
Seq2seq Models with Attention

Thang Vu

08.01.2026



Outline

- Recap
- Introduction
- Encoder-decoder Framework
- Attention Mechanism

The Start

- Basics in math for machine learning
- Basics in machine learning
- Basics in PyTorch

The Start

- Basics in math for machine learning
 - Linear algebra: vectors & matrices
 - Calculus: derivatives
 - Probabilities & optimization
- Basics in machine learning
- Basics in PyTorch

The Start

- Basics in math for machine learning
- Basics in machine learning
 - Unsupervised learning
 - Supervised learning
 - Features
- Basics in PyTorch

The Start

- Basics in math for machine learning
- Basics in machine learning
- Basics in PyTorch
 - Tensors
 - Data processing

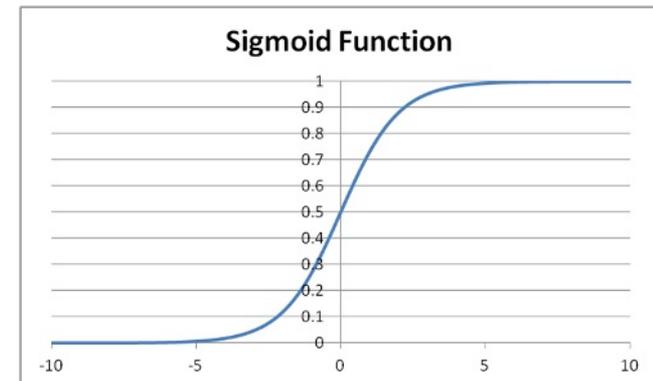
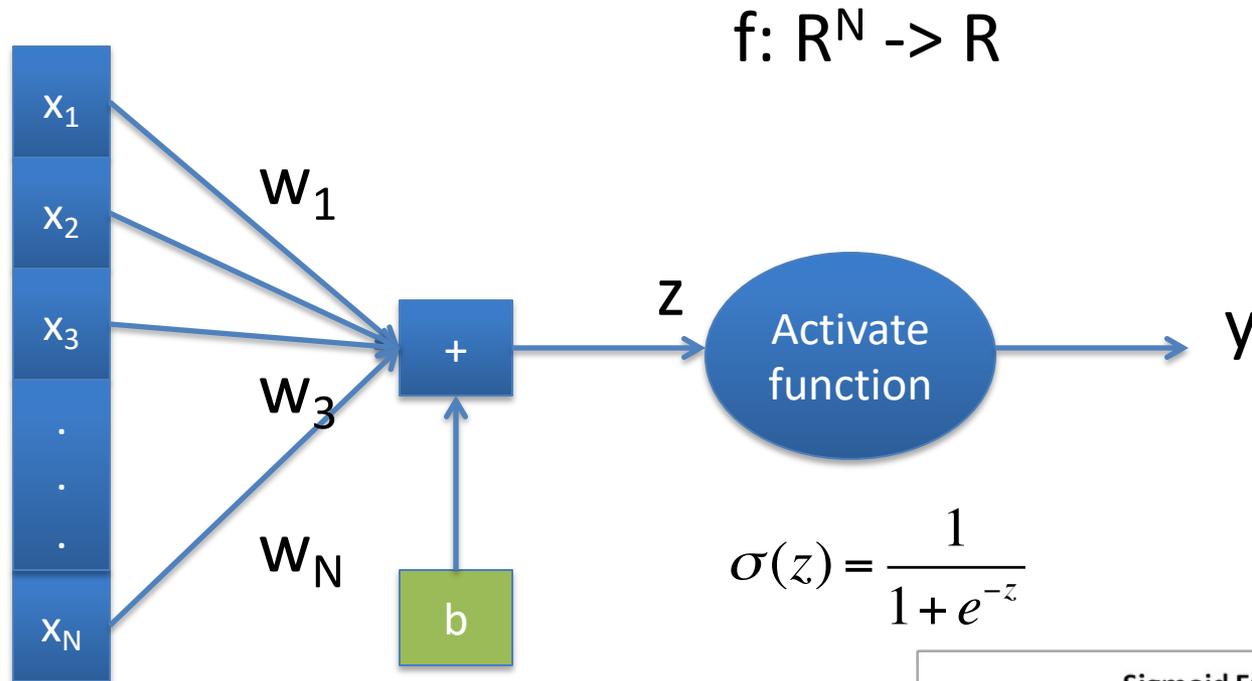
The Core

- Architectures
- Learning
- Tricks

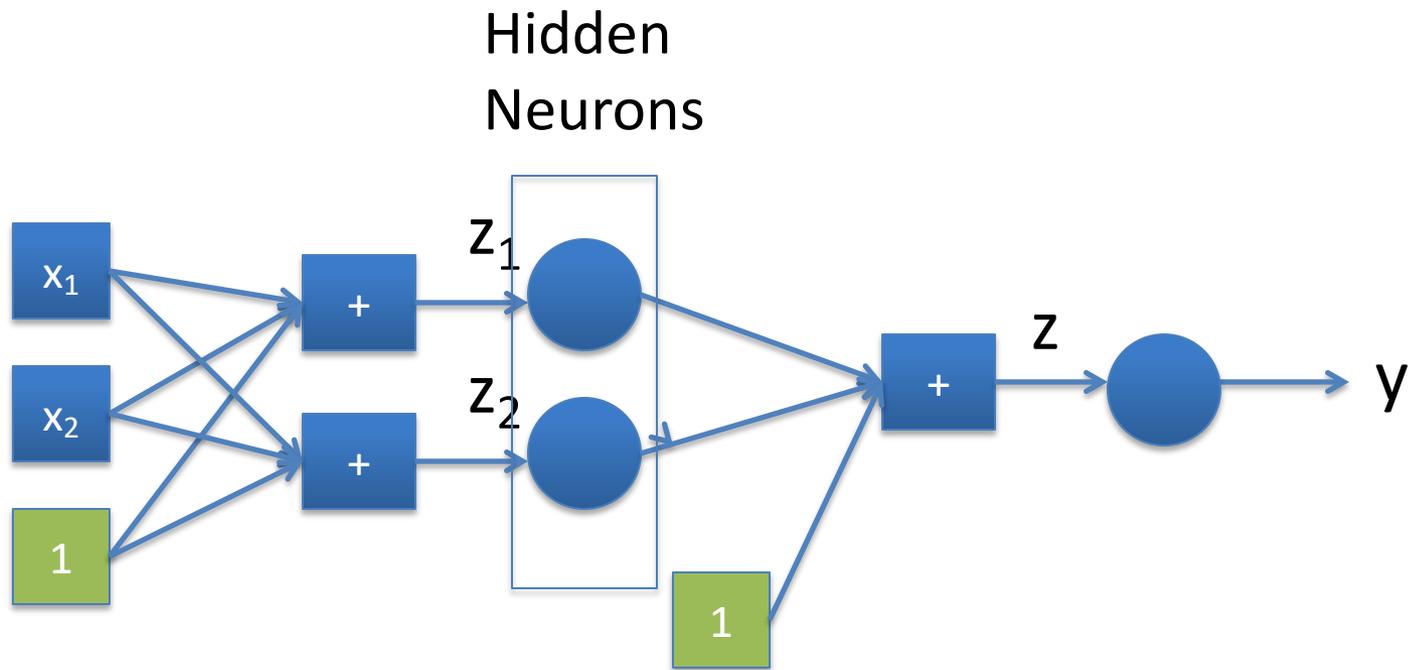
The Core

- Architectures
- Learning
- Tricks

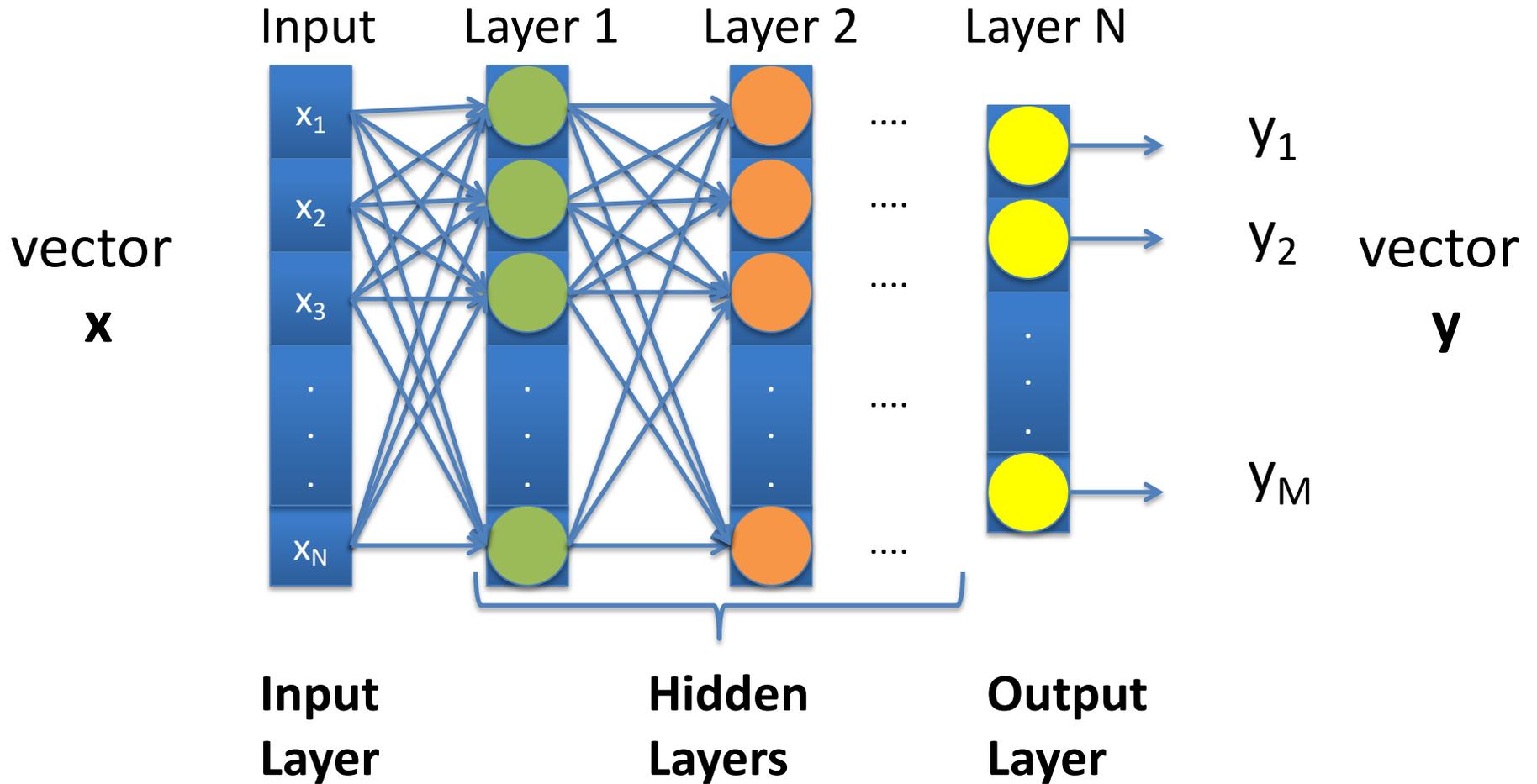
Model: A single neuron



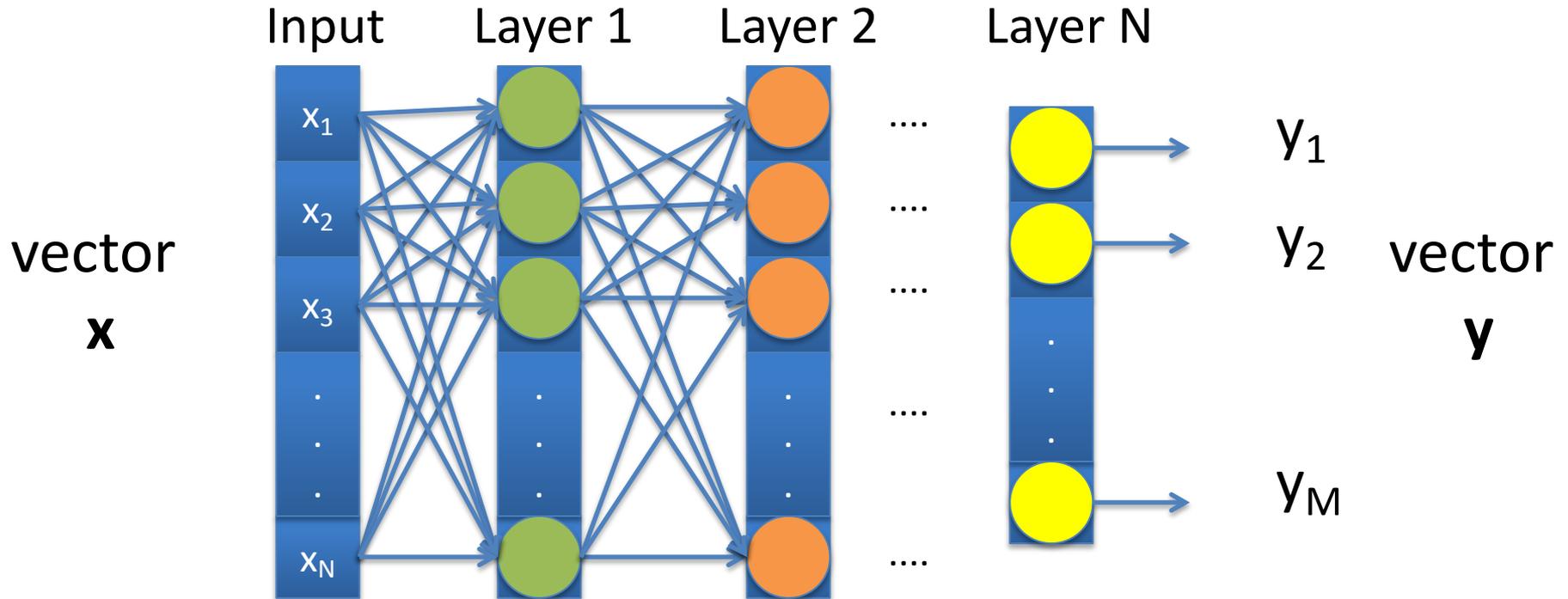
Neural network



Neural network with hidden layers



Feed Forward Neural Nets



$$y = f(x) = \sigma(W^L \dots \sigma(W^2 \sigma(W^1 x + b^1) + b^2) \dots + b^L)$$

Convolutional Neural Networks

- What if the input is a matrix?

0	1	0	0	1
2	3	3	3	4
4	0	1	0	1
3	4	2	3	3

Convolutional Neural Networks

- What if the input is a matrix?

0	1	0	0	1
2	3	3	3	4
4	0	1	0	1
3	4	2	3	3



0
2
4
3
1
3
0
4

...

Problem 1

- Problem 1: What if the input is a matrix?

0	1	0	0	1
2	3	3	3	4
4	0	1	0	1
3	4	2	3	3



0
2
4
3
1
3
0
4

...

- #parameters ↗

Problem 1

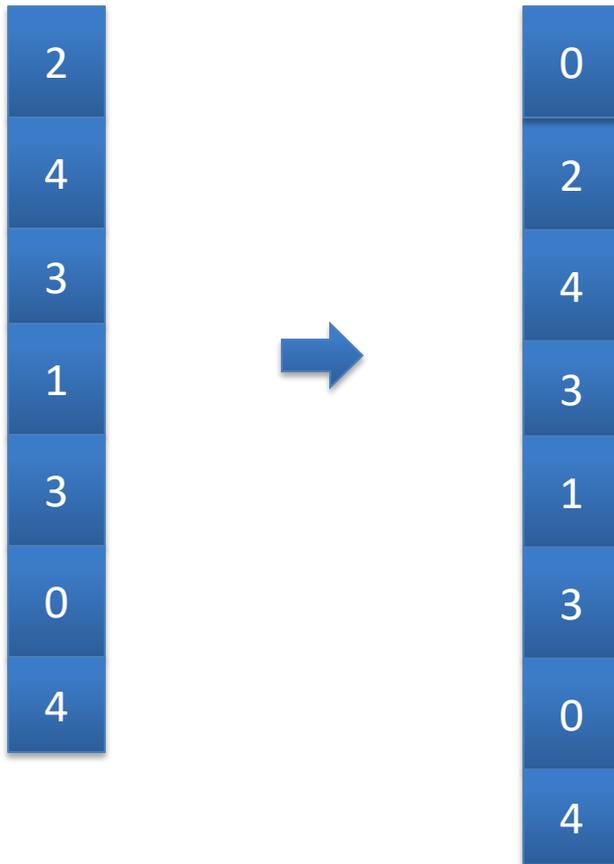
- Problem 1: What if the input is a matrix?

0	1	0	0	1
2	3	3	3	4
4	0	1	0	1
3	4	2	3	3

- Don't want to blow up the number of parameters
 - If possible, reduce the number of parameters

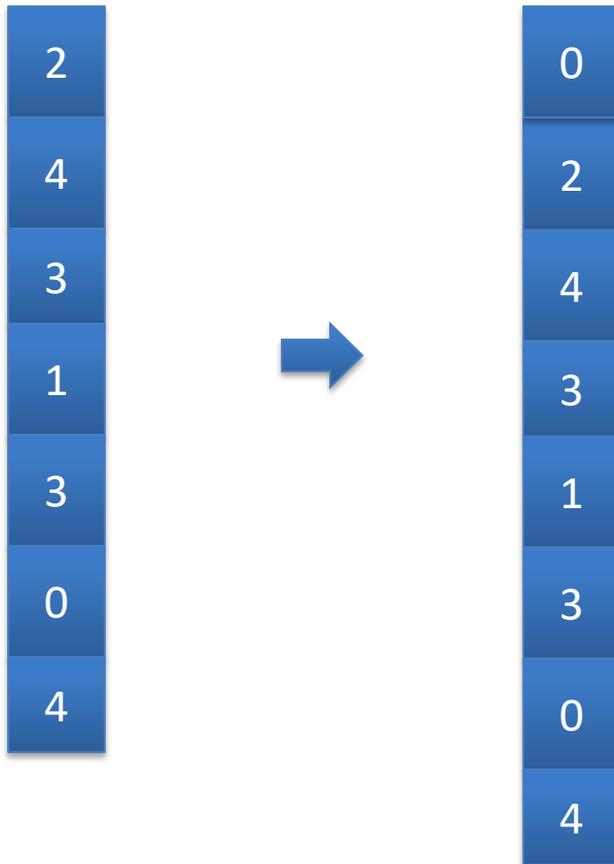
Problem 2

- Problem 2: What if the input is shifted?



Problem 2

- Problem 2: What if the input is shifted?

