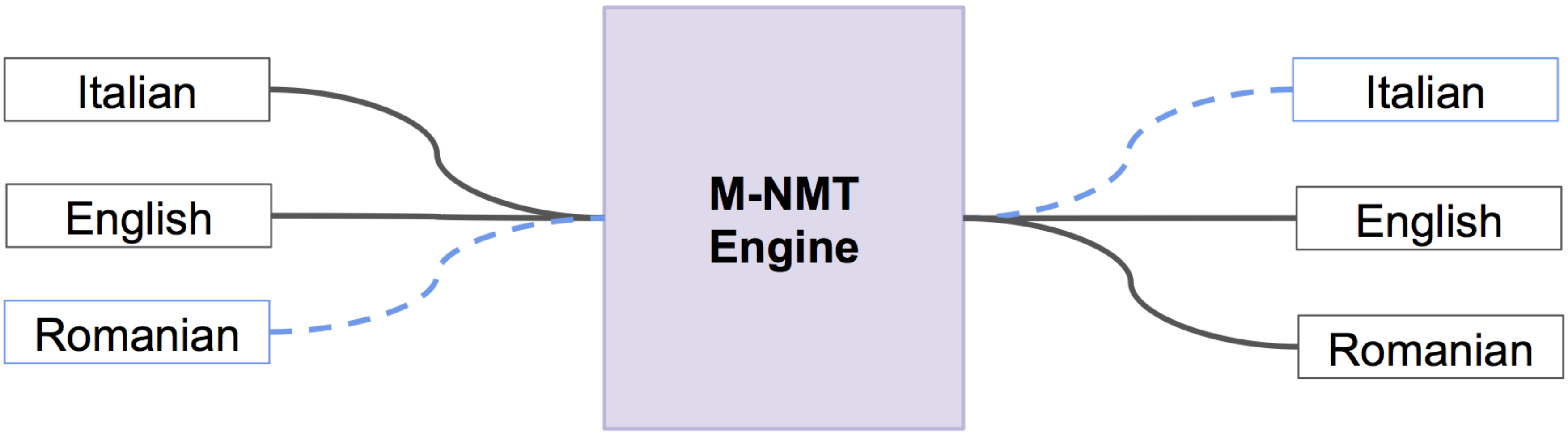
Police have clashed with opposition supporters, some of whom have been blocking access to polling stations.

La polizia si sono scontrati con i sostenitori dell'opposizione, alcuni dei quali hanno bloccato l'accesso ai seggi elettorali.



La polizia si è scontrata con i sostenitori dell'opposizione, alcuni dei quali hanno bloccato l'accesso ai seggi elettorali.

Multilingual MT



Overview

- Basic building blocks: functions & neurons
- Neural Networks
  – Feed-Forward Neural Networks
  – Inputs, Outputs, Weights, Errors
  – Word Embeddings
  – Recurrent Neural Networks
- Neural Machine Translation: Architecture
  – Encoders
  – Decoders (language models)
  – Attention
- Neural Machine Translation: Improvements
- Future Work in NMT
- Concluding Remarks

Engaging Content
Engaging People

Concluding Remarks

- Neural MT has rapidly kick-started a new era
  - Significant improvement in performance
  - But exaggerated and hysterical claims
- We are living in *turbulent times*:
  - Novel ideas coming out almost every week!
  - DL is offering new powerful tools
  - It will take time to maximise their full potential
- MT remains a very hard problem to tackle!

Other Applications

MT vs. Self-Driving Cars

Technical
translation

MT vs. Self-Driving Cars

Technical
translation

Creative
translation

MT vs. Self-Driving Cars

Technical
translation

Creative
translation

Difficult
translation

Thanks for listening!

Engaging Content
Engaging People

References

[Bahdanau et al., 2015] Neural Translation by Jointly Learning to Align and Translate. arXiv.
[Cho et al., 2014a] Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. EMNLP.
[Bentivogli et al, 2016] Neural versus. Phrase-Based Machine Translation Quality: a Case Study. EMNLP.
[Farajian et al., 2016] Neural vs. Phrase-Based Machine Translation in a Multi-Domain Scenario. EACL.
[Farajian et al., 2017] . Multi-Domain Neural Machine Translation through Unsupervised Adaptation. WMT.
[Sennrich al., 2016] Neural Machine Translation of Rare Words with Subword Units, ACL.
[Ataman et al. 2017] Linguistically Motivated Vocabulary Reduction for Neural Machine Translation from Turkish to English, EAMT.
[Wu et al. 2016] Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv
[Gehring et al., 2017] Convolutional Sequence to Sequence Learning. arXiv.
[Barone et al. 2017] Deep Architectures for Neural Machine Translation, WMT.
[Ranzato et al., 2016] Sequence level training with recurrent neural networks. arXiv.