UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

UNIVERSITÉ CÔTE D'AZUR

Master DSAI
Advanced Deep Learning
2024-2025

Local correlations to Transformers

Frederic Precioso

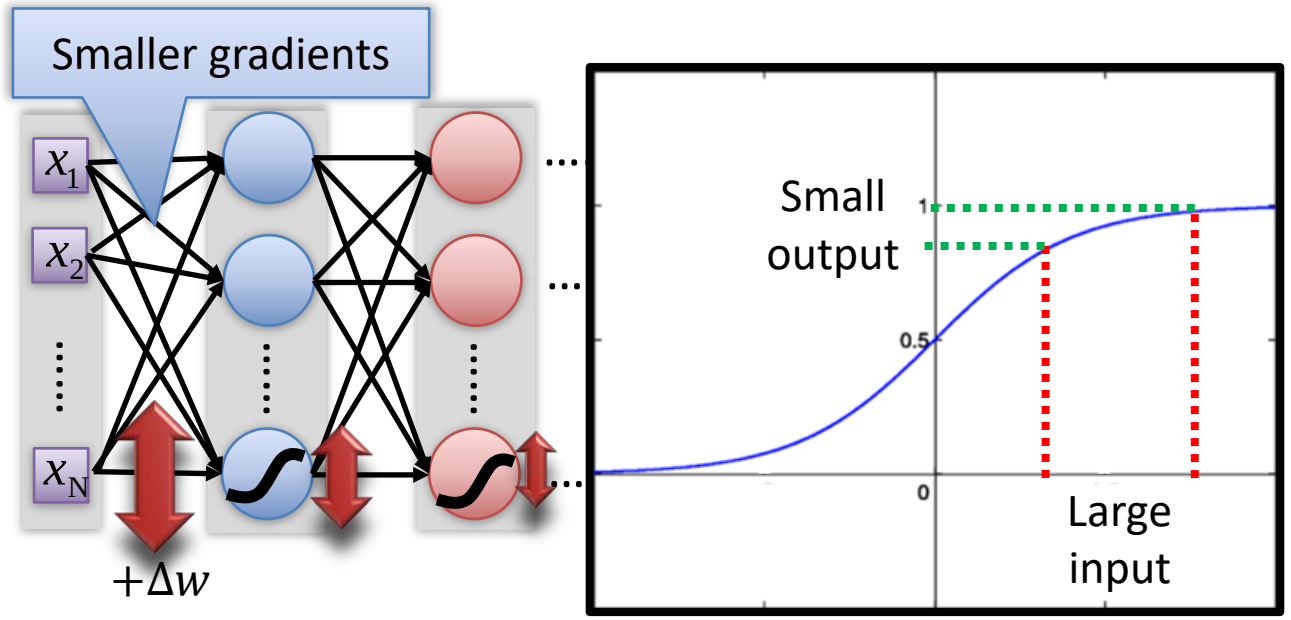
A large set of these slides come from the lecture of Prof. Adriana Kovashka University of Pittsburgh

univ-cotedazur.fr

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Vanishing gradient problem



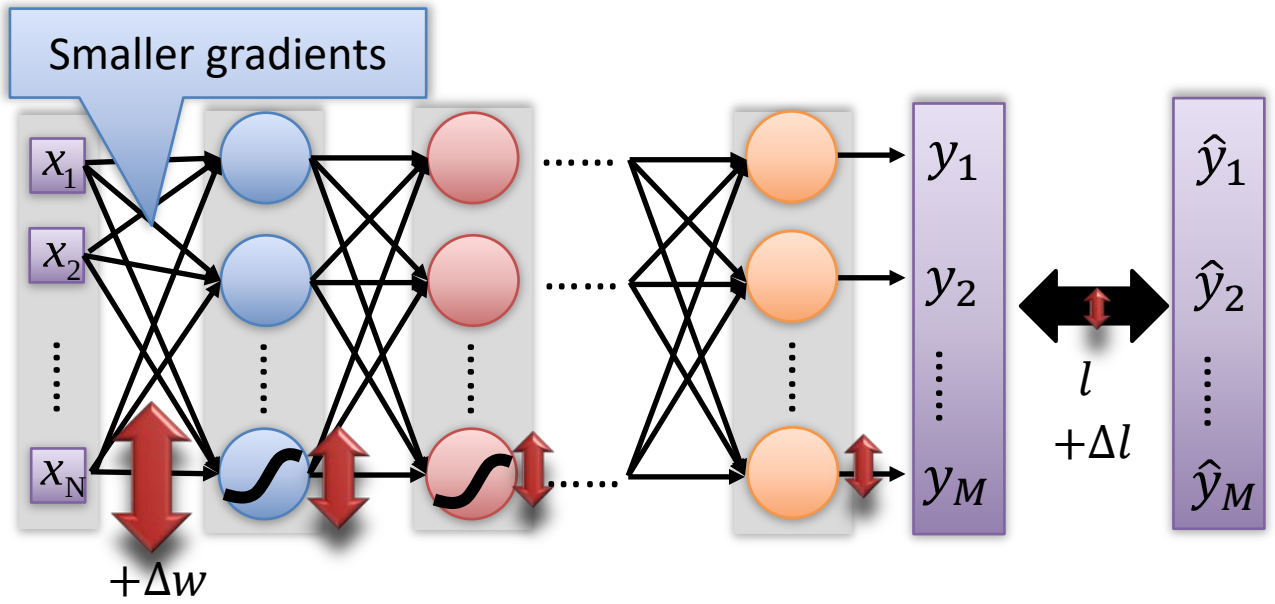
Intuitive way to compute the derivatives …