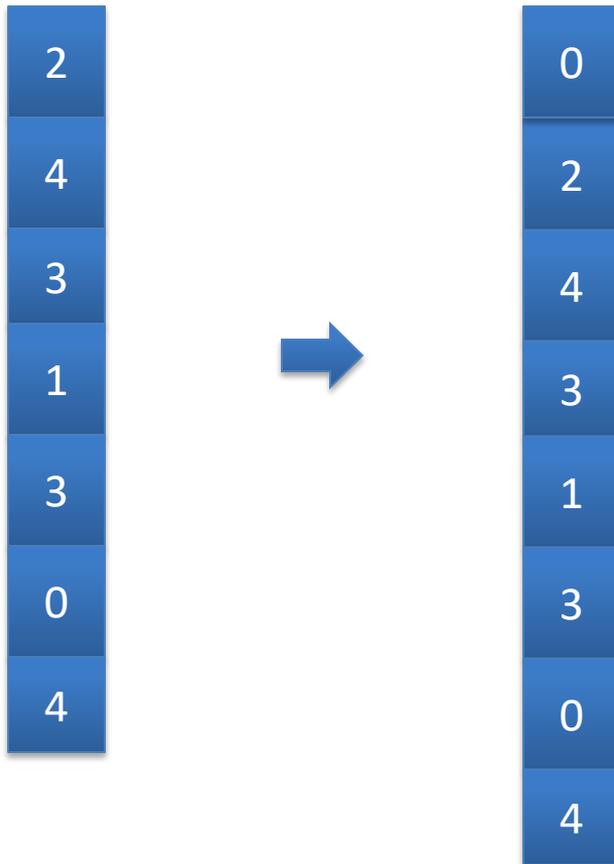


Real-world example 1:

- Language processing:
 - Input 1: „I am happy today“
 - Input 2: „Today I am happy“

Problem 2

- Problem 2: What if the input is shifted?

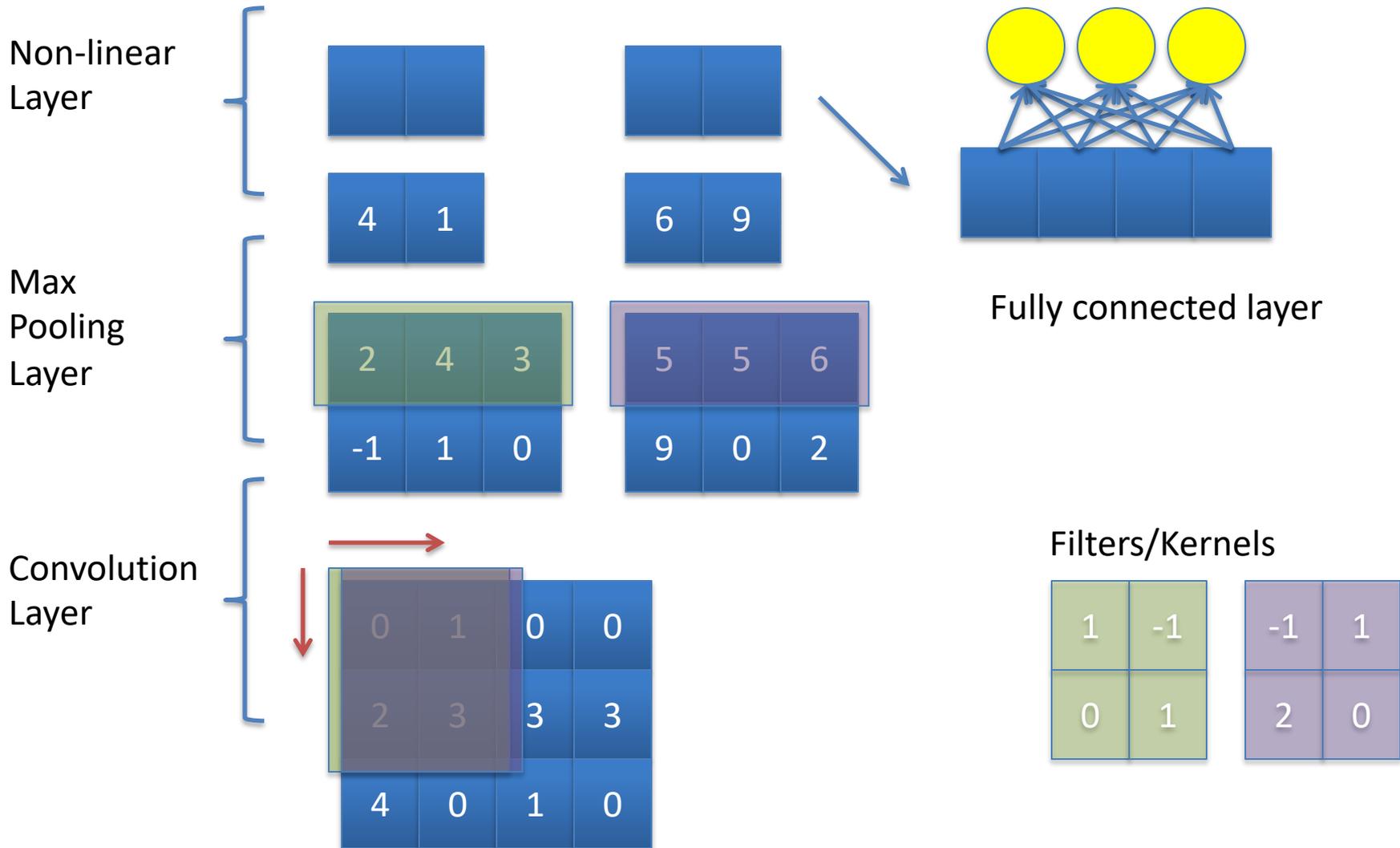


Real-world example 2:

- Computer vision:



Convolutional Neural Nets



Recurrent neural nets (RNNs)

- Input data: $[x_1, x_2, x_3, \dots, x_n]$
 - Sequence of data points
 - Prediction at each time step is influenced by previous time steps

Recurrent neural nets (RNNs)

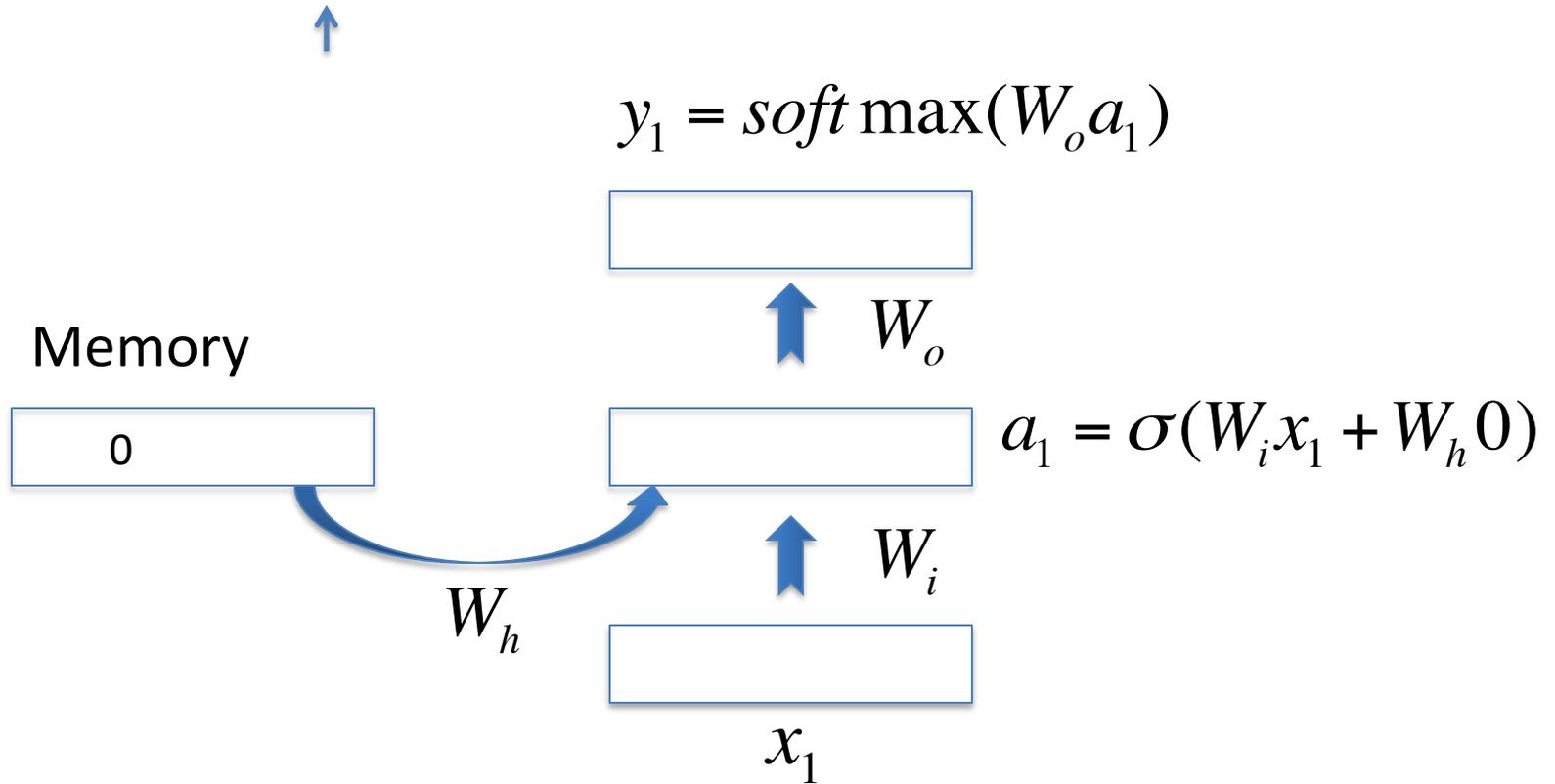
- Input data: $[x_1, x_2, x_3, \dots, x_n]$
 - e.g. language modeling task, i.e. predict the next word given the history
 - e.g. I apologize for being late, I am very _____

Recurrent neural nets (RNNs)

- Input data: $[x_1, x_2, x_3, \dots, x_n]$
 - Information which is observed from $[x_1, x_2, \dots, x_{t-1}]$ should be stored in memory
 - Then used for the prediction at the time step t

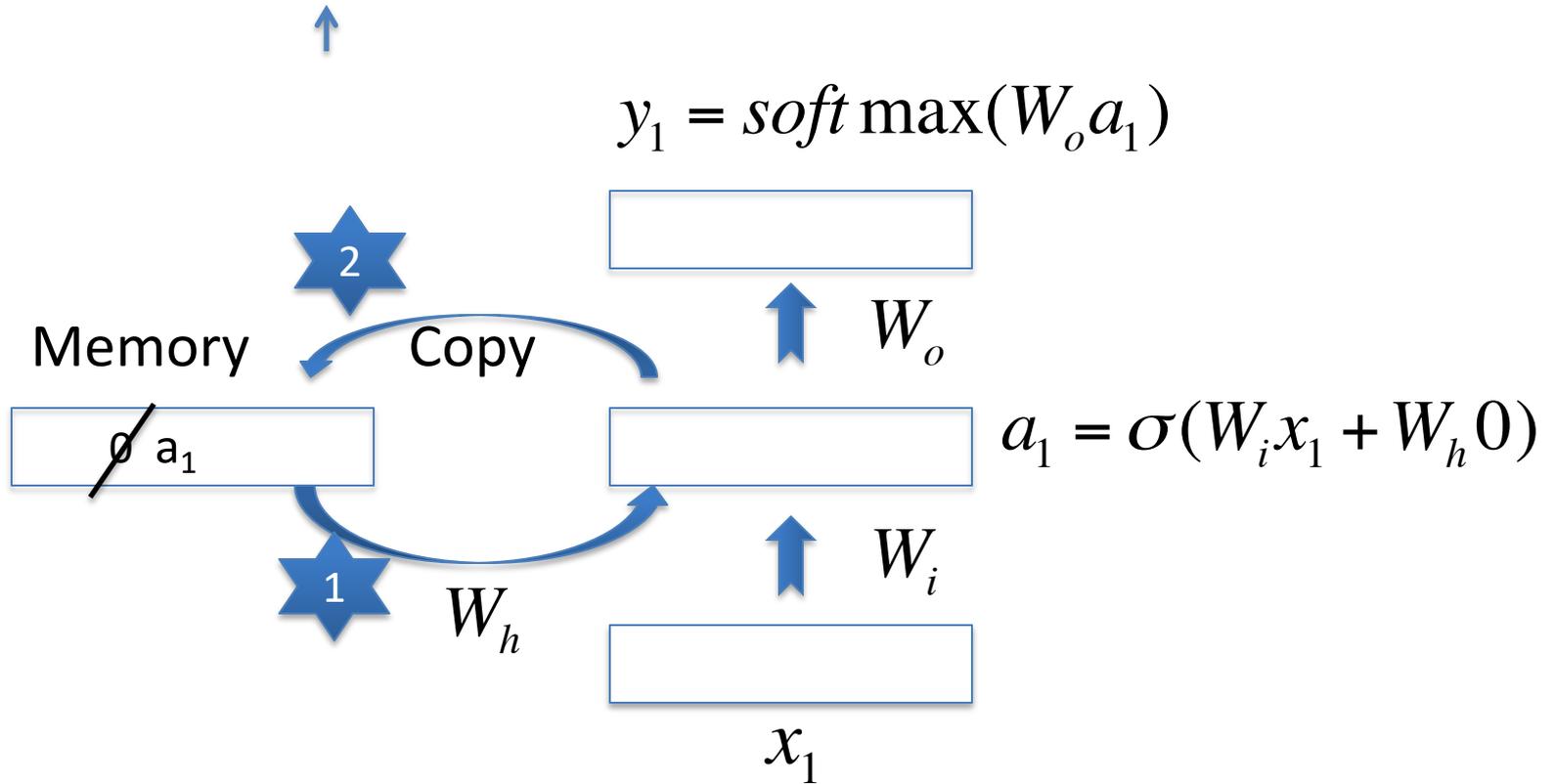
RNN

- Input data: $[x_1, x_2, x_3, \dots, x_n]$



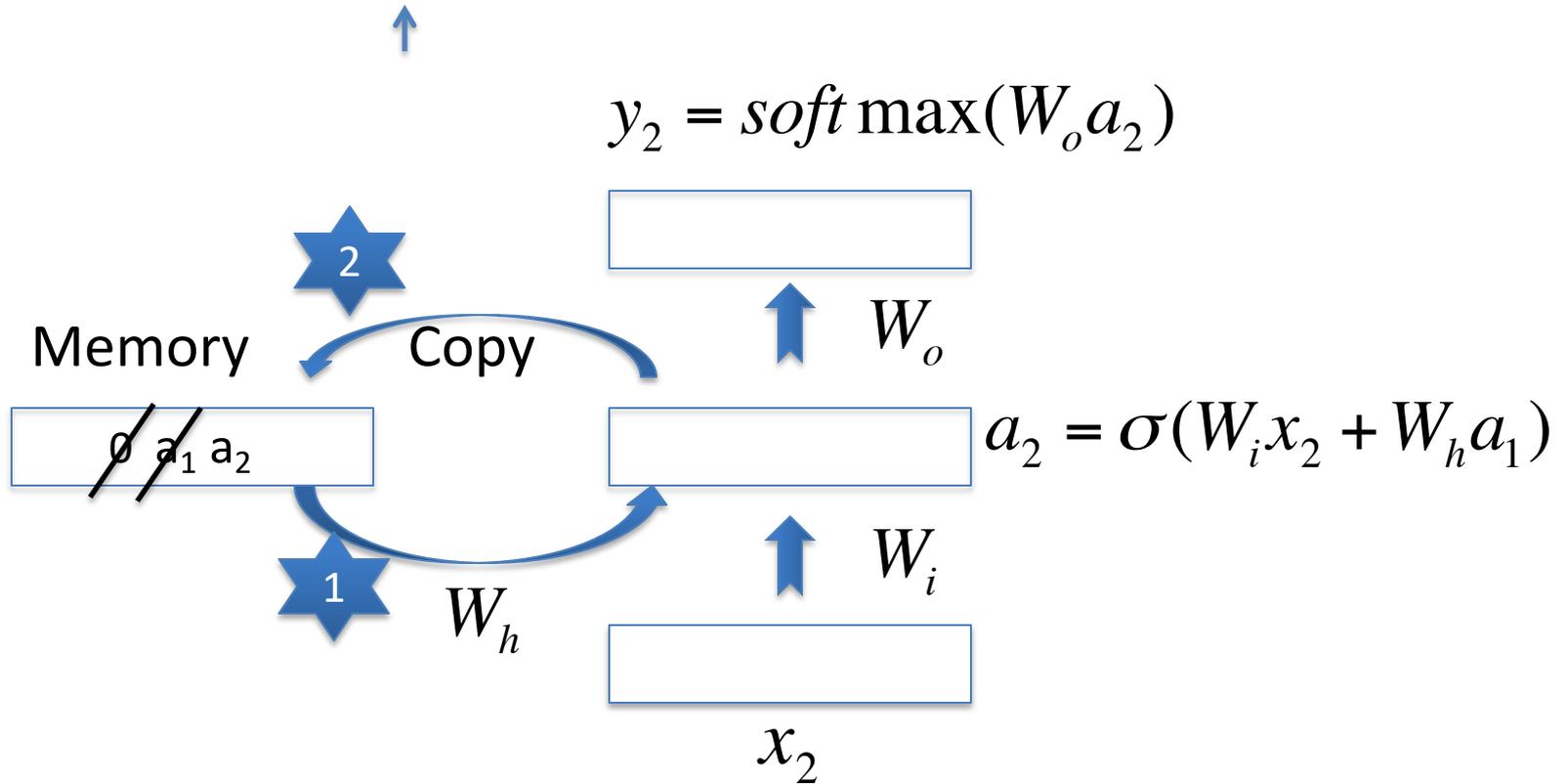
RNN

- Input data: $[x_1, x_2, x_3, \dots, x_n]$



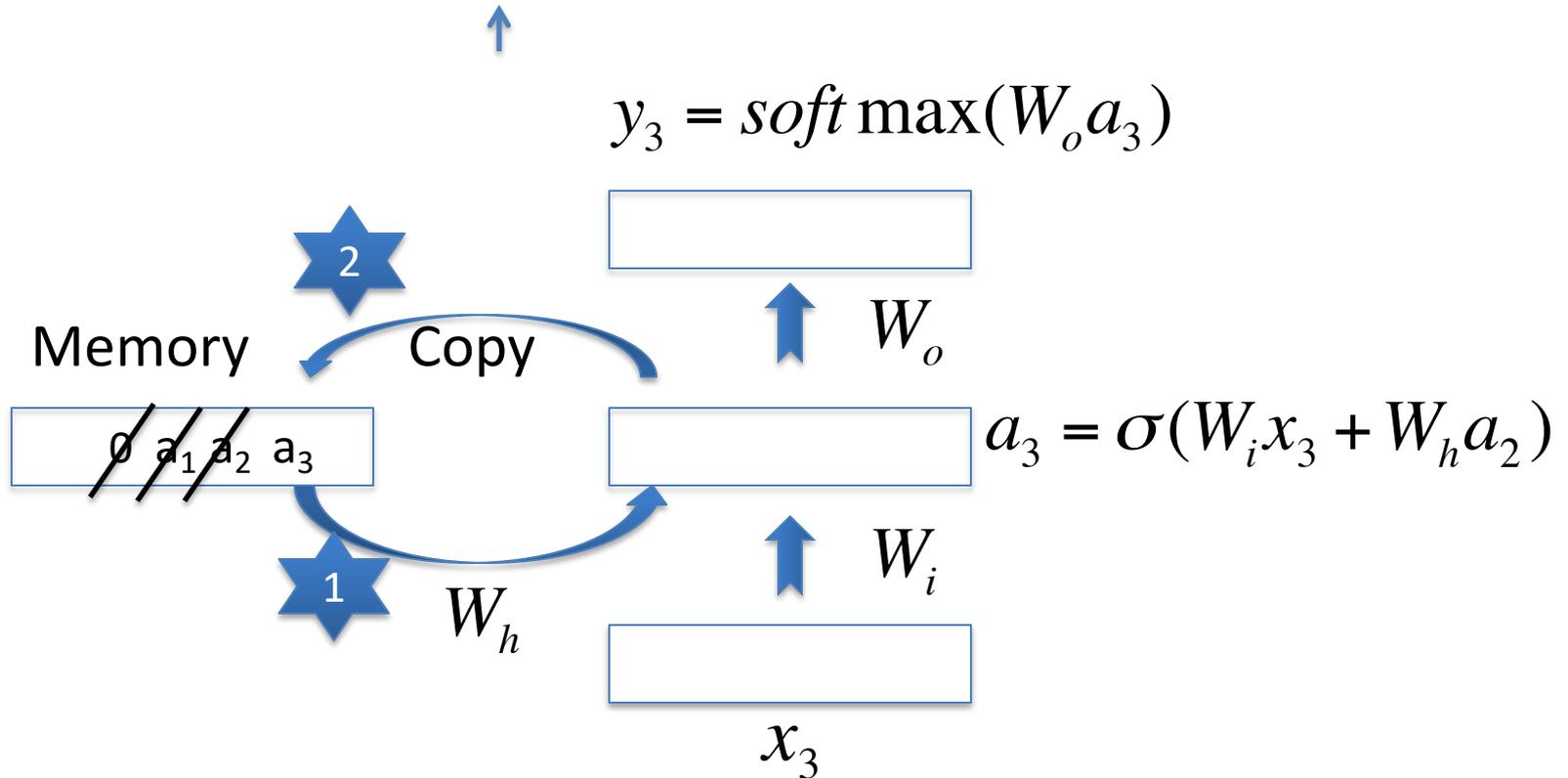
RNN

- Input data: $[x_1, x_2, x_3, \dots, x_n]$

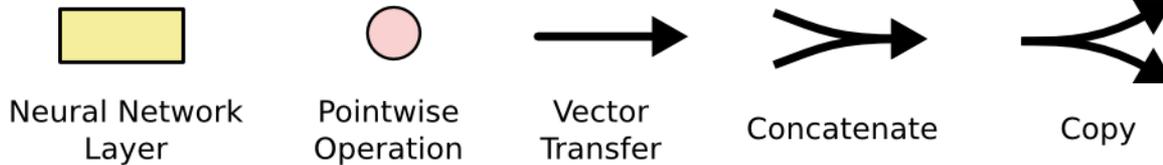
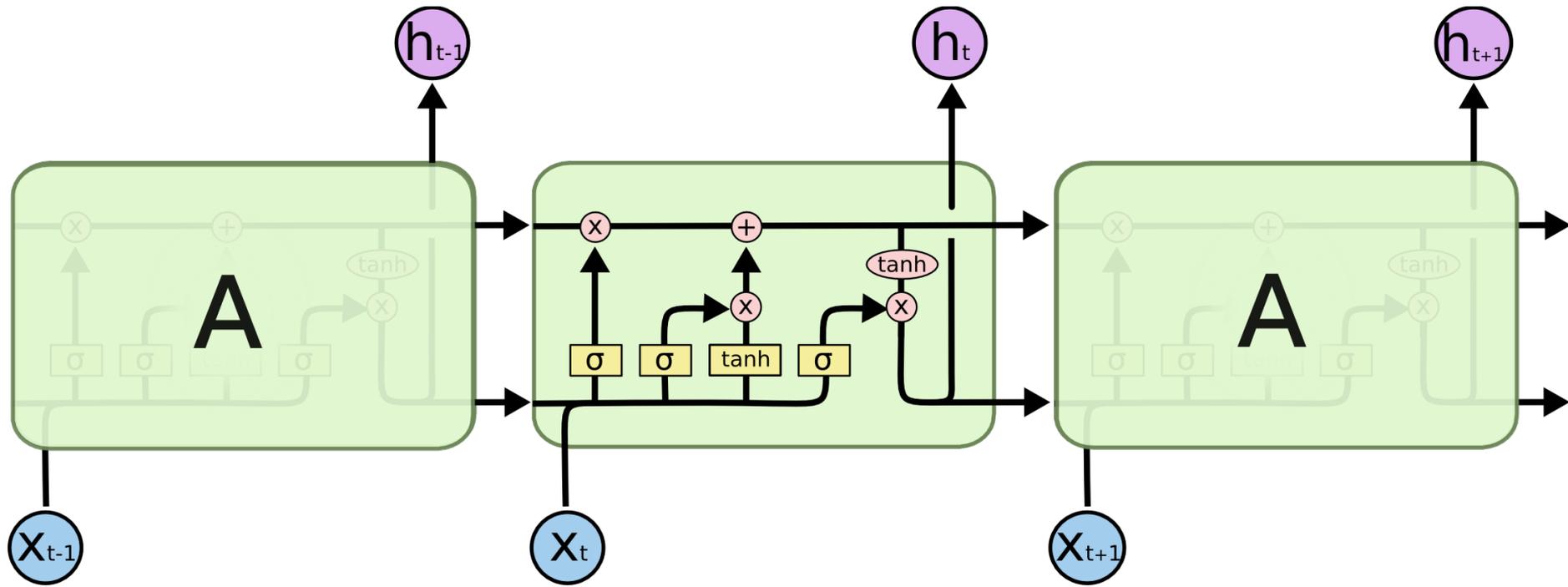


RNN

- Input data: $[x_1, x_2, x_3, \dots, x_n]$



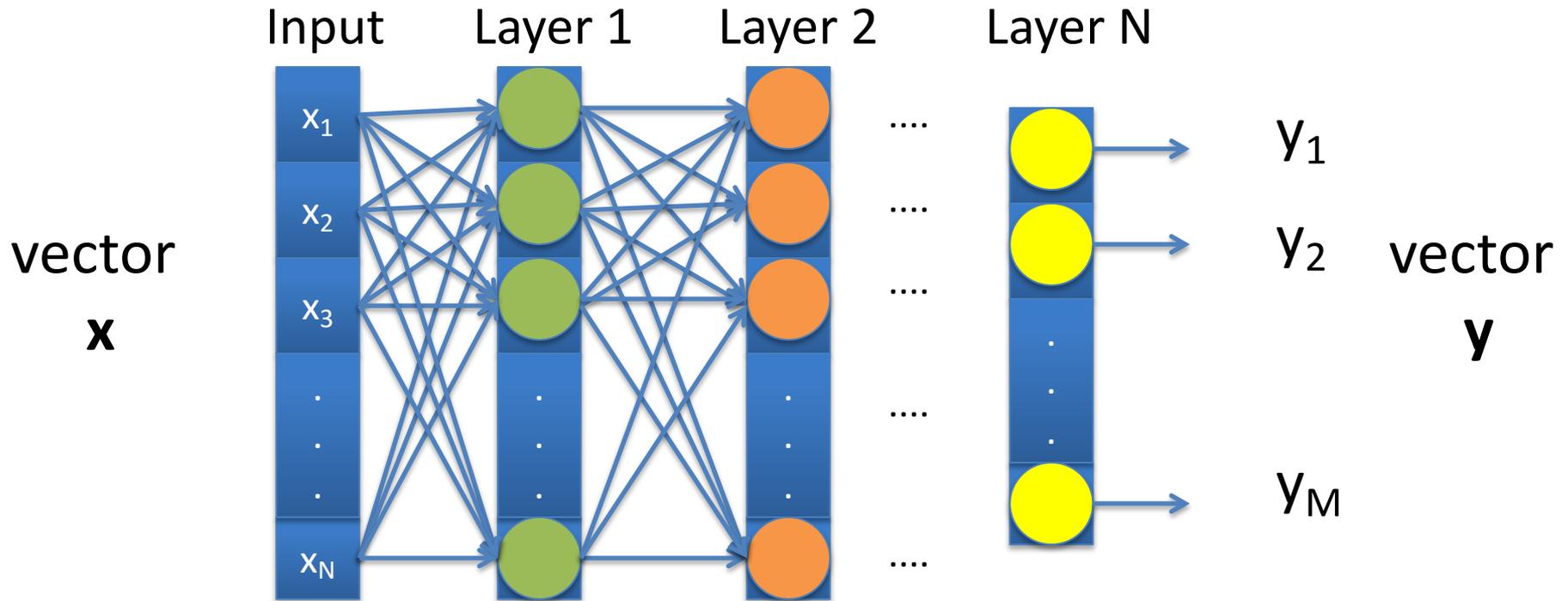
Long Short-term Memory (LSTM)



The Core

- Architectures
- **Learning**
- Tricks

Computation of the final output



$$y = f(x) = \sigma(W^L \dots \sigma(W^2 \sigma(W^1 x + b^1) + b^2) \dots + b^L)$$

Which function is the best function?

- Best function = best parameters

$$y = f(x) = \sigma(W^L \dots \sigma(W^2 \sigma(W^1 x + b^1) + b^2) \dots + b^L)$$

- The function itself depends on the parameter set

$$f(x) = f(x, \theta)$$

$$\theta = \{W^1, b^1, W^2, b^2, \dots, W^L, b^L\}$$

- Search the best function f^*

 Search the best parameter set θ^*

Loss function

- Define a function for parameter set $C(\theta)$
 - C evaluate how bad a parameter set is
 - The best parameter is the one that minimizes $C(\theta)$

$$\theta^* = \underset{\theta}{\operatorname{argmin}} C(\theta)$$

- $C(\theta)$ is called loss function
- If you define the goodness of the parameter set by function $O(\theta)$
 - $O(\theta)$ is called then objective function

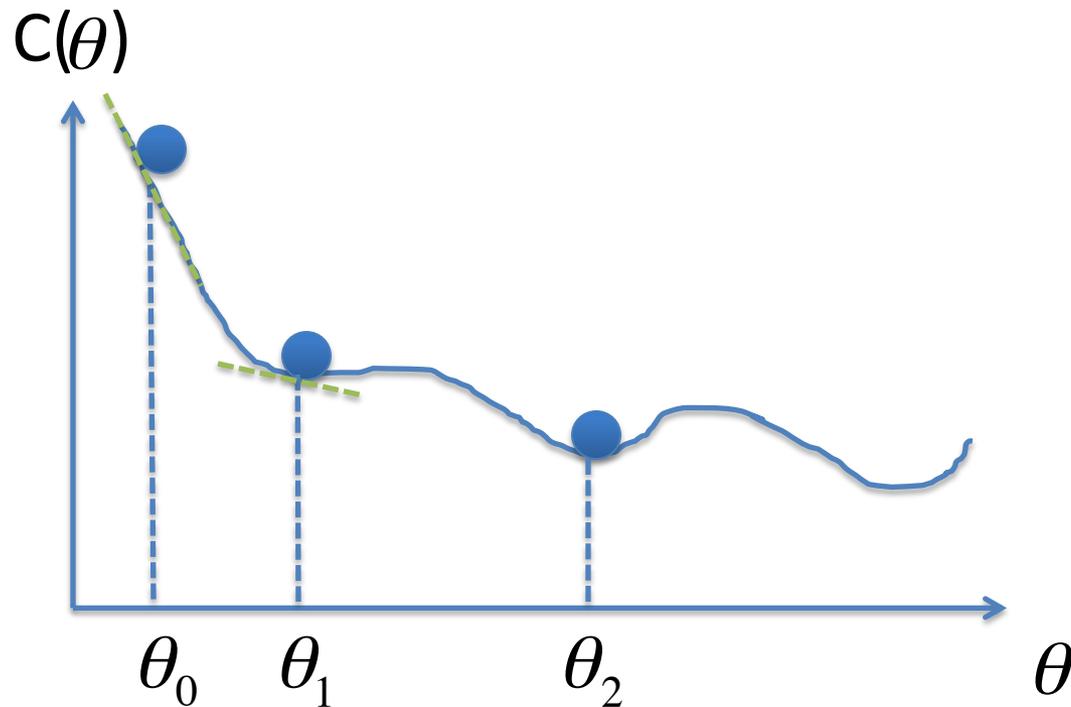
Empirical risk minimization

- Given a finite set of training data
- Empirical risk = average loss on this training data

$$\begin{aligned} C(\theta) &= \frac{1}{|D|} \sum_{(x,y)} c(f(x), y) \\ &= \frac{1}{|D|} \sum_{(x,y)} c(\theta) \end{aligned}$$

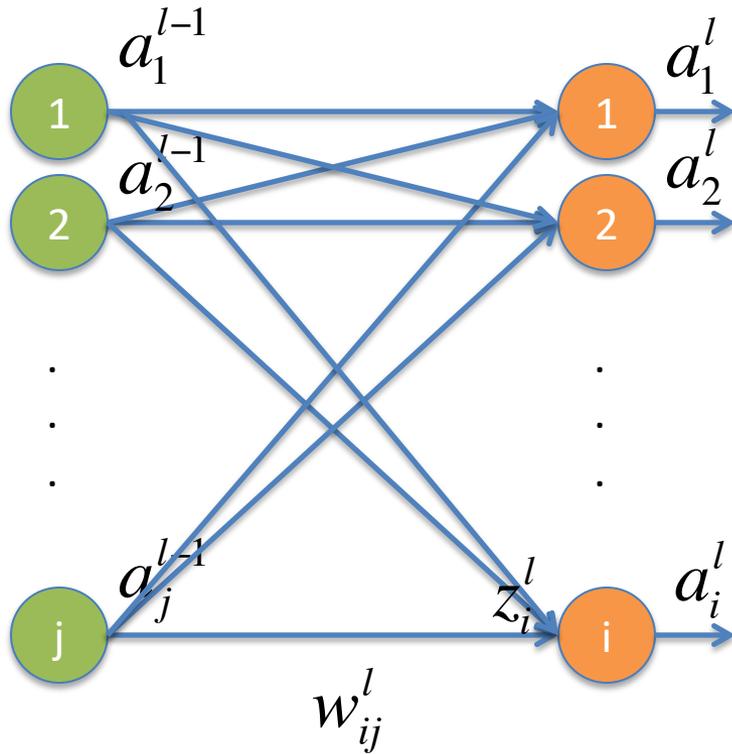
Gradient descent

- Consider that θ has only one variable



- Randomly start at θ_0
- Compute $dC(\theta_0)/d\theta$
 $\theta_1 \leftarrow \theta_0 - \eta dC(\theta_0)/d\theta$
- Compute $dC(\theta_1)/d\theta$
 $\theta_2 \leftarrow \theta_1 - \eta dC(\theta_1)/d\theta$
-

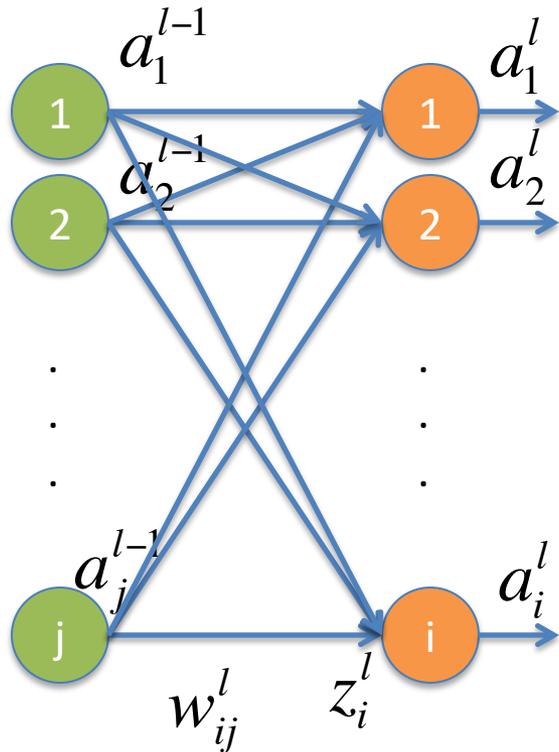
Compute $\frac{\partial C}{\partial w_{ij}^l}$



$$\Delta w_{ij}^l \rightarrow \Delta z_i^l \dots \rightarrow \Delta C$$

$$\frac{\partial C}{\partial w_{ij}^l} = \boxed{\frac{\partial z_i^l}{\partial w_{ij}^l}} \boxed{\frac{\partial C}{\partial z_i^l}}$$