$$\frac{\partial l}{\partial w} =? \frac{\Delta l}{\Delta w}$$

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Vanishing gradient problem



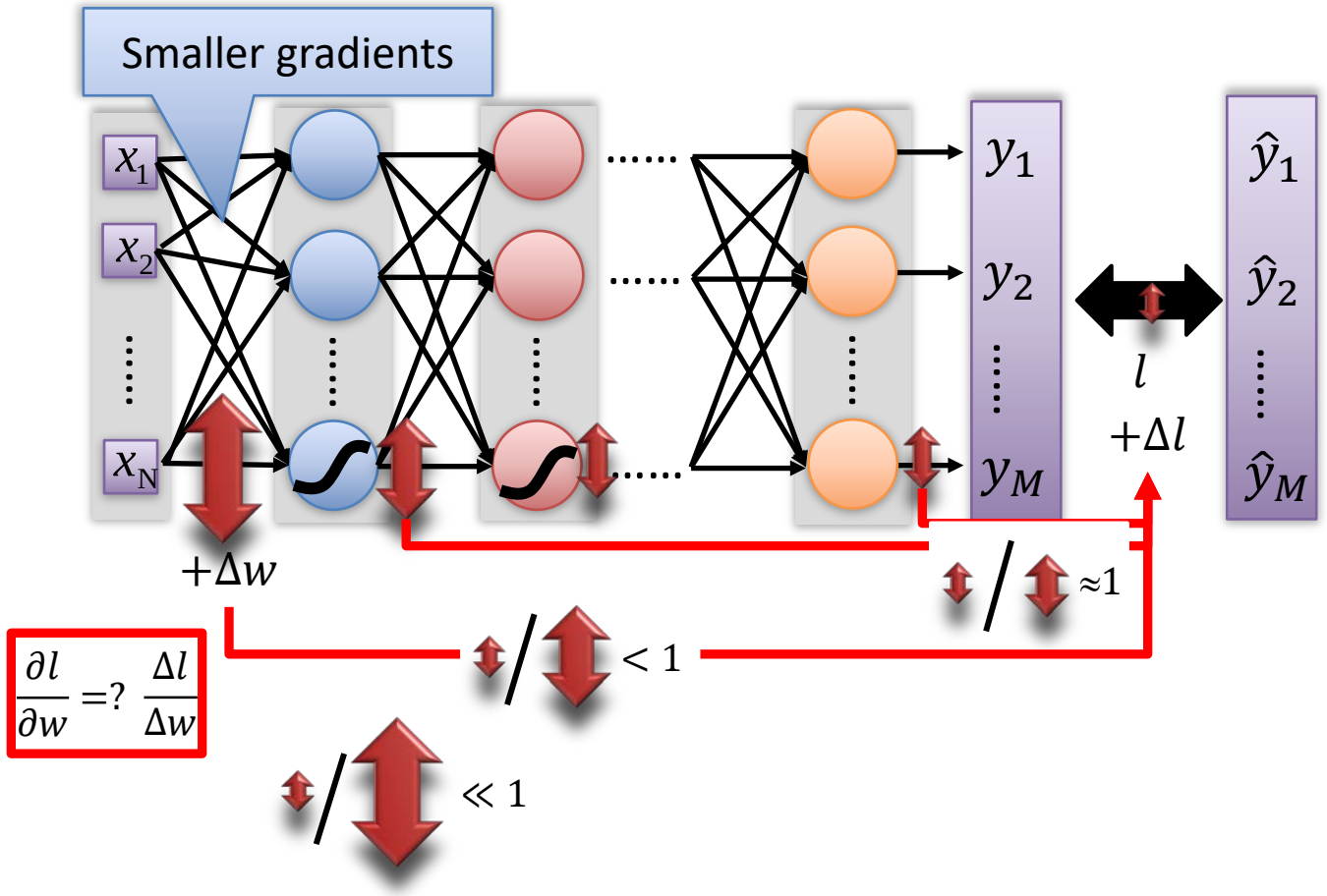
Intuitive way to compute the derivatives …

$$\frac{\partial l}{\partial w} =? \frac{\Delta l}{\Delta w}$$

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Vanishing gradient problem



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Sequential correlations

Recurrent Neural Networks (aka RNN, LSTM / GRU)

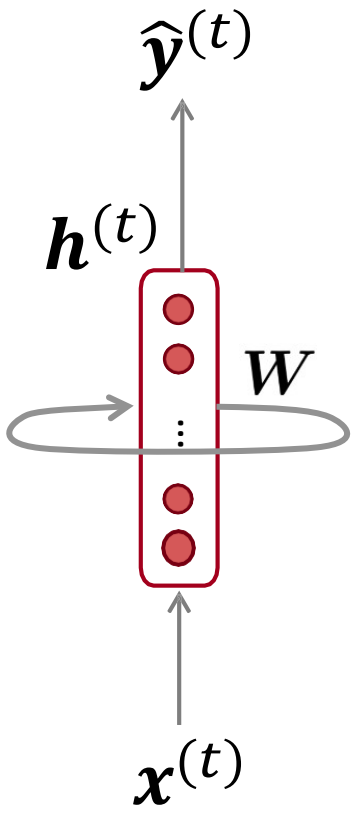
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Recurrent Neural Networks (RNN)

A family of neural architectures

Core idea: Apply the same weights W *repeatedly*



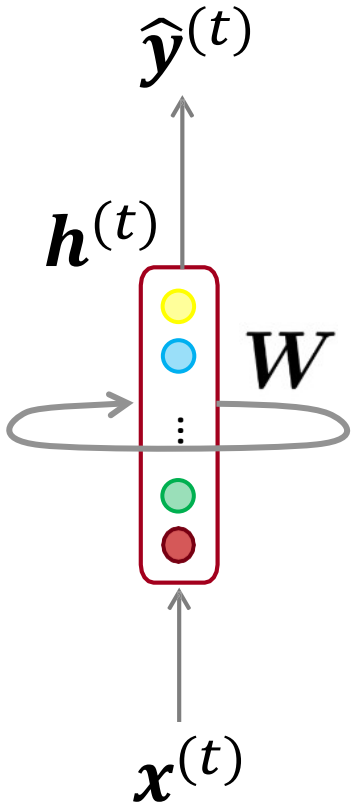
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Recurrent Neural Networks (RNN)