Compute $\frac{\partial C}{\partial w_{ij}^l}$



$$\frac{\partial C}{\partial w_{ij}^l} = \frac{\partial z_i^l}{\partial w_{ij}^l} \frac{\partial C}{\partial z_i^l}$$

$$\begin{cases} a_j^{l-1} & l > 1 \\ x_j & l = 1 \end{cases}$$

$$\delta_i^l$$

Forward pass

$$z^l = W^l a^{l-1} + b^l$$

$$a^l = \sigma(z^l)$$

Backward pass

$$\delta^L = \sigma'(z^L) \nabla C(y)$$

$$\delta^l = \sigma'(z^l) (W^{l+1})^T \delta^{l+1}$$

The Core

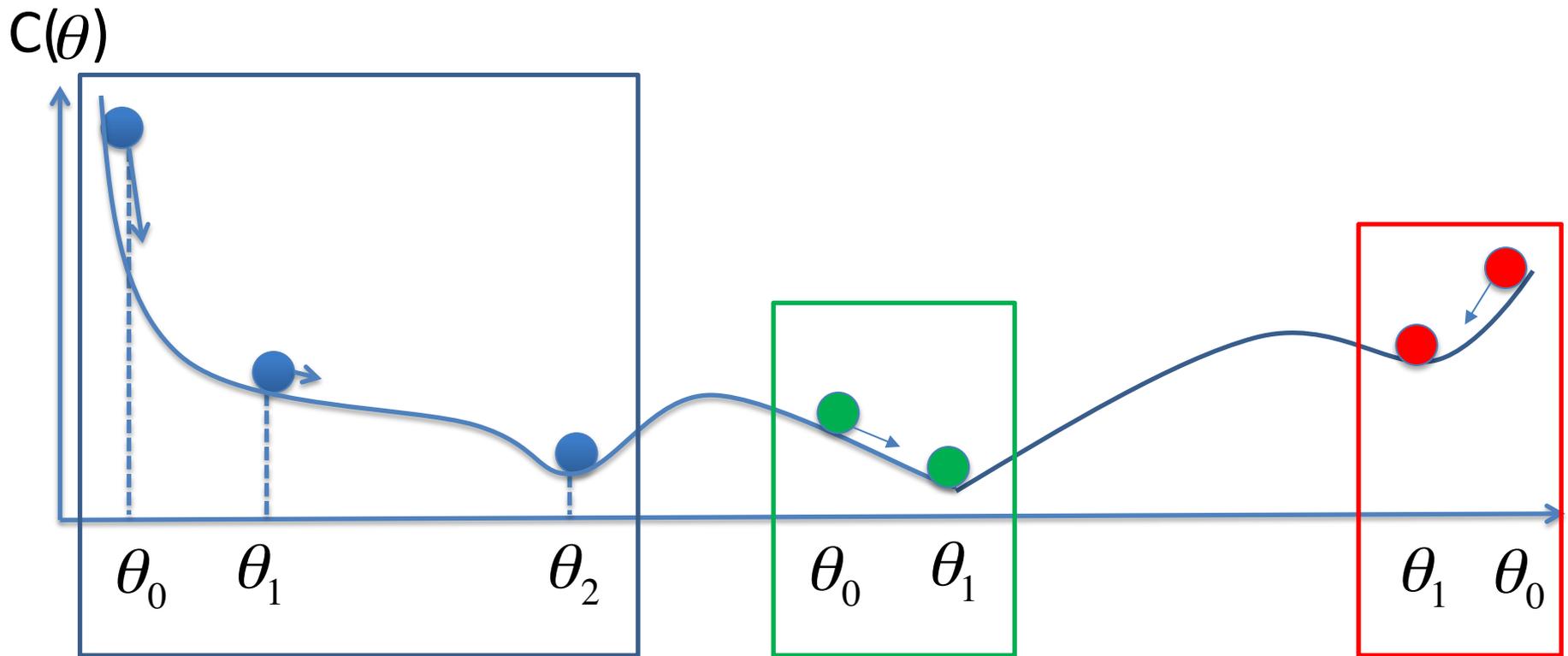
- Architectures
- Learning
- Tricks

Tricks

- Initialization Methods
- Adaptive Learning Rate
- Avoid Overfitting
- Very Deep Network

Challenges of Gradient Descent

- Depending on the initialization points, we will obtain different models and, therefore different results



Good Initialization Methods – Rule of Thumb

- Observations:
 - When initializing all weights with zeros, the neurons learn the same features during training
 - A too-large initialization leads to exploding gradients
 - A too-small initialization leads to vanishing gradients
- Rule of thumb:
 - The *mean* of the activations should be zero
 - The *variance* of the activations should stay the same across every layer

Initialization Methods

- Uniform Distribution
- Normal Distribution
- Xavier or Glorot methods - 2010
 - Either using uniform or normal distribution
- Kaiming or He methods – 2015
 - Either using uniform or normal distribution

Xavier or Glorot Methods

- Understanding the difficulty of training deep feedforward neural networks, Xavier Glorot and Yoshua Bengio, 2010
- Uniform:
 - Values are drawn from a uniform distribution $U(-a,a)$

$$a = gain \cdot \sqrt{\frac{6}{fan_{in} + fan_{out}}}$$

- Normal:
 - Values are drawn from a normal distribution $N(0, std^2)$

$$std = gain \cdot \sqrt{\frac{2}{fan_{in} + fan_{out}}}$$

Kaiming or He Methods

- Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification, Kaiming He et al 2015

- Uniform:

– Values are drawn from a uniform distribution $U(-b,b)$

$$b = gain \cdot \sqrt{\frac{3}{fan_{mode}}}$$

mode could be either in or out

- Normal:

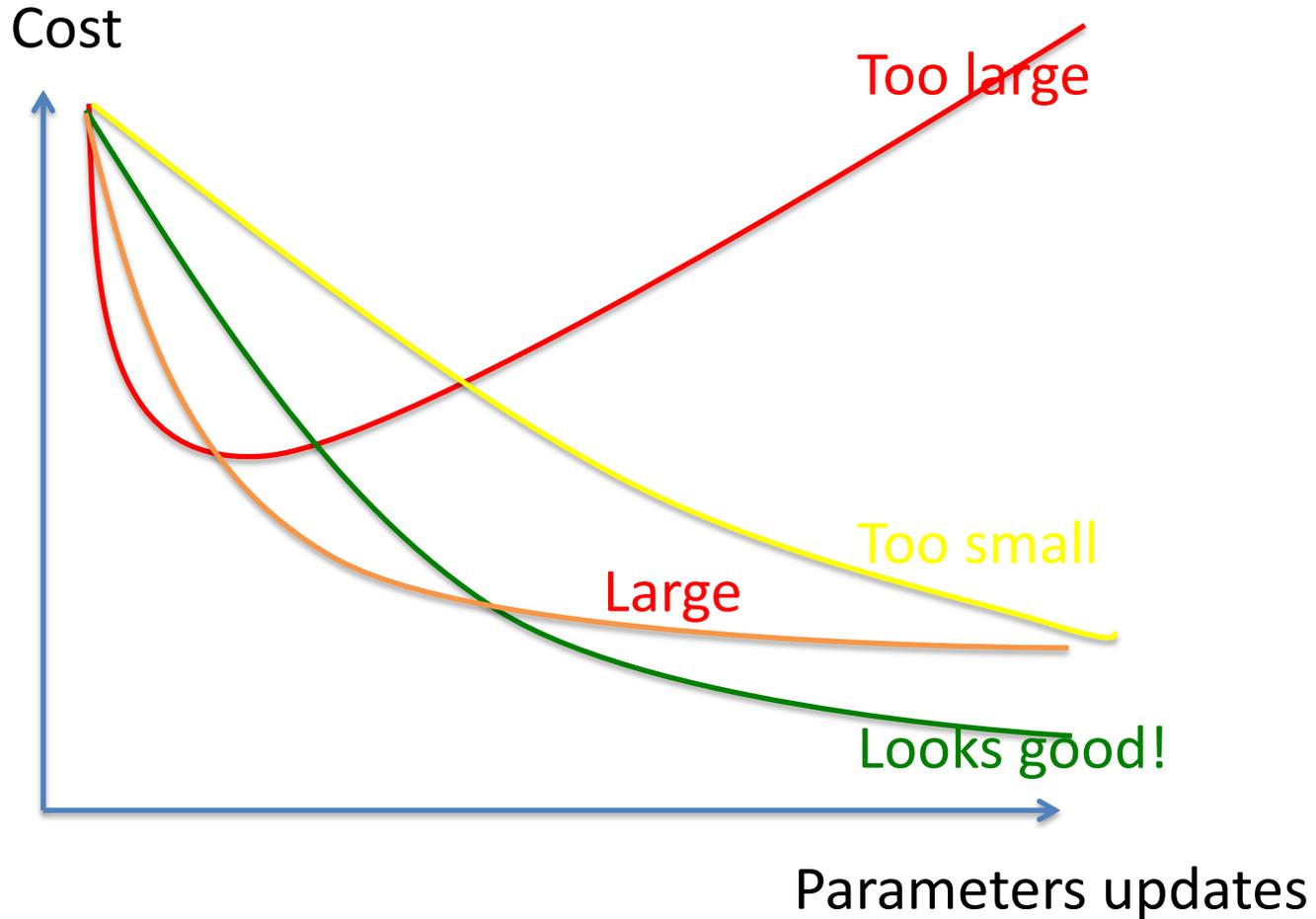
– Values are drawn from a normal distribution $N(0, std^2)$

$$std = gain \cdot \sqrt{\frac{2}{fan_{mode}}}$$

Tricks

- Initialization Methods
- **Adaptive Learning Rate**
- Avoid Overfitting
- Very Deep Network

Learning Rate



Learning Rate

- Popular & Simple idea: Reduce the learning rate by some factor every few epochs
 - At the beginning, larger learning rate
 - After several epochs, reduce the learning rate

$$\eta^t = \eta / (t + 1)$$

When to reduce the learning rate?

How much should we reduce the learning rate?

Challenges of Gradient Descent

- Choosing a proper learning rate can be difficult
- Learning rate schedules are helpful but need to be defined in advance
- The same learning rate for all parameter updates is problematic
- The problem of saddle points:
 - they are not local minima
 - the gradients there are close to zero, i.e. , difficult to escape

AdaGrad – Duchi et al 2011

- Divide the learning rate by the average gradient

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sigma} g^t$$

σ : Average gradient of parameter w

- If w has small average gradient
➔ Larger learning rate
- If w has large average gradient
➔ Smaller learning rate

Tricks

- Initialization Methods
- Adaptive Learning Rate
- **Avoid Overfitting**
- Very Deep Network

Methods

- Weight Decay
- Early Stopping
- Dropout
- Batch Normalization

Weight Decay

- It is also well known as ,Regularization' (L2)
- New cost function to be minimized

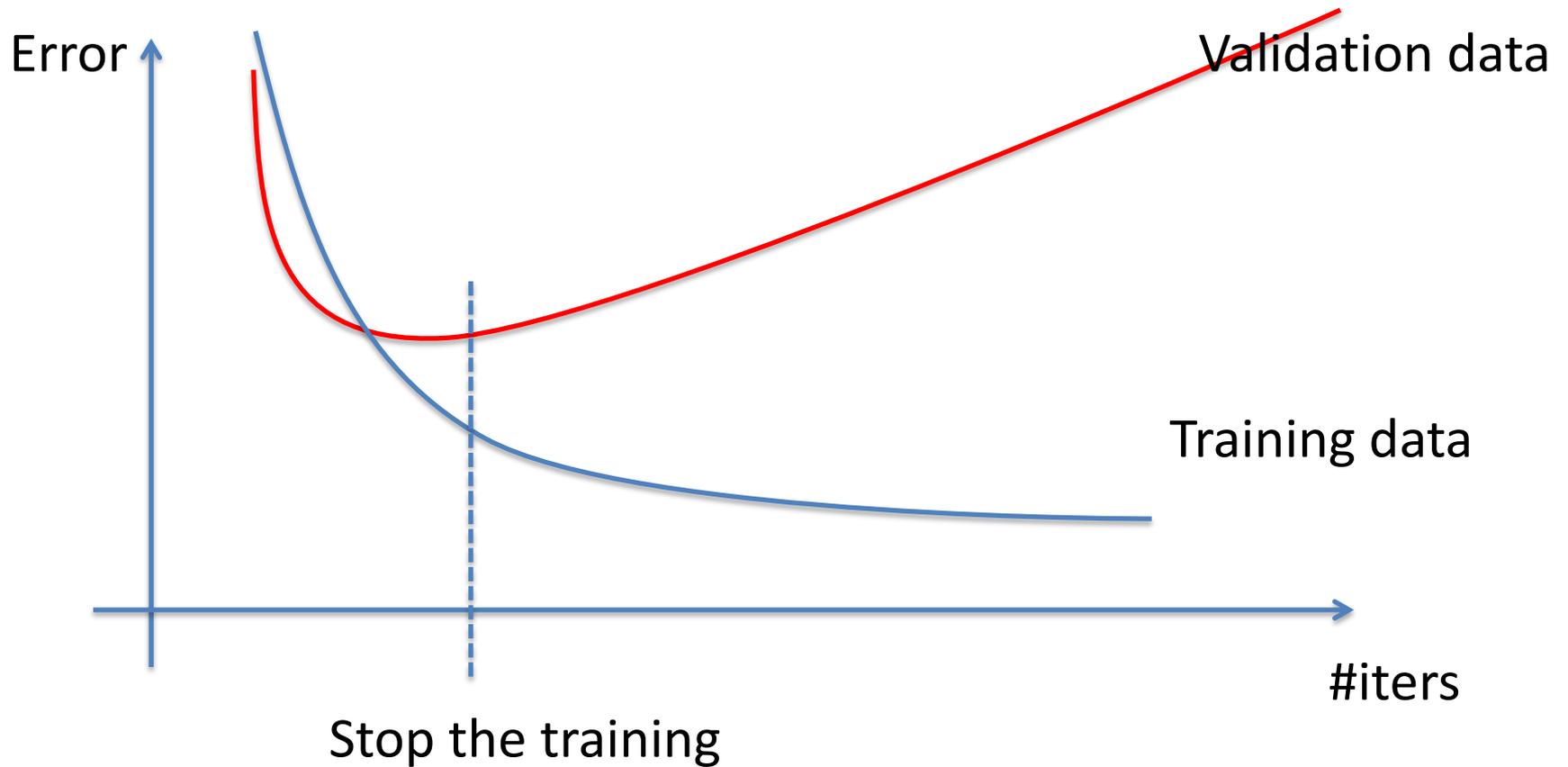
$$C'(\theta) = C(\theta) + \lambda \frac{1}{2} \|\theta\|^2 \rightarrow \text{Regularization term}$$



- Original cost to minimize
(e.g. cross entropy)

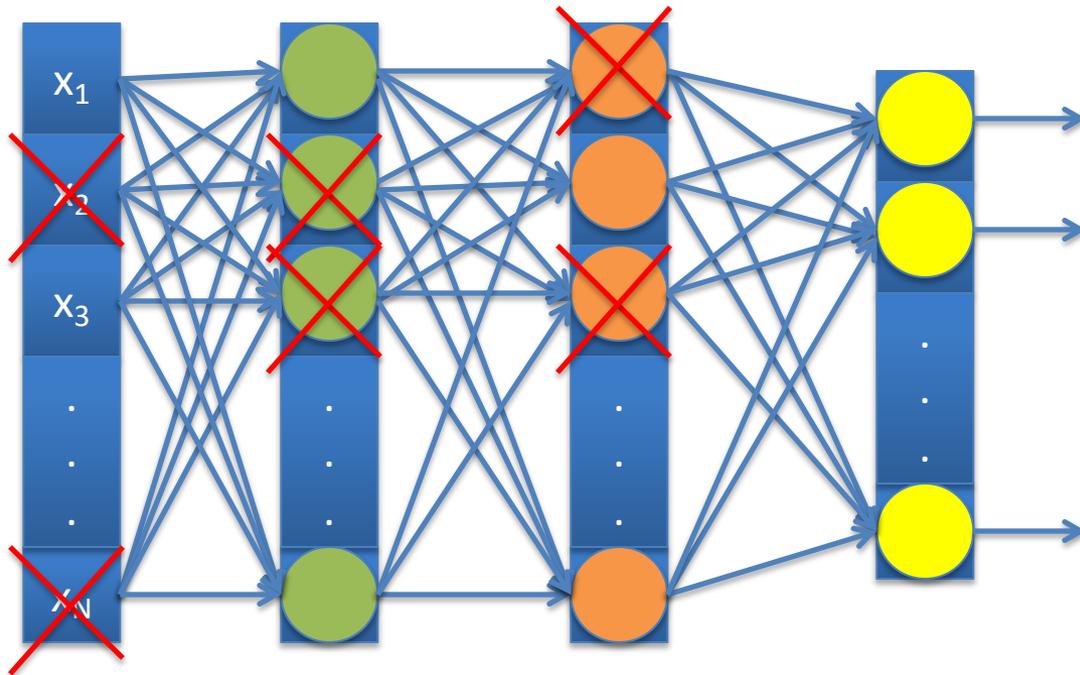
$$\theta = W^1, W^2, \dots$$

Early Stopping



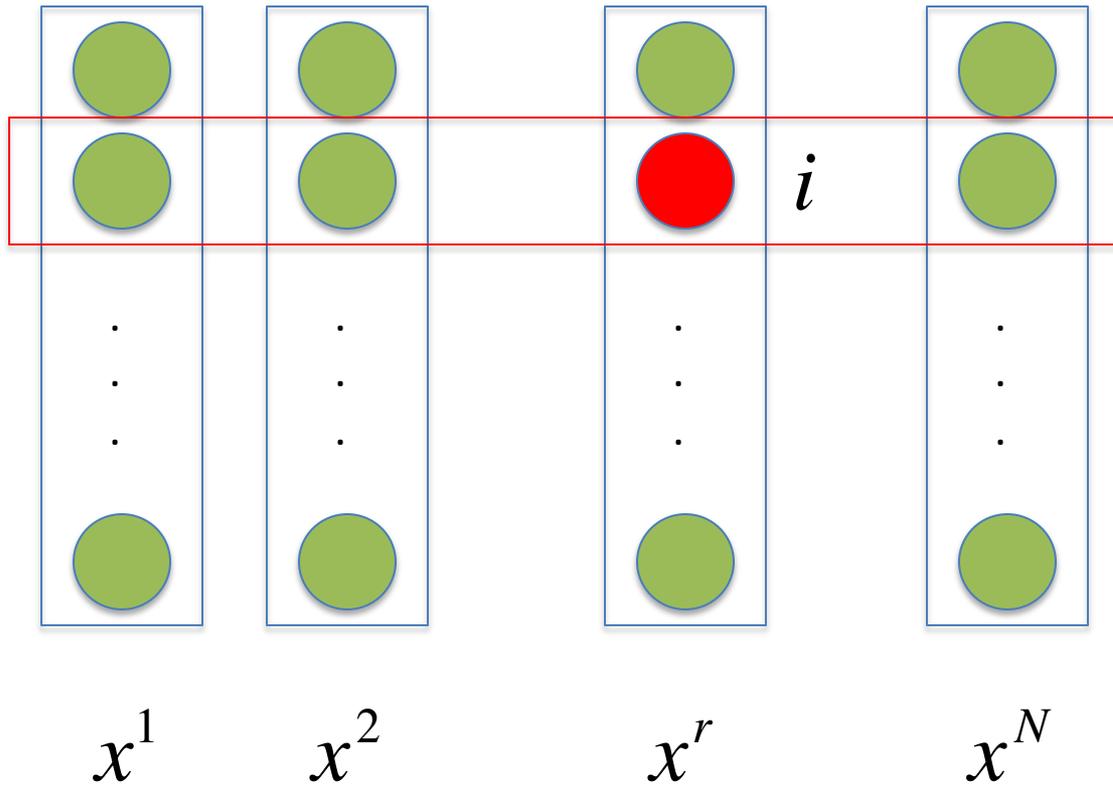
Dropout

Dropout: A Simple Way to Prevent Neural Networks from Overfitting, 2014



- In each iteration: Each neuron has $p\%$ to dropout during **training**

Data Processing

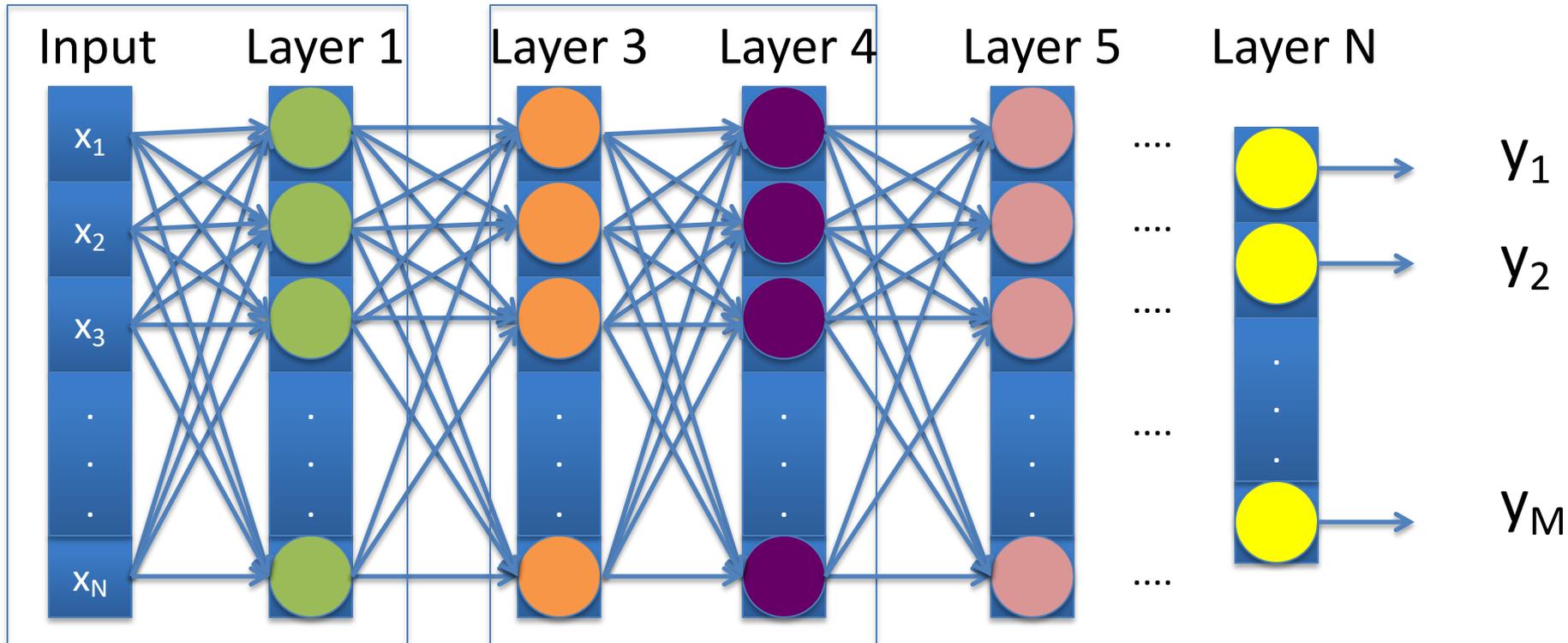


- For each of the dimension compute the mean m_i σ_i

$$x_i^r \leftarrow \frac{x_i^r - m_i}{\sigma_i}$$

- Mean = 0
- Variance = 1

Multilayer Perceptron



Batch Normalization

- Normalization of the input
- BN normalizes each layer, for each mini-batch

$$x \quad \rightarrow \quad \hat{x} \leftarrow \frac{x - \mu}{\sigma} \quad \rightarrow \quad \gamma \hat{x} + \beta$$

μ Mean of x in a mini-batch

γ Scale, parameter to learn

σ Std. of x in a mini-batch

β Shift, parameter to learn

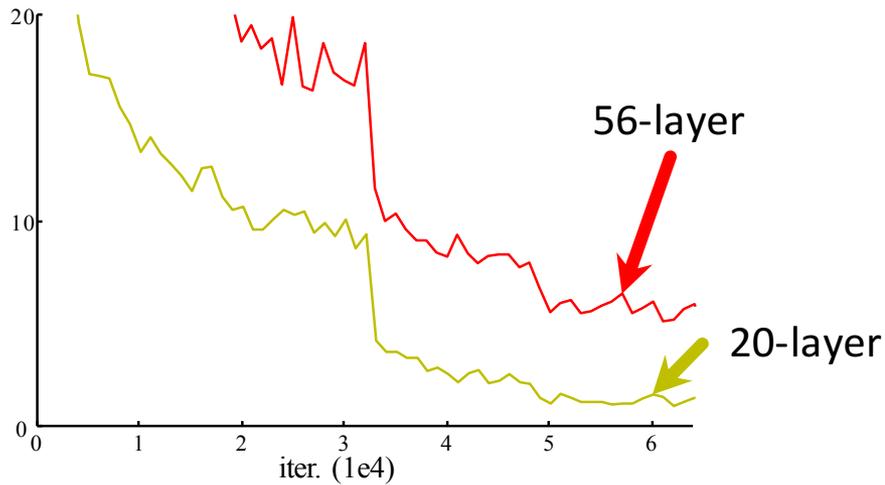
Tricks

- Initialization Methods
- Adaptive Learning Rate
- Avoid Overfitting
- **Very Deep Network**

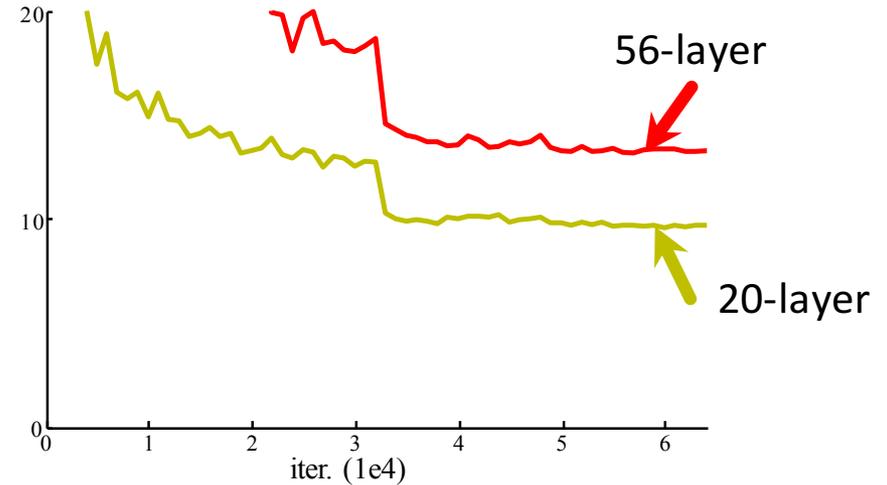
Simply Stacking Layer

CIFAR-10

train error (%)

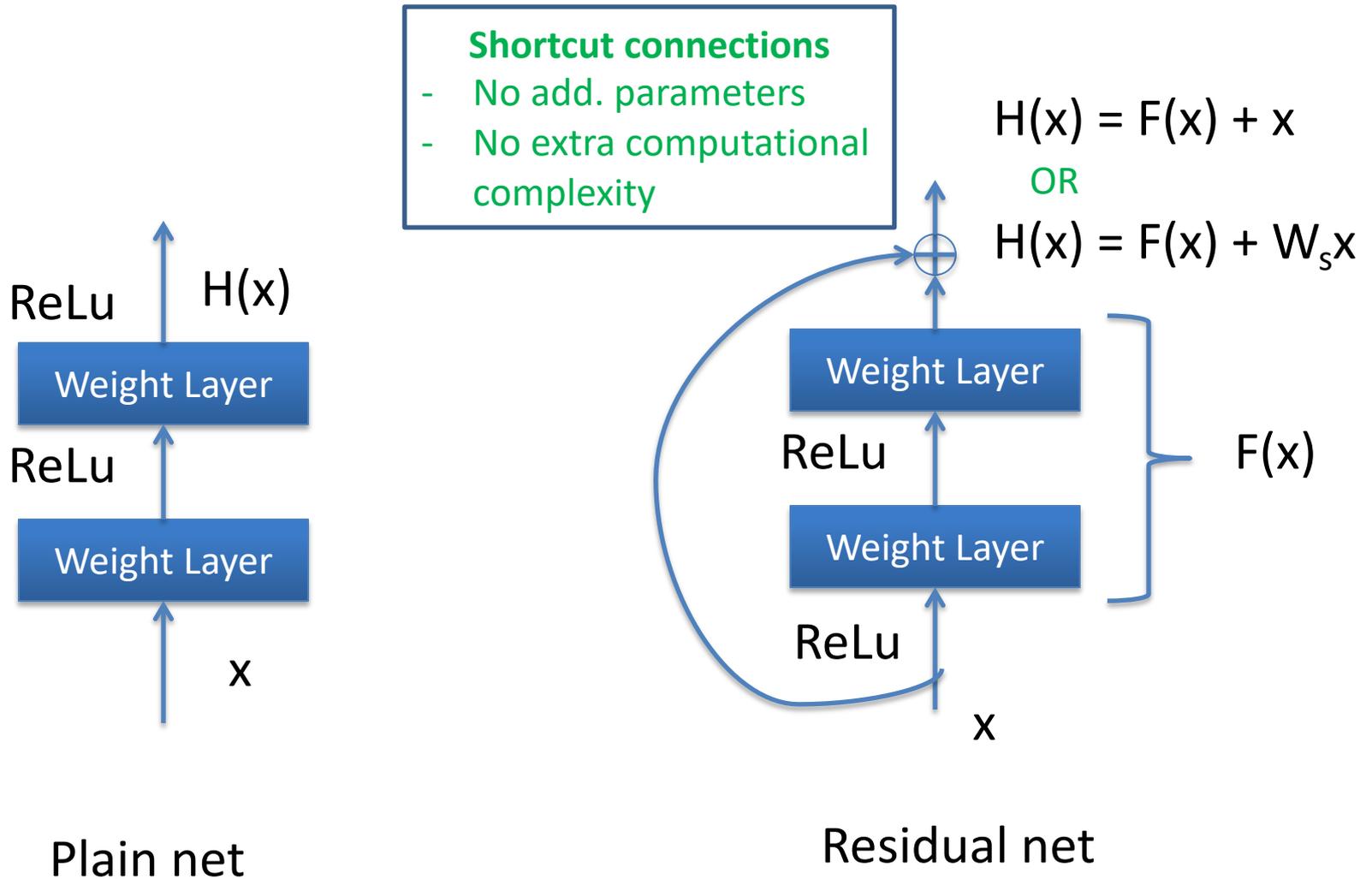


test error (%)



He et al. 2016

Deep Residual Network



Outline

- Recap
- Introduction
- Encoder-decoder Framework
- Attention Mechanism

Sequence to sequence (Seq2seq)

- The first paper:

*Sequence to sequence learning with neural networks –
Sutskever, Vinyals, Le - 2014*

Sequence to sequence (Seq2seq)

- The first paper:

Sequence to sequence learning with neural networks – Sutskever, Vinyals, Le - 2014

$$y^* = \arg \max_y p(y|x)$$

Sequence to sequence (Seq2seq)

- Sequence-to-sequence tasks are everywhere:
 - Speech Recognition
 - Machine translation
 - Text summarization
 - Conversational modeling
 - Image captioning

Neural Machine Translation

Auxiliary-main verb construction [aux:V]:

	SRC	in this experiment , individuals were shown hundreds of hours of YouTube videos	
	HPB	in diesem Experiment , Individuen gezeigt wurden Hunderte von Stunden YouTube-Videos	
(a)	PE	in diesem Experiment wurden Individuen Hunderte von Stunden Youtube-Videos gezeigt	✗
	NMT	in diesem Experiment wurden Individuen hunderte Stunden YouTube Videos gezeigt	
	PE	in diesem Experiment wurden Individuen hunderte Stunden YouTube Videos gezeigt	✓

Verb in subordinate (adjunct) clause [neb:V]:

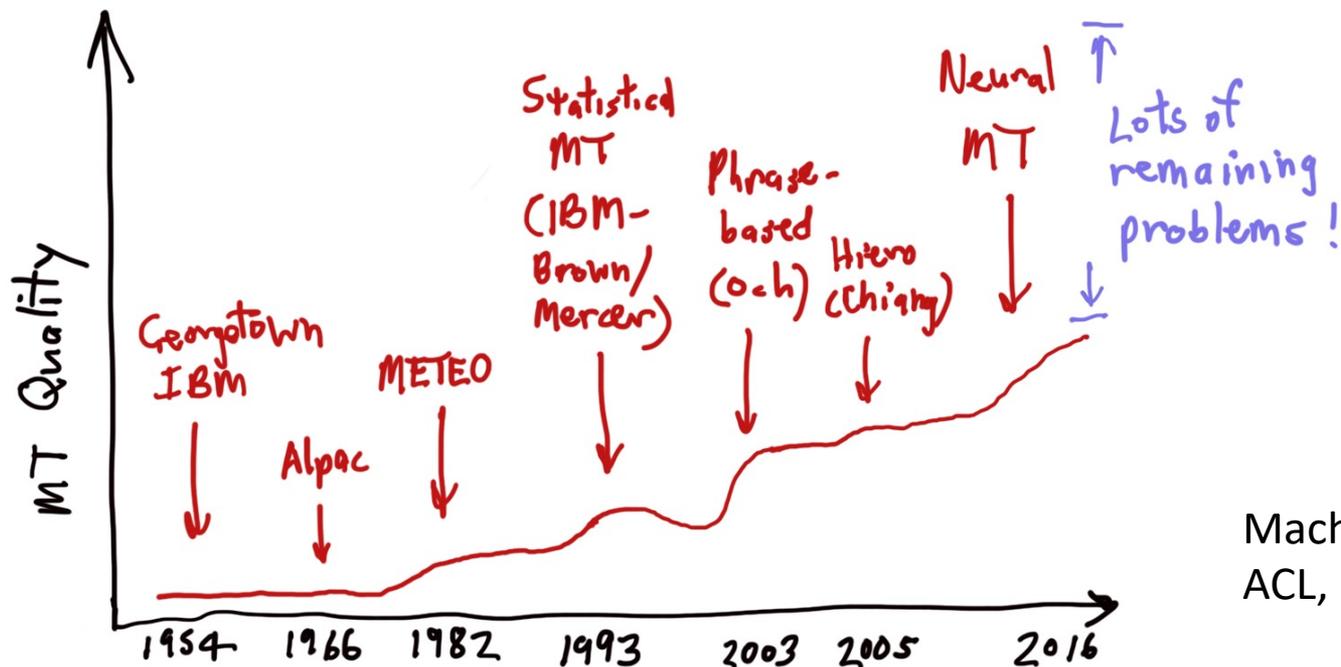
	SRC	... when coaches and managers and owners look at this information streaming ...	
	PBSY	... wenn Trainer und Manager und Eigentümer betrachten diese Information Streaming ...	
(b)	PE	... wenn Trainer und Manager und Eigentümer dieses Informations-Streaming betrachten ...	✗
	NMT	... wenn Trainer und Manager und Besitzer sich diese Informationen anschauen ...	
	PE	... wenn Trainer und Manager und Besitzer sich diese Informationen anschauen ...	✓

Bentivogli et al. 2016

Automatic Translation Systems

- Systems which translate text (speech) from one language to another language

Progress in MT



Machine Translation Tutorial
ACL, 2016

Neural Conversation Models

Human: *what is the purpose of life ?*

Machine: *to serve the greater good .*

Human: *what is the purpose of living ?*

Machine: *to live forever .*

Human: *what is the purpose of existence ?*

Machine: *to find out what happens when we get to the planet earth .*

Human: *where are you now ?*

Machine: *i 'm in the middle of nowhere .*

Human: *what is the purpose of dying ?*

Machine: *to have a life .*

Human: *what is the purpose of being intelligent ?*

Machine: *to find out what it is .*

Human: *what is the purpose of emotions ?*

Machine: *i don 't know .*

Machine: *hi*

Human: *hello*

Machine: *could you please let me know what are the operating systems you are using ?*

Human: *i am using linux*

Machine: *ok it was in the ssl vpn on which you stated it was broken*

Machine: *please check if you are logged in to corp network*

Machine: *what is the error that you are running please*

Human: *i am seeing an error related to vpn*

Machine: *what is the error message that you are getting when connecting to vpn using network connect ?*

Human: *connection refused or something like that*

Machine: *may i know the version of network connect you connect ?*

Human: *i am not sure i know that*

Vinyals and Le, 2015

Seq2seq Modeling

$$y^* = \arg \max_y p(y|x)$$

$$y' = \arg \max_y p(y|x, \theta)$$

model parameters

Questions we need to answer

- modeling

How does the model for $p(y|x, \theta)$ look like?

- learning

How to find θ ?

- search

How to find the argmax?