A family of neural architectures

Core idea: Apply the same weights W *repeatedly*



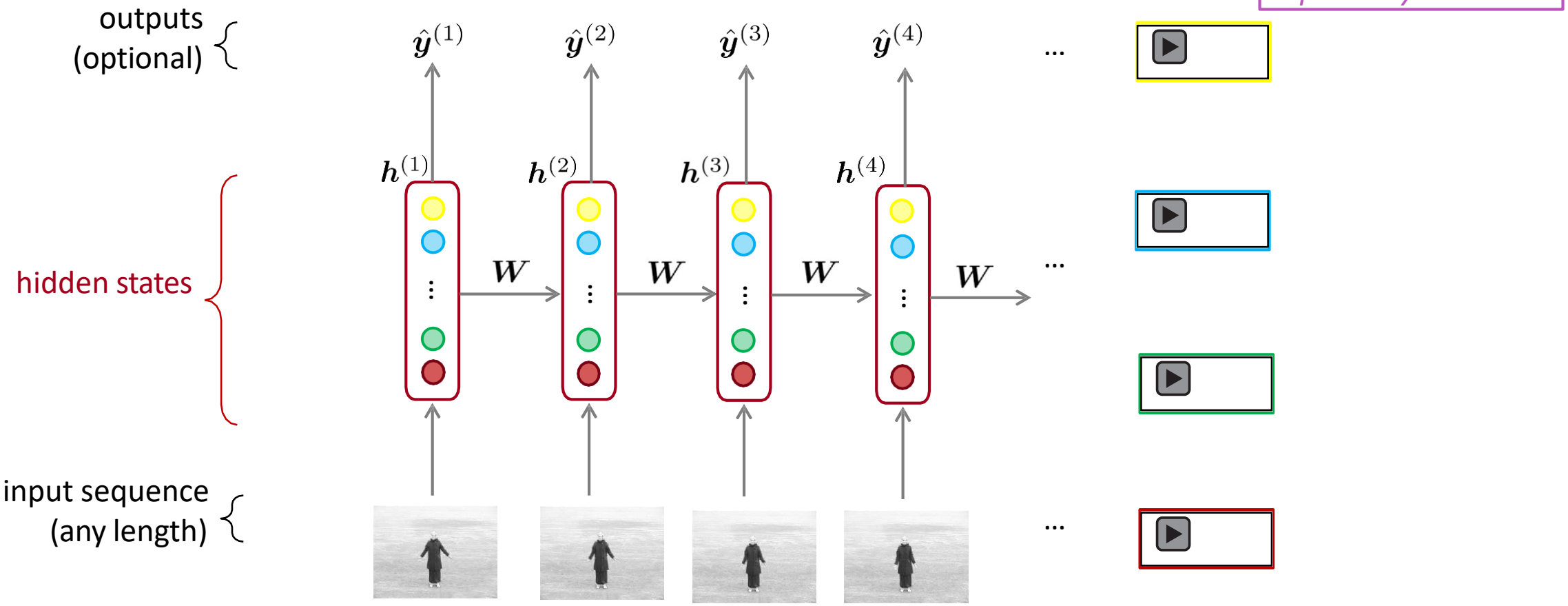
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Recurrent Neural Networks (RNN)

A family of neural architectures

Core idea: Apply the same weights W *repeatedly*



Abigail See

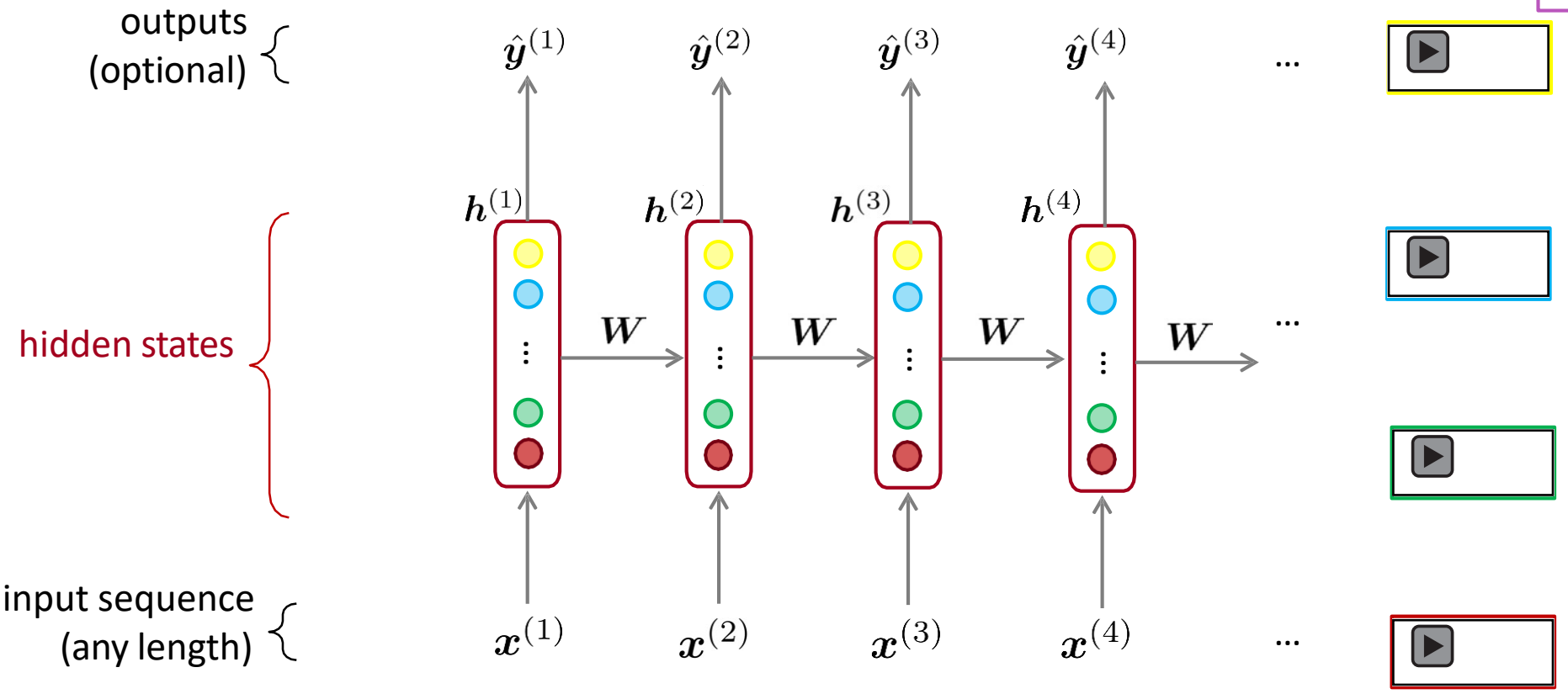
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Recurrent Neural Networks (RNN)

A family of neural architectures

Core idea: Apply the same weights $\boldsymbol{W}$ *repeatedly*



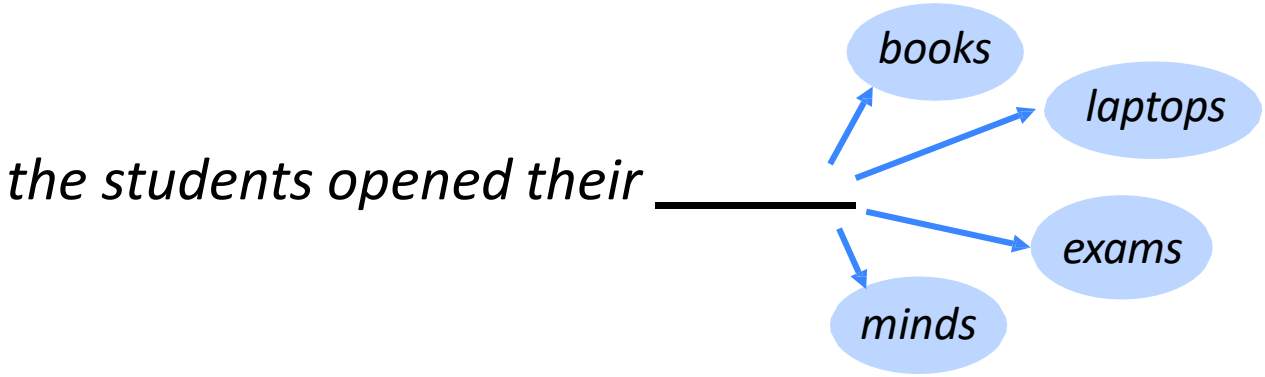
Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Language Modeling

- **Language Modeling** is the task of predicting what word comes next.



- More formally: given a sequence of words $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(t)}$, compute the probability distribution of the next word $\boldsymbol{x}^{(t+1)}$:

$$P(\boldsymbol{x}^{(t+1)} | \ \boldsymbol{x}^{(t)}, \ldots, \boldsymbol{x}^{(1)})$$

where $\boldsymbol{x}^{(t+1)}$ can be any word in the vocabulary $V = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{|V|}\}$

- A system that does this is called a **Language Model**.

Abigail See

n-gram Language Models

- First we make a simplifying assumption: $\boldsymbol{x}^{(t+1)}$ depends only on the preceding *n-1* words.

n-1 words

$$P(\boldsymbol{x}^{(t+1)}|\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(1)}) = P(\boldsymbol{x}^{(t+1)}|\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(t-n+2)}) \quad \text{(assumption)}$$

prob of a n-gram

prob of a (n-1)-gram

$$= \frac{P(\boldsymbol{x}^{(t+1)},\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(t-n+2)})}{P(\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(t-n+2)})} \quad \text{(definition of conditional prob)}$$

- **Question:** How do we get these *n*-gram and (*n*-1)-gram probabilities?
- **Answer:** By counting them in some large corpus of text!

$$\approx \frac{\text{count}(\boldsymbol{x}^{(t+1)},\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(t-n+2)})}{\text{count}(\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(t-n+2)})} \quad \text{(statistical approximation)}$$

Sparsity Problems with n-gram Language Models

Sparsity Problem 1

Problem: What if *"students opened their* $\boldsymbol{w}$*"* never occurred in data? Then $\boldsymbol{w}$ has probability 0!

(Partial) Solution: Add small δ to the count for every $\boldsymbol{w} \in V$. This is called *smoothing*.

$$P(\boldsymbol{w}|\text{students opened their}) = \frac{\text{count}(\text{students opened their } \boldsymbol{w})}{\text{count}(\text{students opened their})}$$

Sparsity Problem 2

Problem: What if *"students opened their"* never occurred in data? Then we can't calculate probability for *any* $\boldsymbol{w}$!

(Partial) Solution: Just condition on *"opened their"* instead. This is called *backoff*.

Note: Increasing *n* makes sparsity problems *worse.* Typically we can't have *n* bigger than 5.

A fixed-window neural Language Model

output distribution

$$\hat{\boldsymbol{y}} = \text{softmax}(\boldsymbol{U}\boldsymbol{h} + \boldsymbol{b}_2) \in \mathbb{R}^{|V|}$$

hidden layer

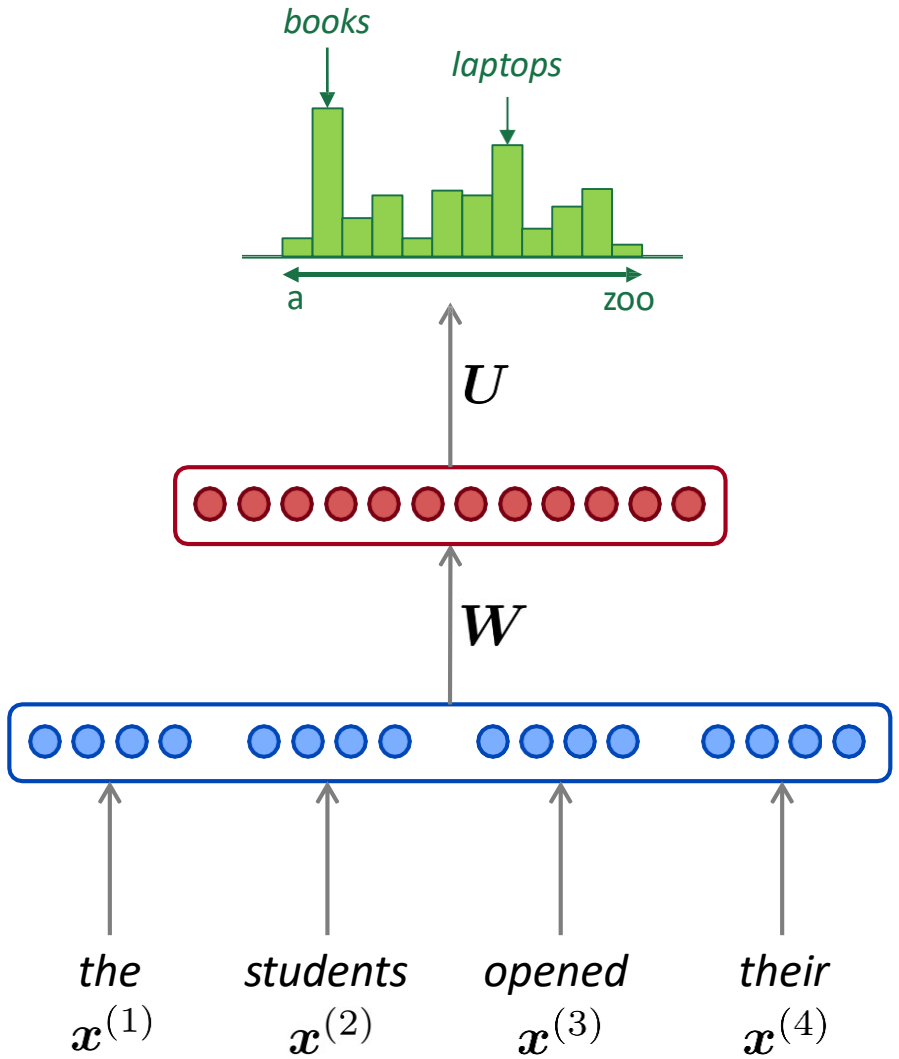
$$\boldsymbol{h} = f(\boldsymbol{W}\boldsymbol{e} + \boldsymbol{b}_1)$$