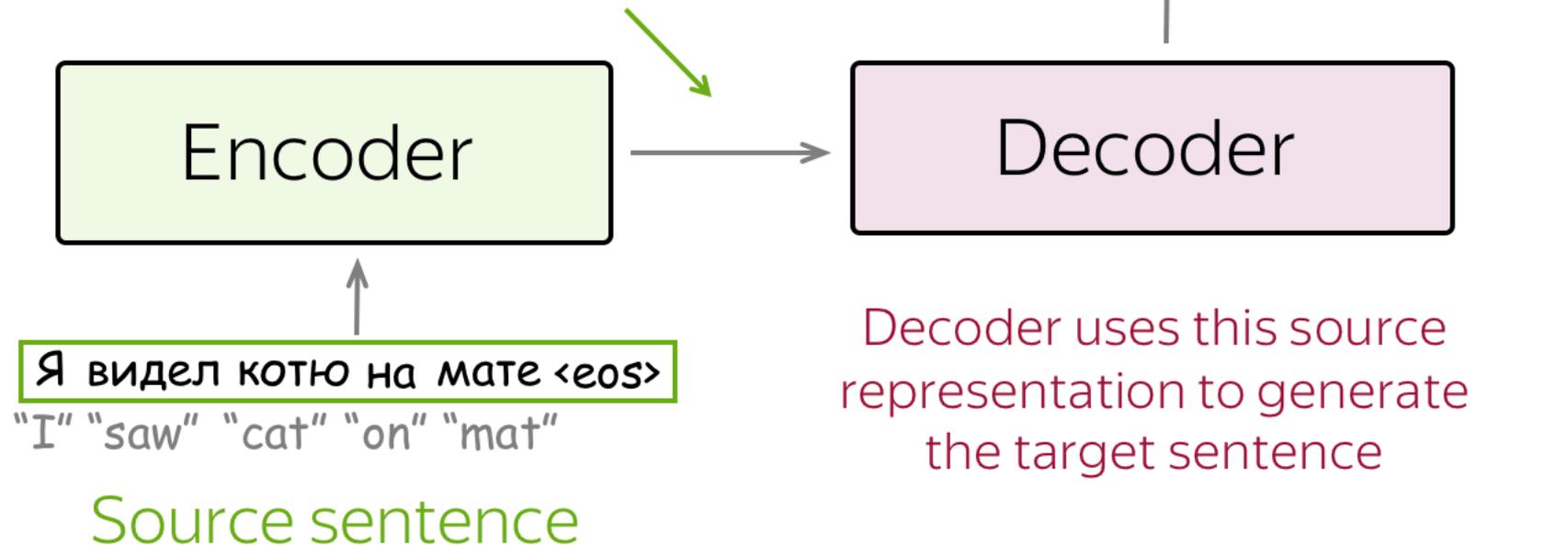
https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Outline

- Introduction
- **Encoder-decoder Framework**
- Attention Mechanism

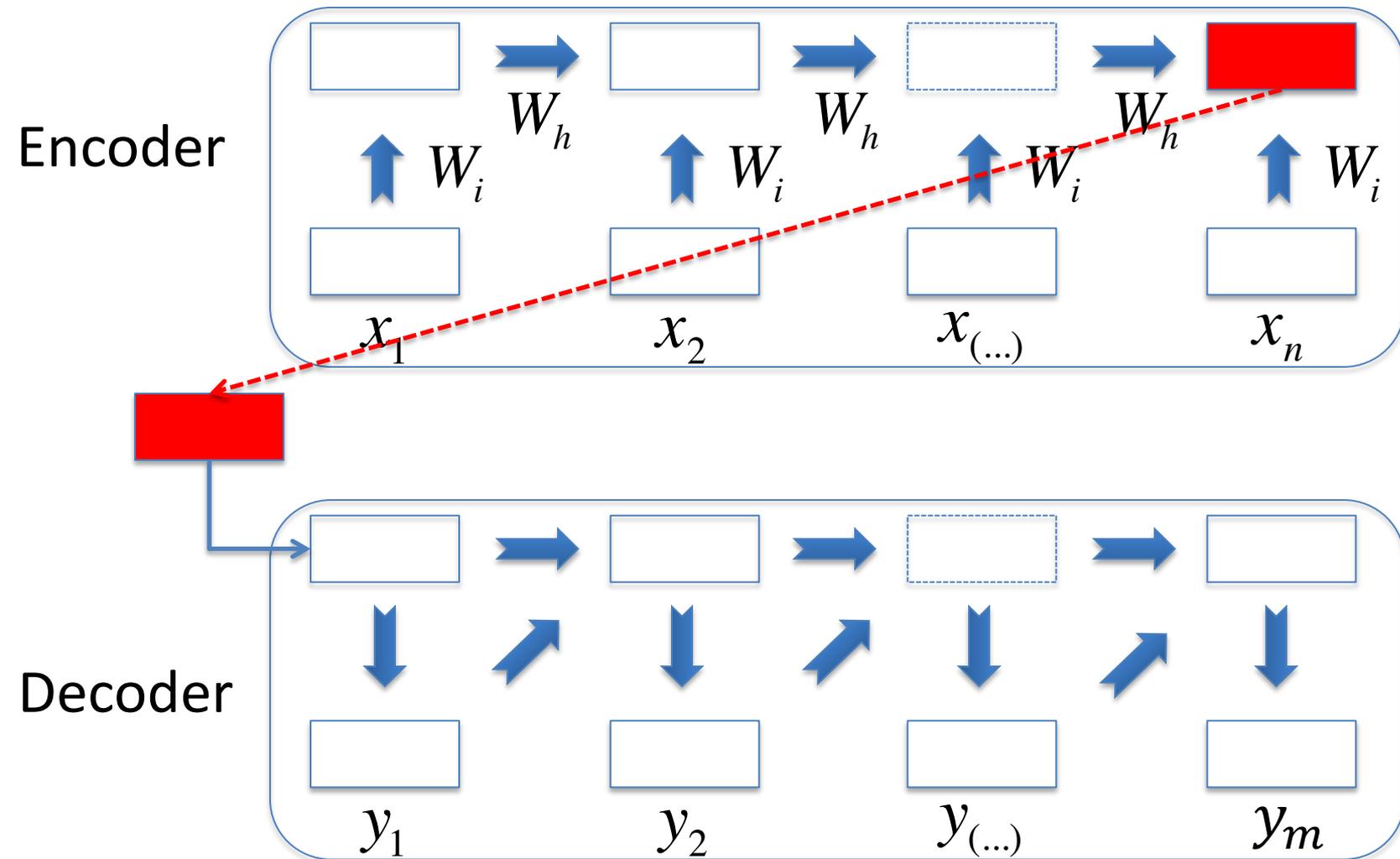
Encoder Decoder Framework

Encoder builds a representation of the source and gives it to the decoder

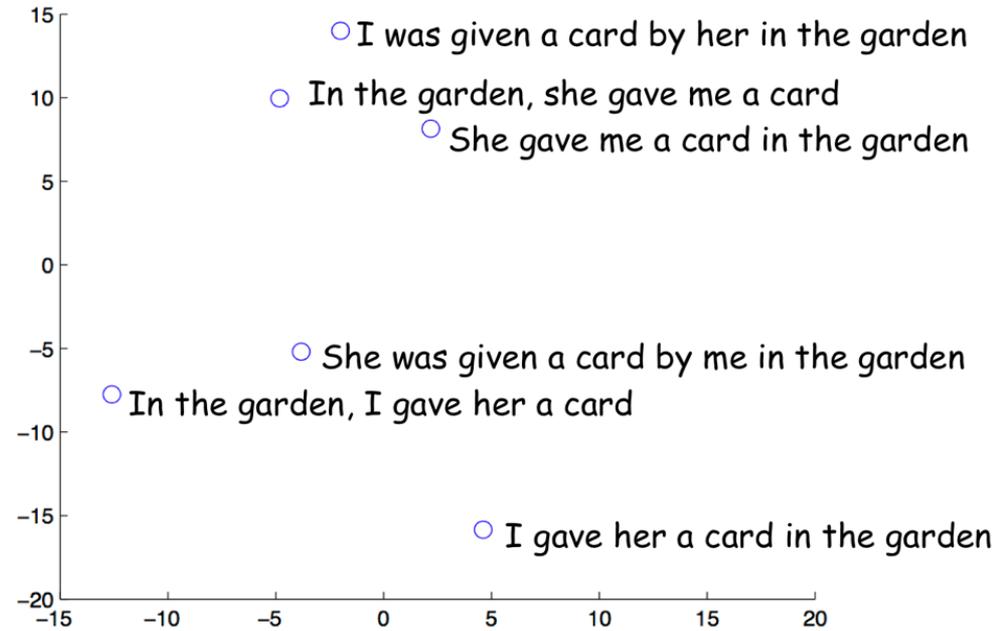
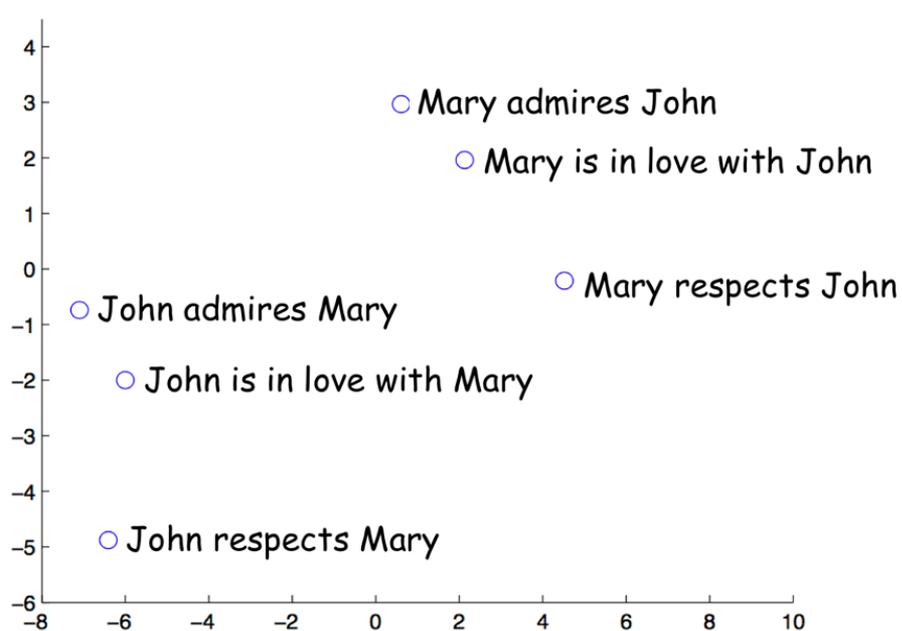


https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

The Simplest Seq2seq Models



Visualization



Sequence to sequence learning with neural networks – Sutskever, Vinyals, Le - 2014

Training Loss

$$\text{Loss}(p^*, p) = -\log(p_{y_t}) = -\log(p(y_t | y_{<t}, x)).$$

Source sequence:

Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"

Target sequence:

I saw a cat on a mat <eos>

← one training example

← one step for this example

previous tokens we want the model to predict this

Model prediction: $p(* | \text{I saw a, Я ... <eos>})$



Target



Loss = $-\log(p(\text{cat})) \rightarrow \min$



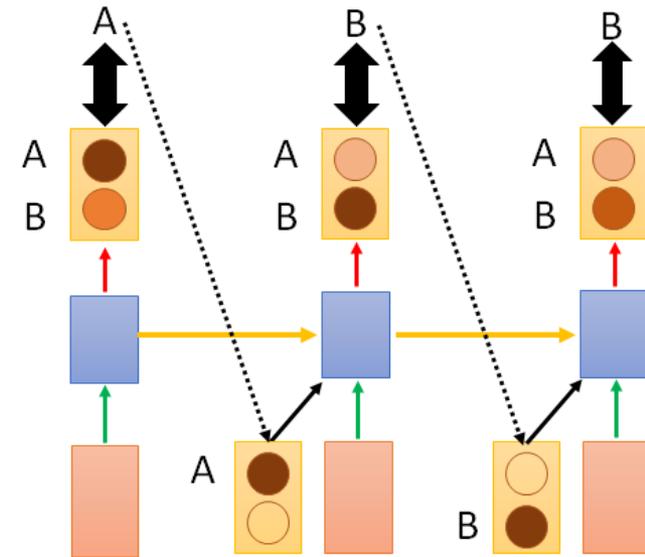
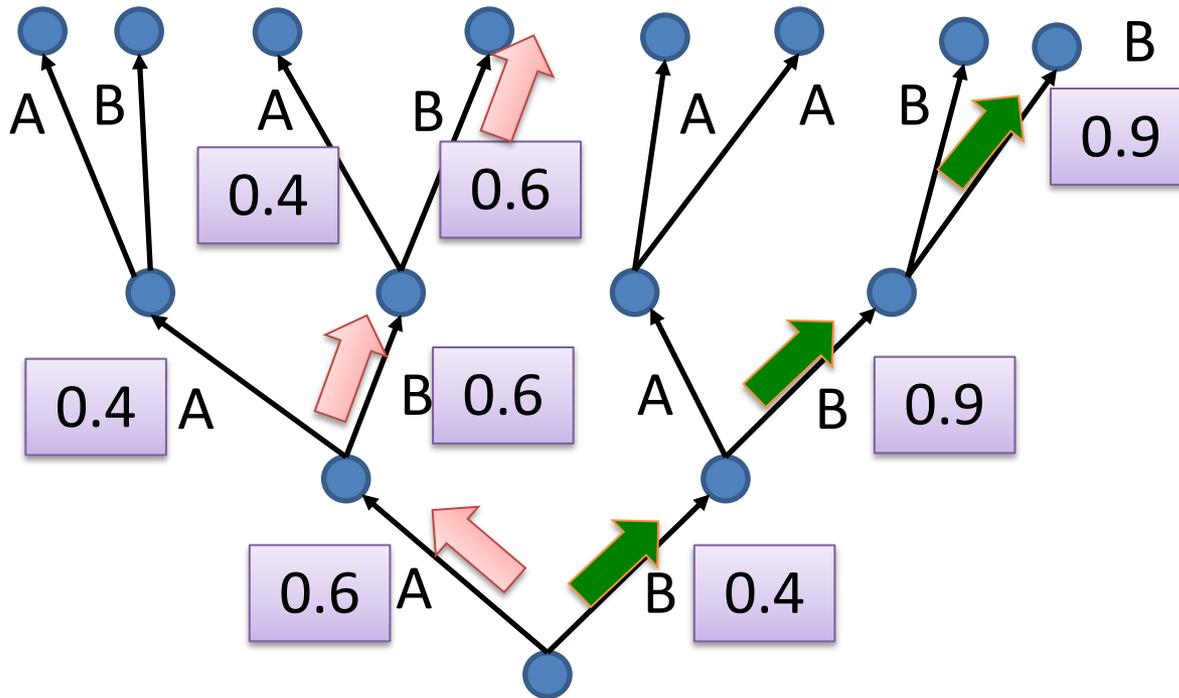
For the whole example, the loss will be:

$$-\sum_{t=1}^n \log(p(y_t | y_{<t}, x)).$$

https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

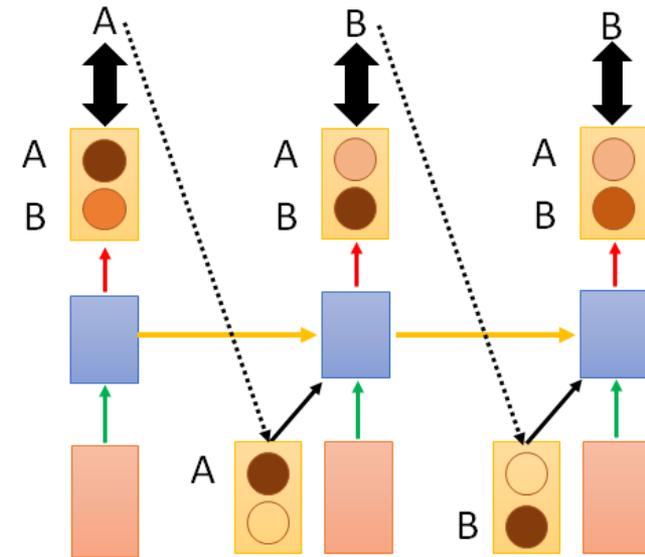
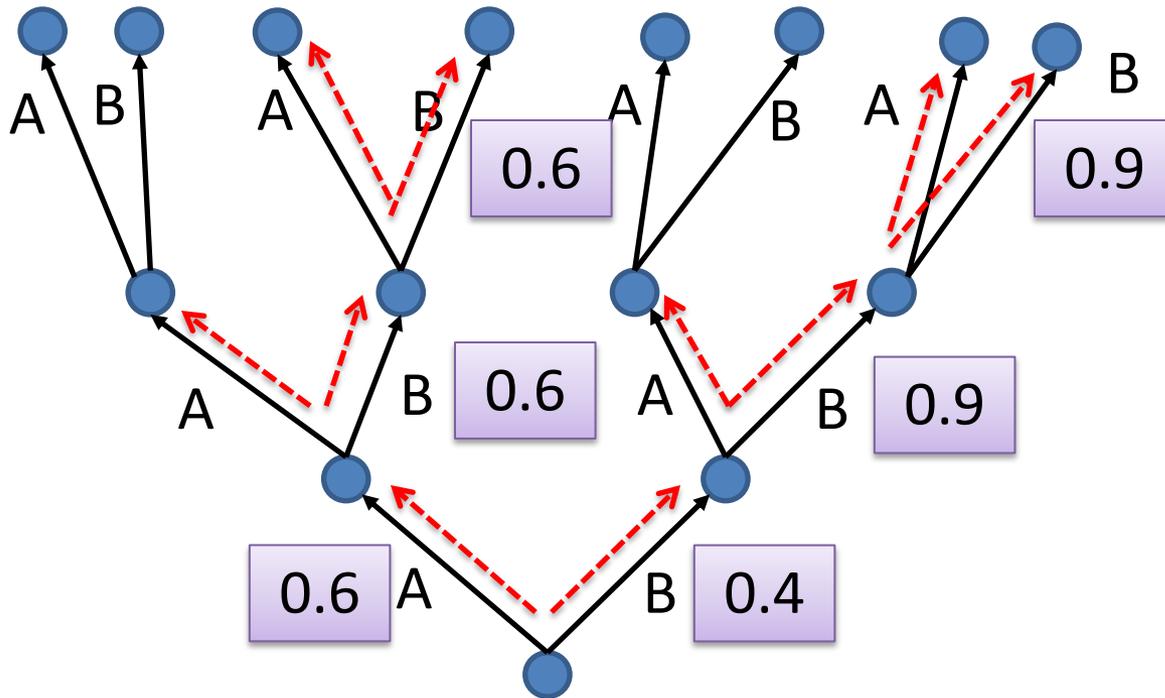
Greedy Search

- Greedy decoding doesn't give the best solution
- Not possible to keep track of all the paths



Beam Search

- In this case, keeping track of 2 hypotheses at each time step will give the best solution



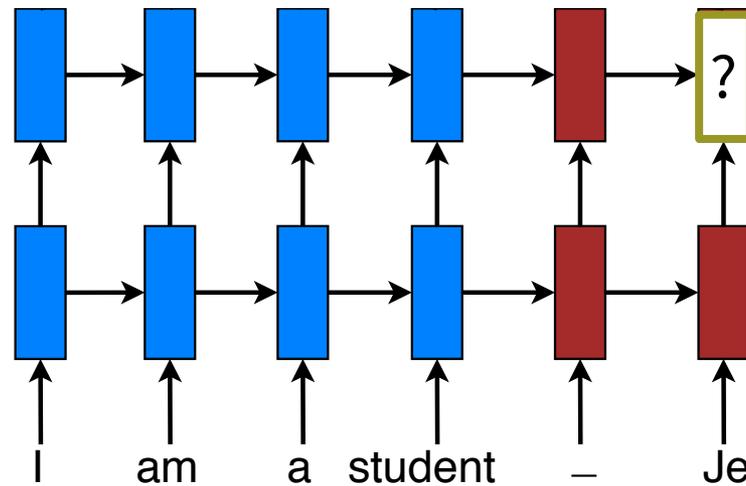
Seq2seq Models - Issues

- Problems of the original seq2seq model:
 - Seq2seq model encodes all the source information in a single vector
- However, we know that
 - Meaning of a sentence is complex
 - Content in a document is even more complex
 - Images have a lot of details
- ➔ It is impossible to put all the complex information in only one single vector
- Furthermore, for the decoder, different information might be relevant at different time steps