concatenated word embeddings

$$\boldsymbol{e} = [\boldsymbol{e}^{(1)}; \boldsymbol{e}^{(2)}; \boldsymbol{e}^{(3)}; \boldsymbol{e}^{(4)}]$$

words / one-hot vectors

$$\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \boldsymbol{x}^{(4)}$$



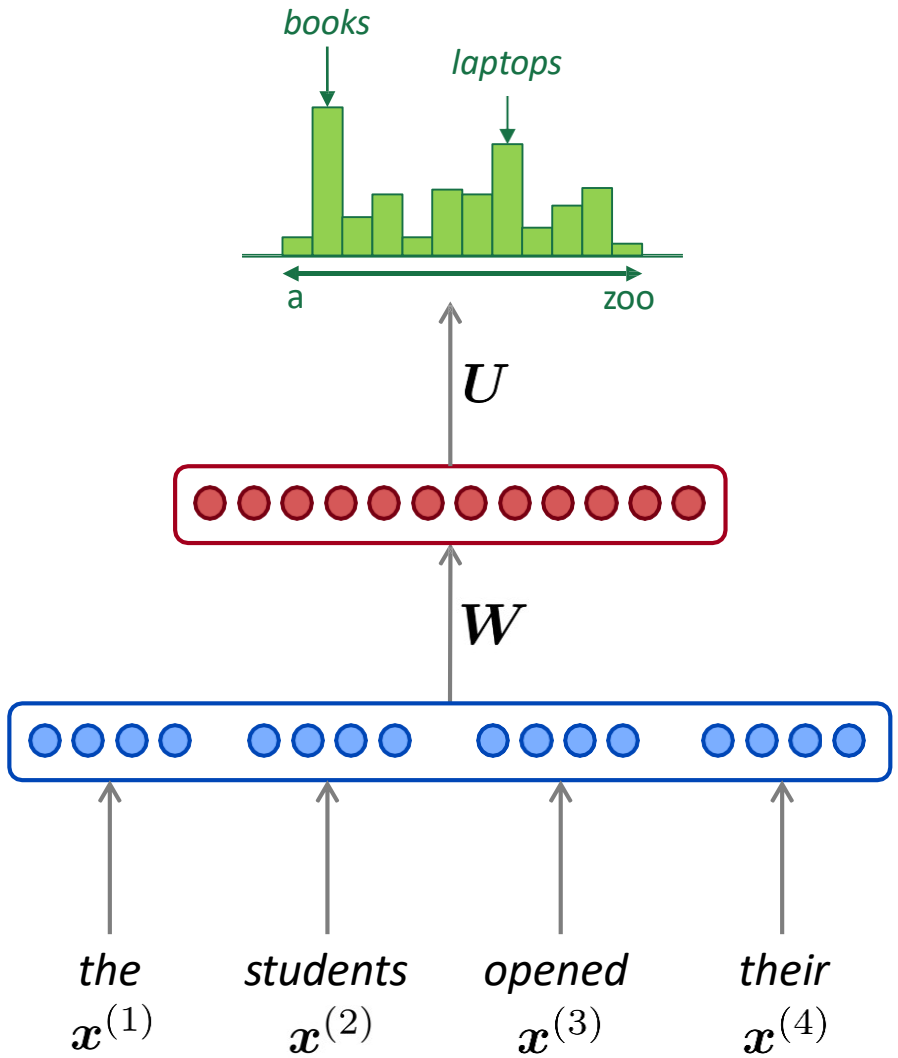
A fixed-window neural Language Model (as CNN)

Remaining **problems**:

- Fixed window is too small
- Enlarging window enlarges $\boldsymbol{W}$
- Window can never be large enough!

- $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ are multiplied by completely different weights in $\boldsymbol{W}$. No symmetry in how the inputs are processed.

We need a neural architecture that can process *any length input*



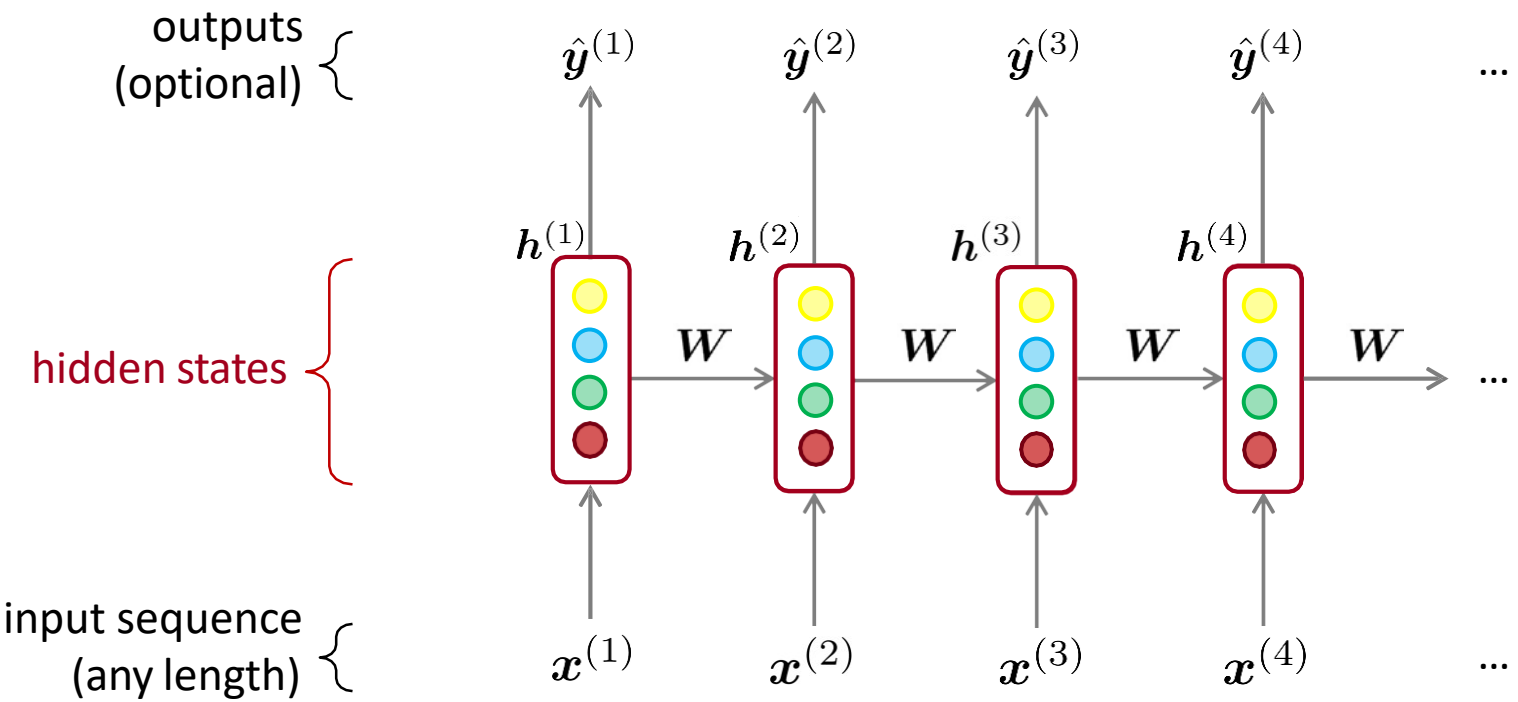
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Recurrent Neural Networks (RNN)

A family of neural architectures

Core idea: Apply the same weights $\boldsymbol{W}$ *repeatedly*

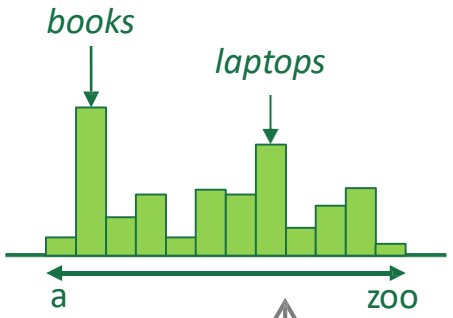


UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

A RNN Language Model

$\hat{\boldsymbol{y}}^{(4)} = P(\boldsymbol{x}^{(5)}|\text{the students opened their})$



output distribution

$$\hat{\boldsymbol{y}}^{(t)} = \text{softmax}\left(\boldsymbol{U}\boldsymbol{h}^{(t)} + \boldsymbol{b}_2\right) \in \mathbb{R}^{|V|}$$

hidden states

$$\boldsymbol{h}^{(t)} = \sigma\left(\boldsymbol{W}_h\boldsymbol{h}^{(t-1)} + \boldsymbol{W}_e\boldsymbol{e}^{(t)} + \boldsymbol{b}_1\right)$$

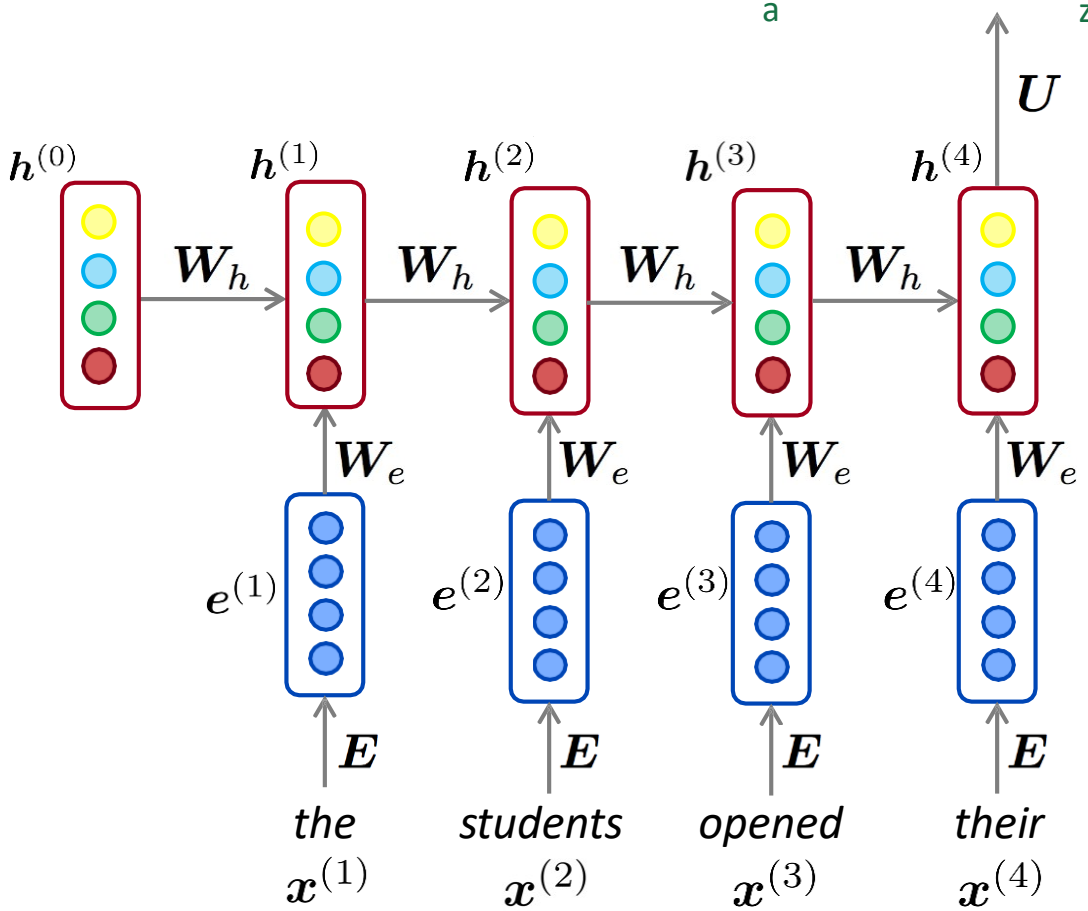
$\boldsymbol{h}^{(0)}$ is the initial hidden state