Outline

- Introduction
- Encoder-decoder Framework
- **Attention Mechanism**

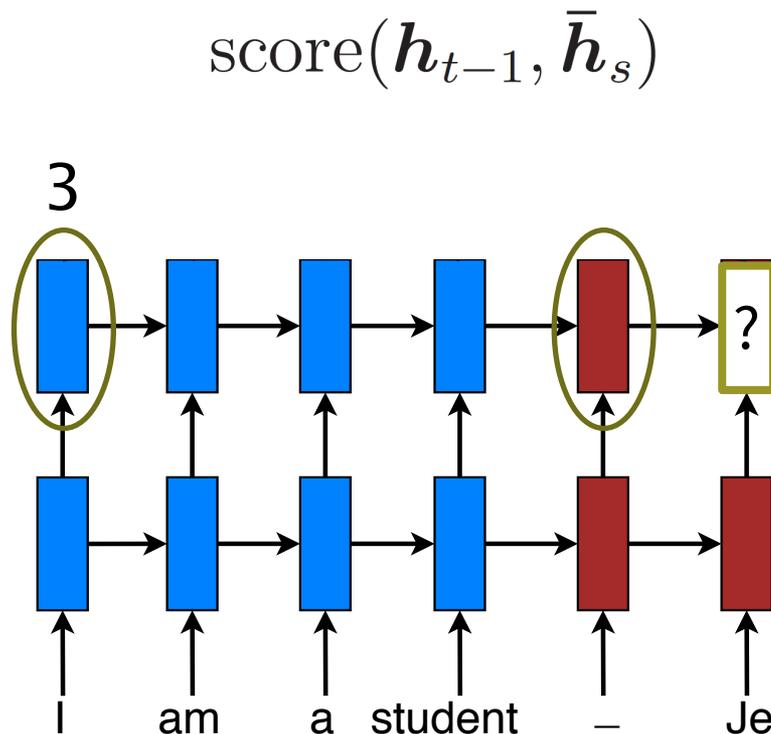
Attention mechanism – One Example



Machine Translation
Tutorial ACL, 2016

Attention Mechanism – One Example

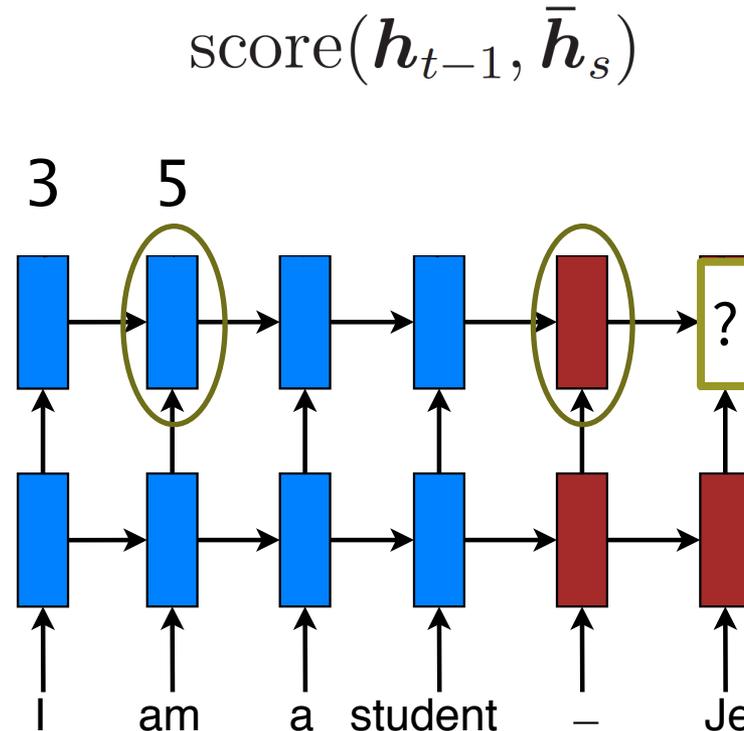
- Compare the source and the current target state



Machine Translation
Tutorial ACL, 2016

Attention Mechanism – One Example

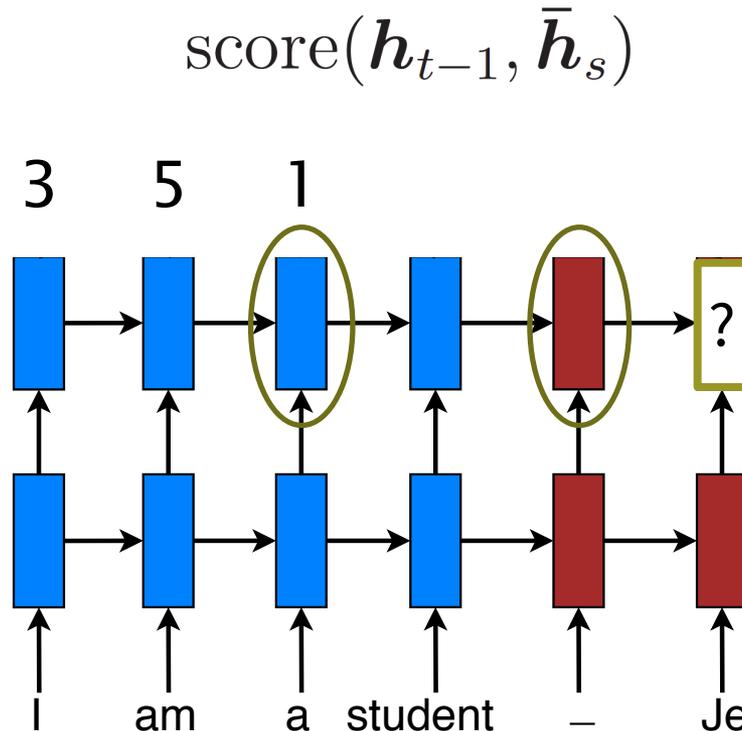
- Compare the source and the current target state



Machine Translation
Tutorial ACL, 2016

Attention Mechanism – One Example

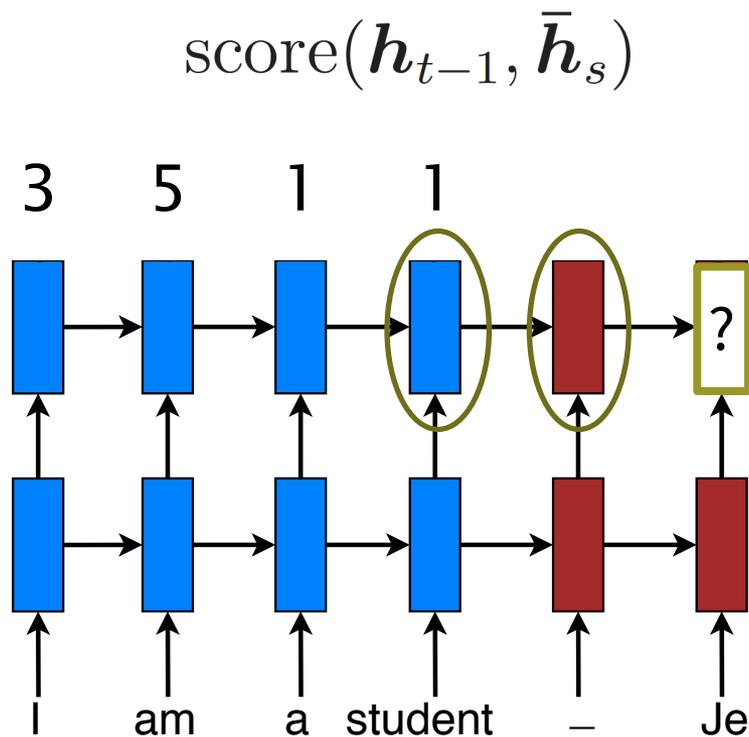
- Compare the source and the current target state