word embeddings

$$\boldsymbol{e}^{(t)} = \boldsymbol{E}\boldsymbol{x}^{(t)}$$

words / one-hot vectors

$$\boldsymbol{x}^{(t)} \in \mathbb{R}^{|V|}$$



***Note**: this input sequence could be much longer, but this slide doesn't have space!*

Abigail See

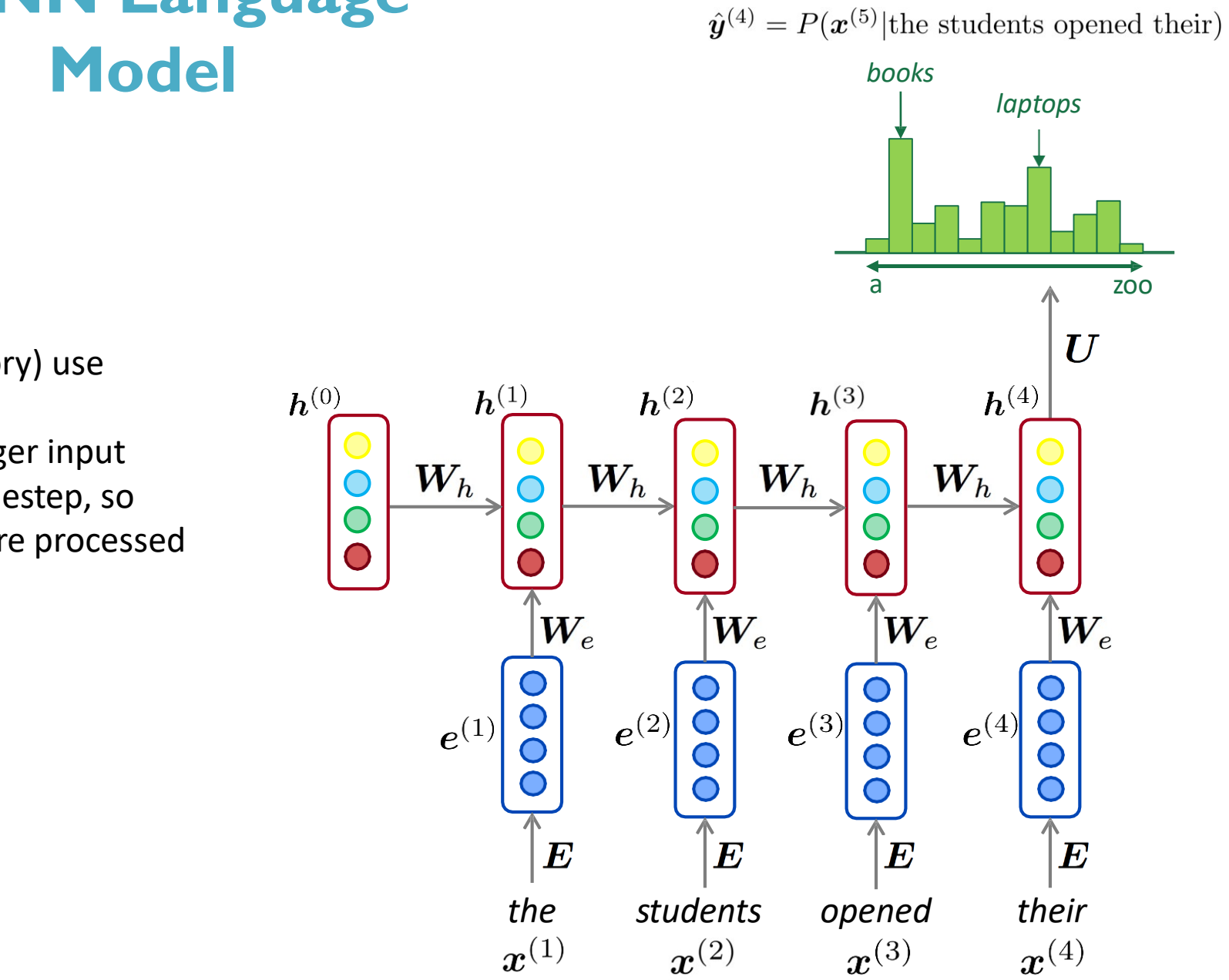
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

A RNN Language Model

RNN **Advantages**:

- Can process any length input
- Computation for step *t* can (in theory) use information from many steps back
- Model size doesn't increase for longer input
- Same weights applied on every timestep, so there is symmetry in how inputs are processed



Abigail See

Recall: Training a RNN Language Model

- Get a big corpus of text which is a sequence of words $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}$
- Feed into RNN-LM; compute output distribution $\hat{\boldsymbol{y}}^{(t)}$ for *every step t*.
  - i.e. predict probability distribution of *every word*, given words so far

- Loss function on step *t* is cross-entropy between predicted probability distribution $\hat{\boldsymbol{y}}^{(t)}$, and the true next word $\boldsymbol{y}^{(t)}$ (one-hot for $\boldsymbol{x}^{(t+1)}$):

$$J^{(t)}(\theta) = CE(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)}) = -\sum_{w \in V} \boldsymbol{y}_w^{(t)} \log \hat{\boldsymbol{y}}_w^{(t)} = -\log \hat{\boldsymbol{y}}_{\boldsymbol{x}_{t+1}}^{(t)}$$