Machine Translation
Tutorial ACL, 2016

Attention Mechanism – One Example

- Compare the source and the current target state



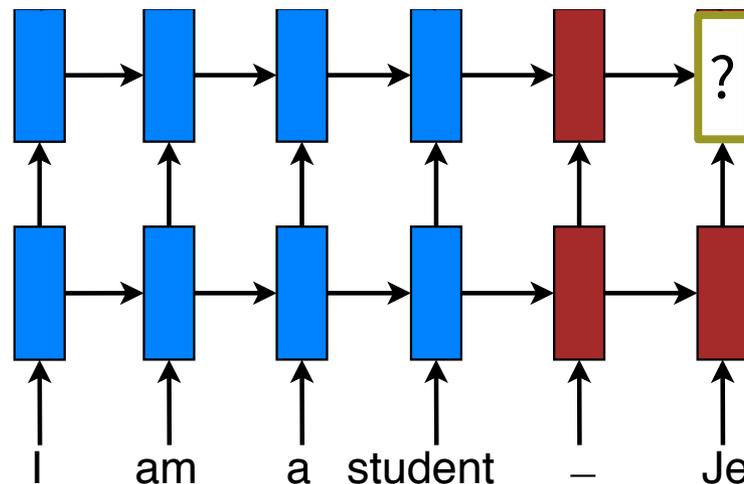
Machine Translation
Tutorial ACL, 2016

Attention Mechanism – One Example

- Normalize the scores

$$a_t(s) = \frac{e^{\text{score}(s)}}{\sum_{s'} e^{\text{score}(s')}}$$

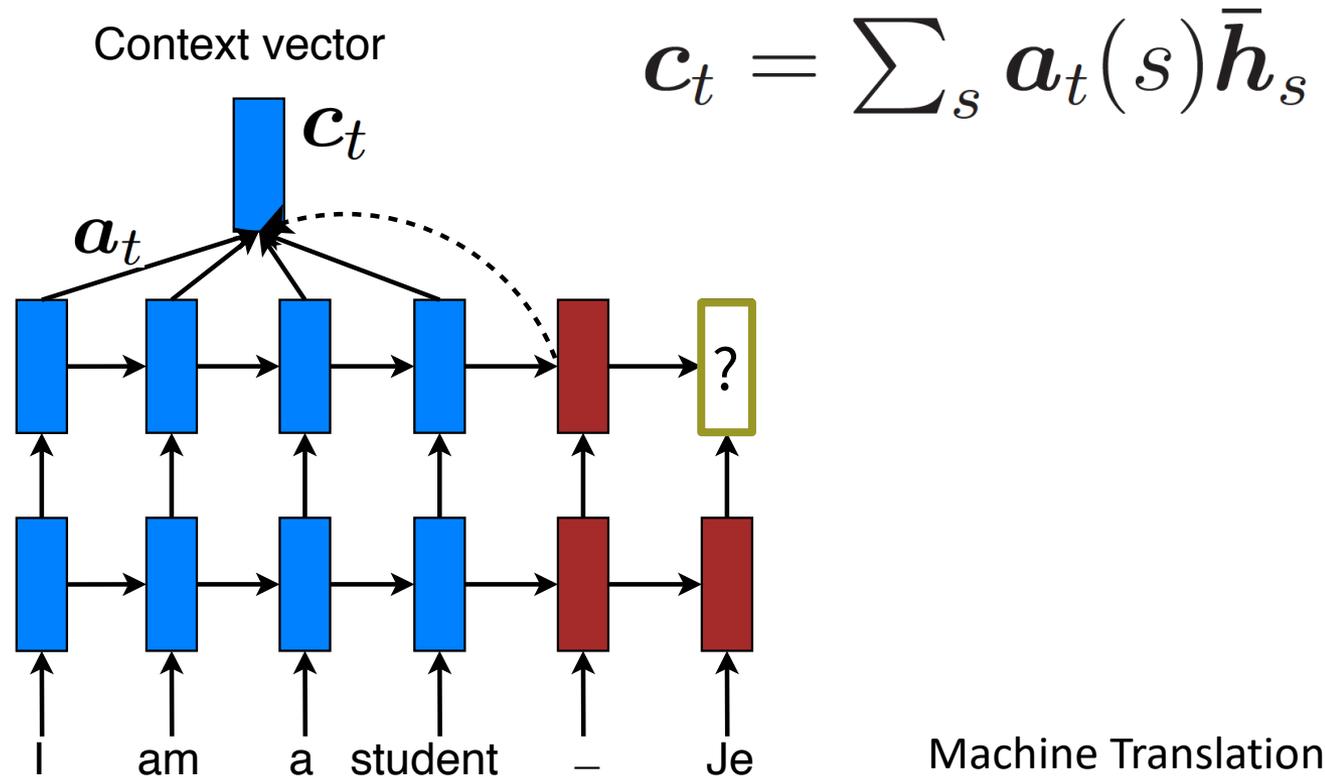
a_t 0.3 0.5 0.1 0.1



Machine Translation
Tutorial ACL, 2016

Attention Mechanism – One Example

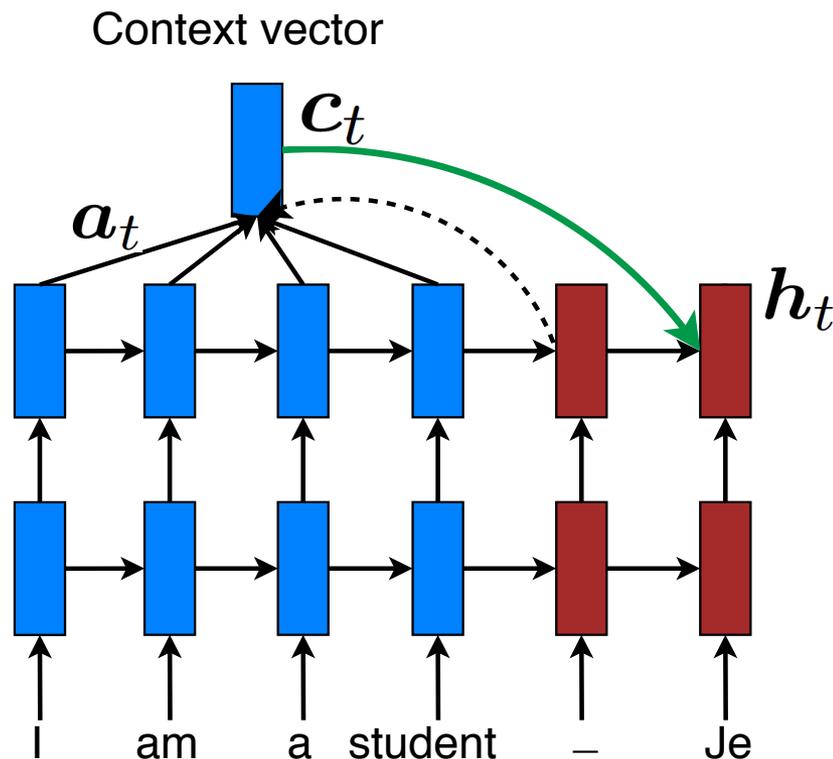
- Compute the context vector



Machine Translation
Tutorial ACL, 2016

Attention Mechanism – One Example

- Compute the next hidden state



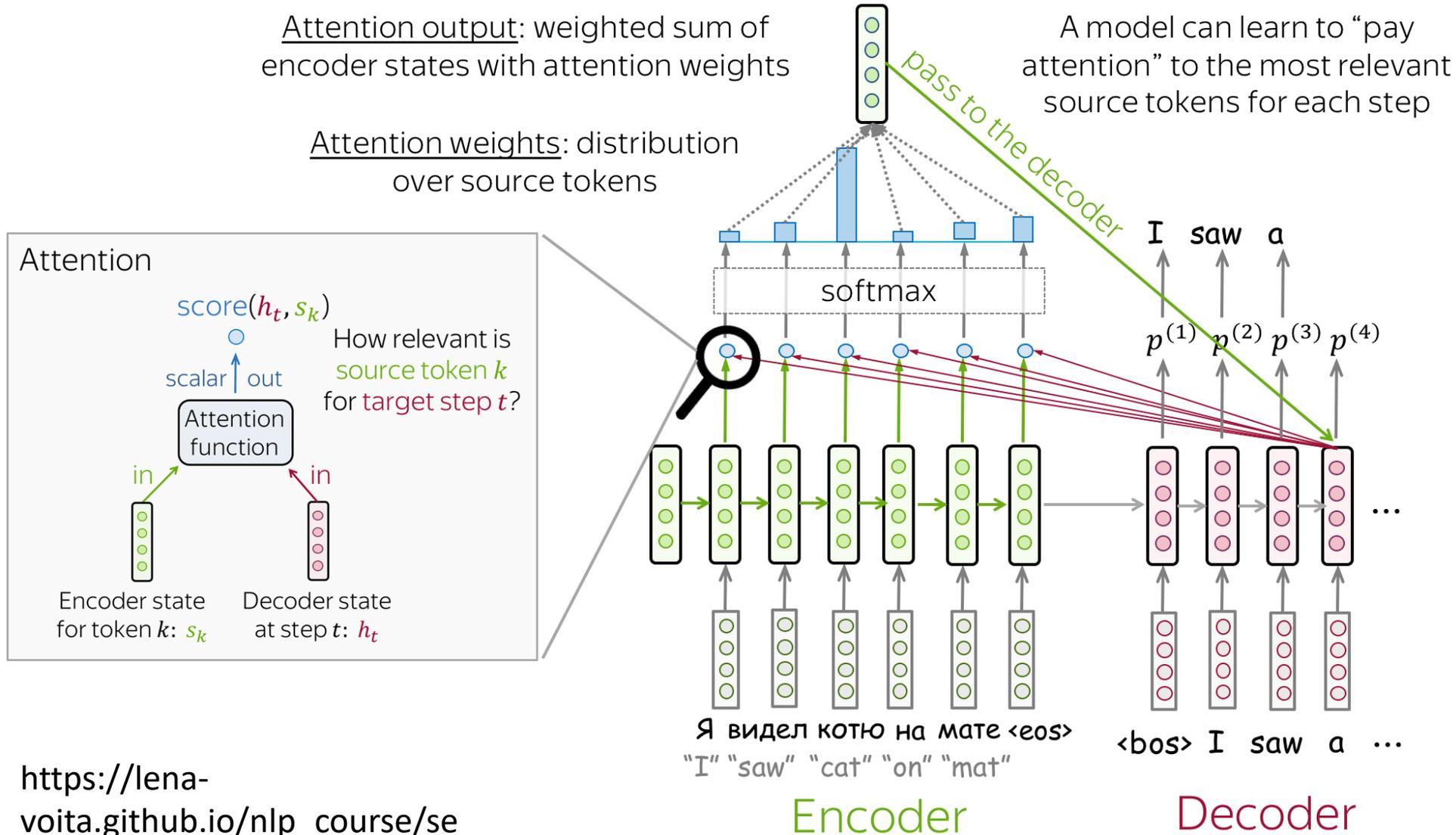
Machine Translation
Tutorial ACL, 2016

Attention Mechanism

Attention output: weighted sum of encoder states with attention weights

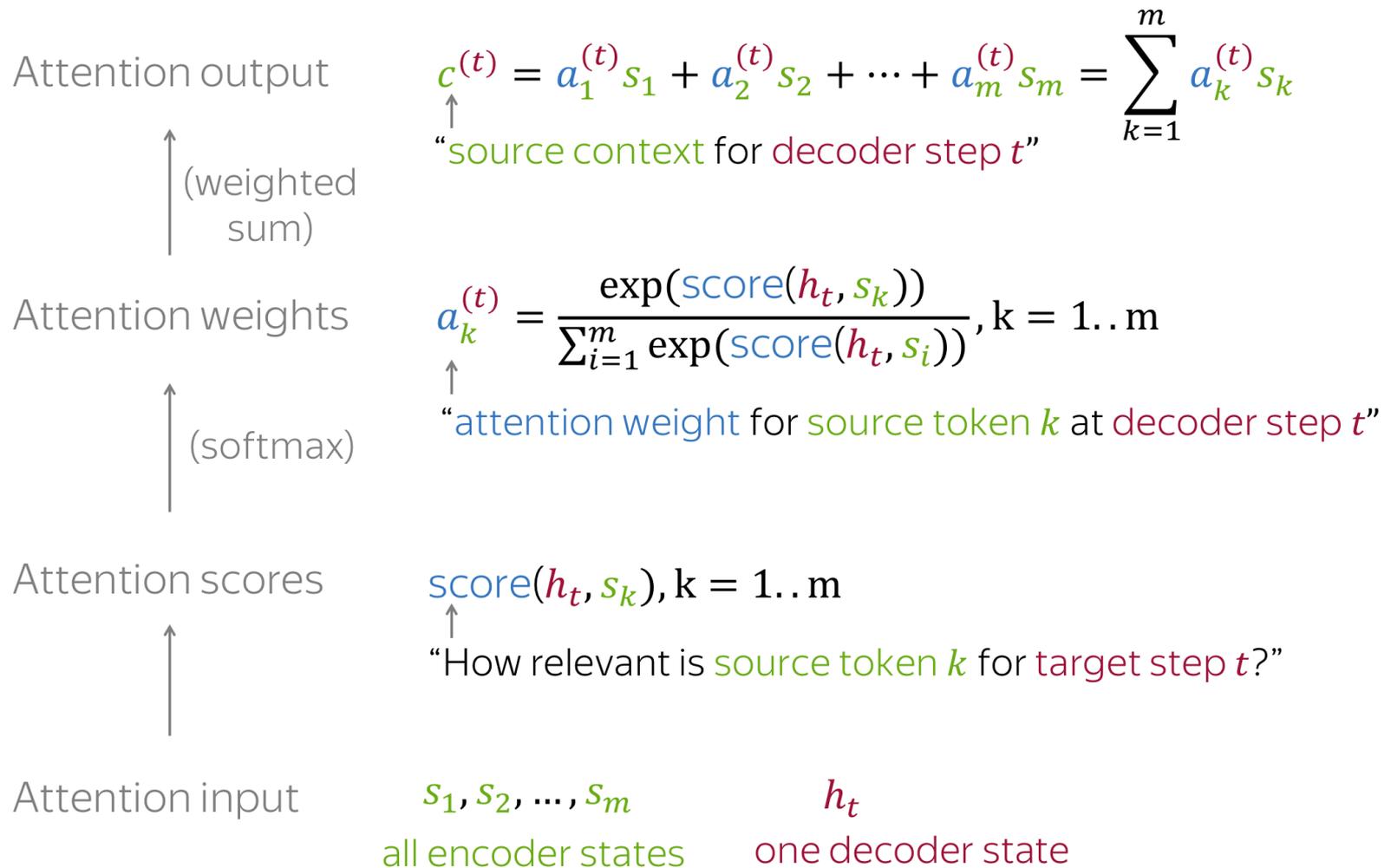
A model can learn to “pay attention” to the most relevant source tokens for each step

Attention weights: distribution over source tokens



https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Computational Process



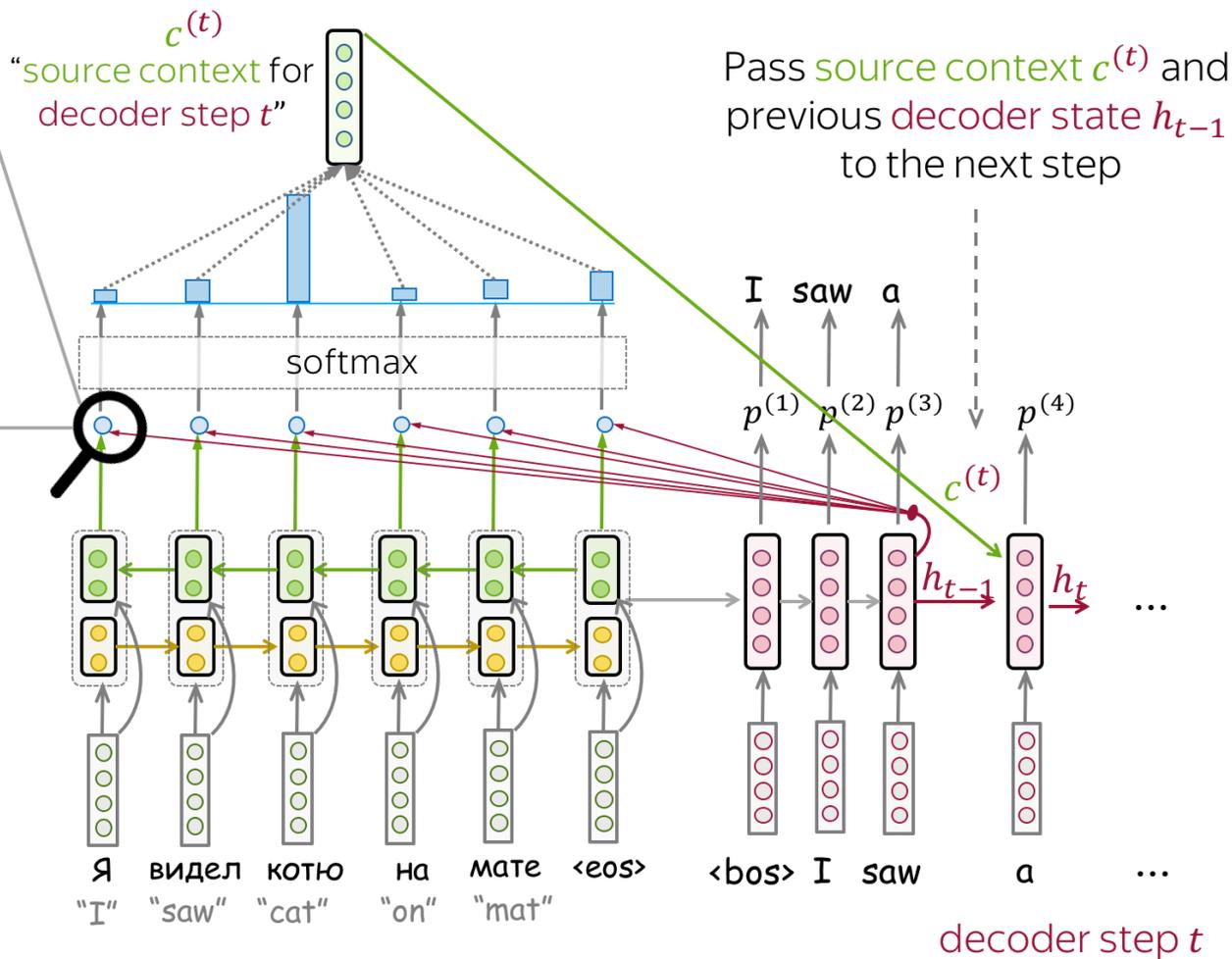
https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Bahdanau et al 2014

Multi-Layer Perceptron

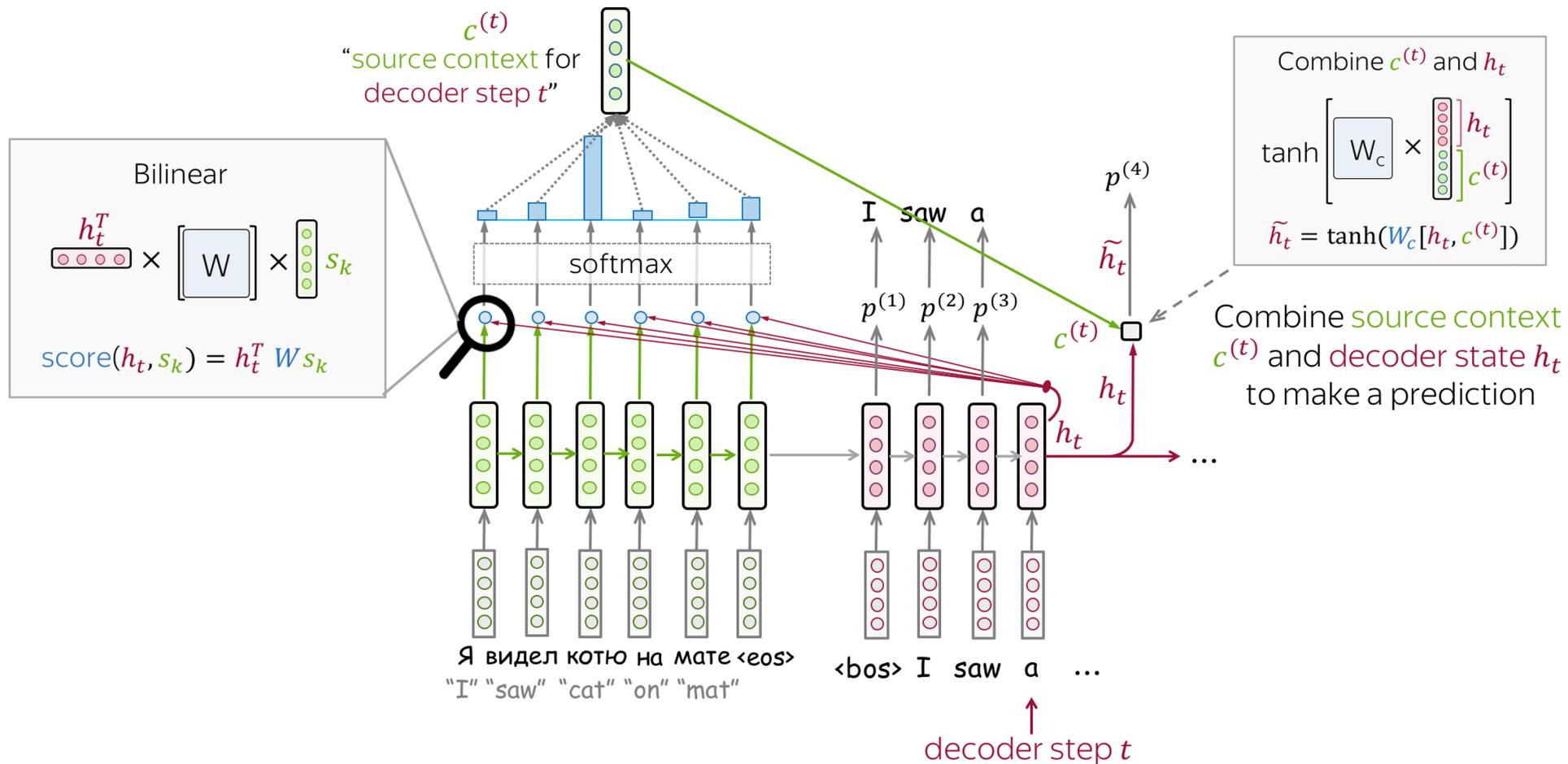
$$\text{score}(h, s_k) = w_2^T \cdot \tanh(W_1 [h, s_k])$$

Bidirectional encoder
 Concatenate states from
 forward and backward RNNs



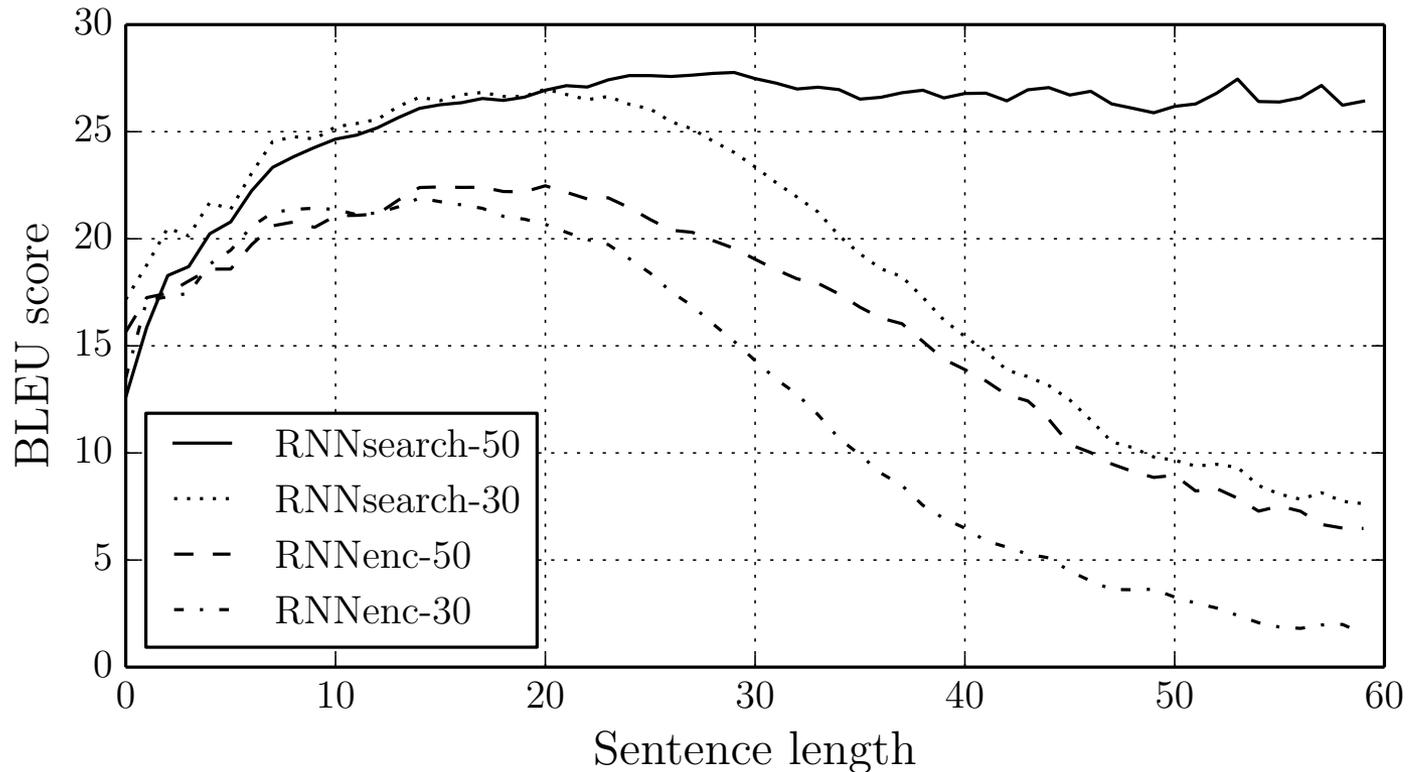
https://lenna-voita.github.io/nlp_course/seq2seq_and_attention.html

Luong et al 2015



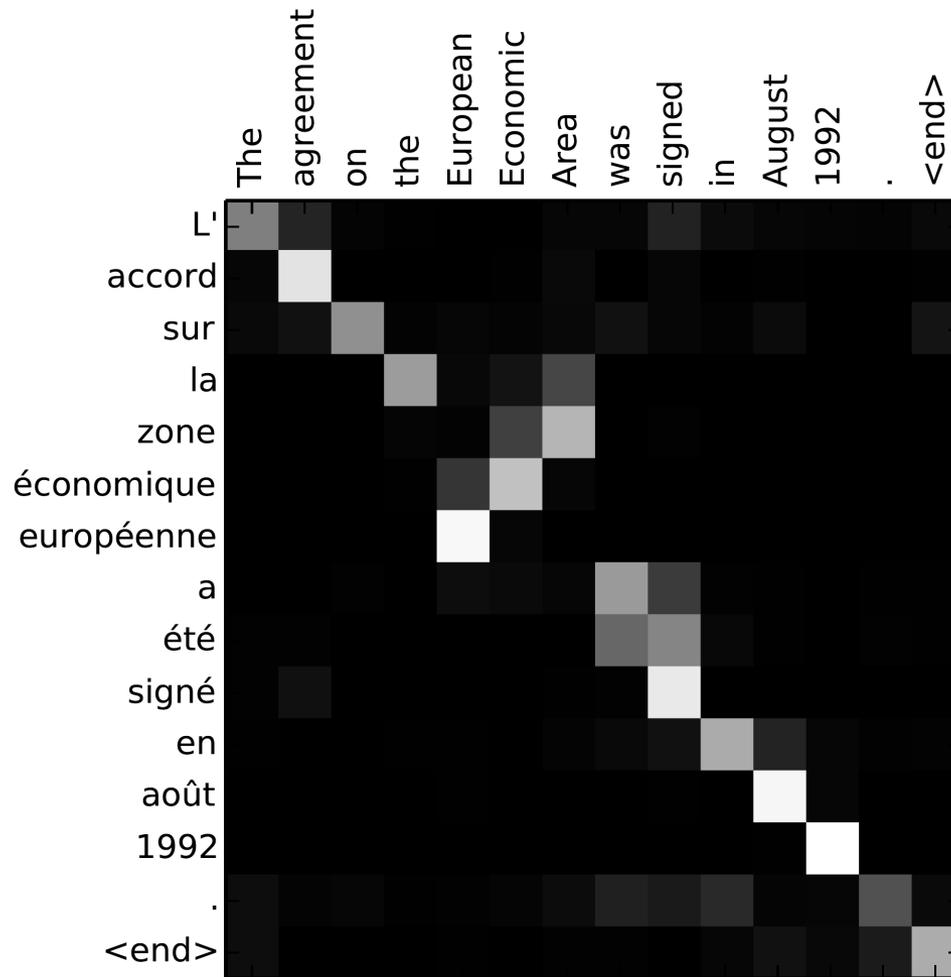
https://lenna-voita.github.io/nlp_course/seq2seq_and_attention.html

Attention Mechanism Improves Results



Neural Machine Translation by Jointly Learning to Align and Translate, Bahdanau et al 2014

Attention Mechanism Gives Insights



Neural Machine Translation by Jointly Learning to Align and Translate, Bahdanau et al 2014

References

- Machine Translation Tutorial ACL, 2016
- https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Thanks for listening!