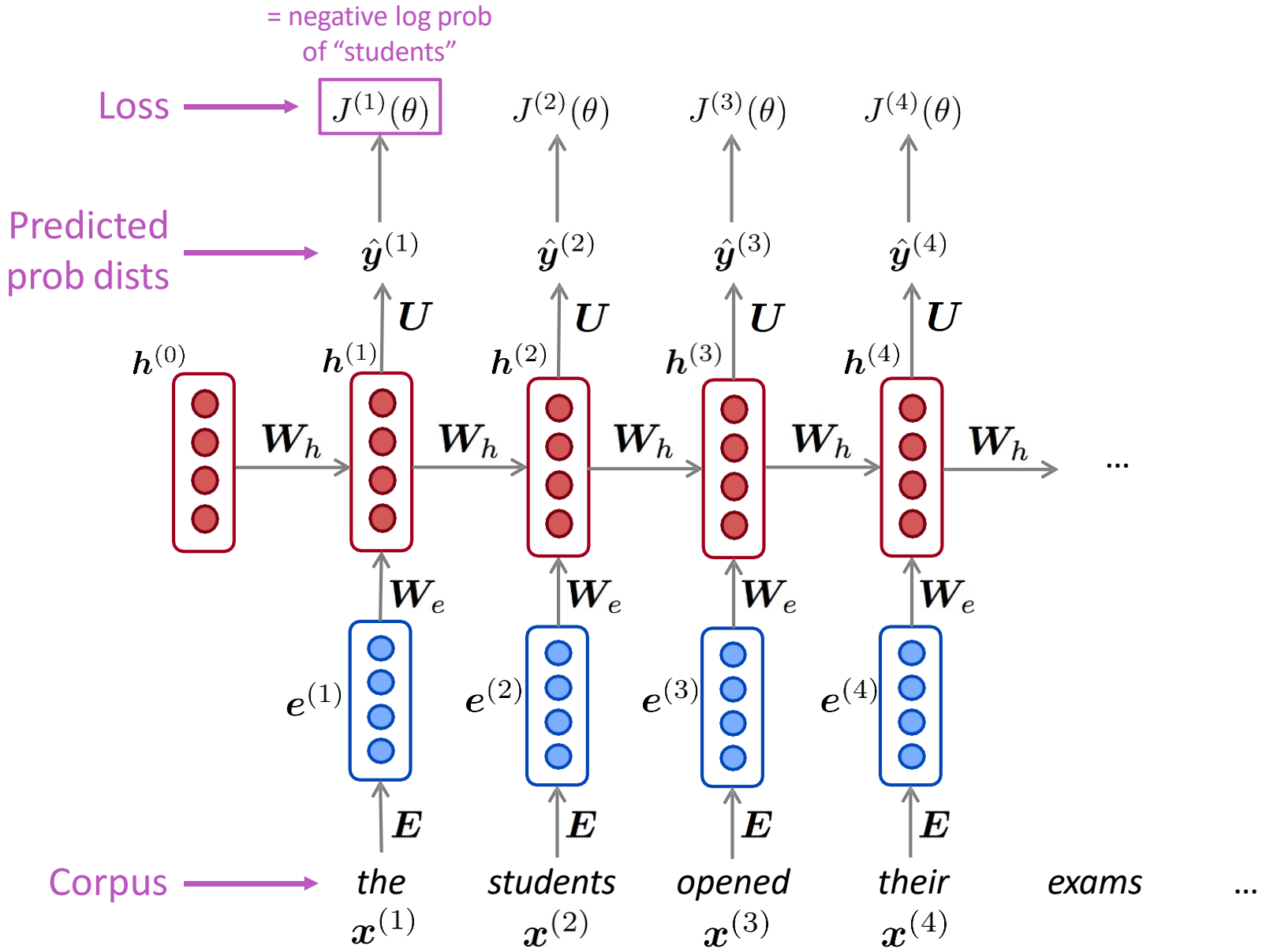
- Average this to get overall loss for entire training set:

$$J(\theta) = \frac{1}{T} \sum_{t=1}^{T} J^{(t)}(\theta) = \frac{1}{T} \sum_{t=1}^{T} -\log \hat{\boldsymbol{y}}_{\boldsymbol{x}_{t+1}}^{(t)}$$

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Training a RNN Language Model

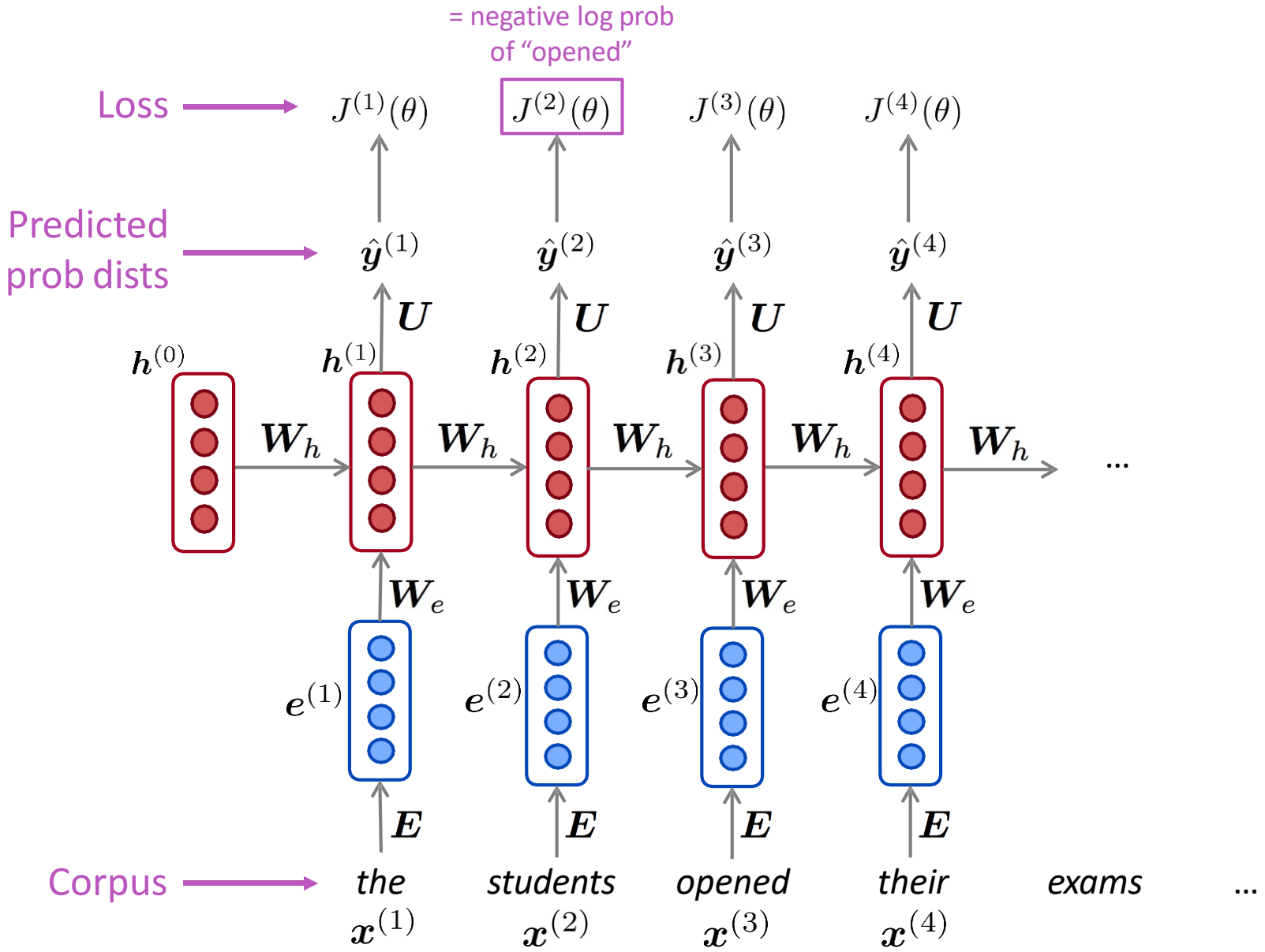


Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Training a RNN Language Model

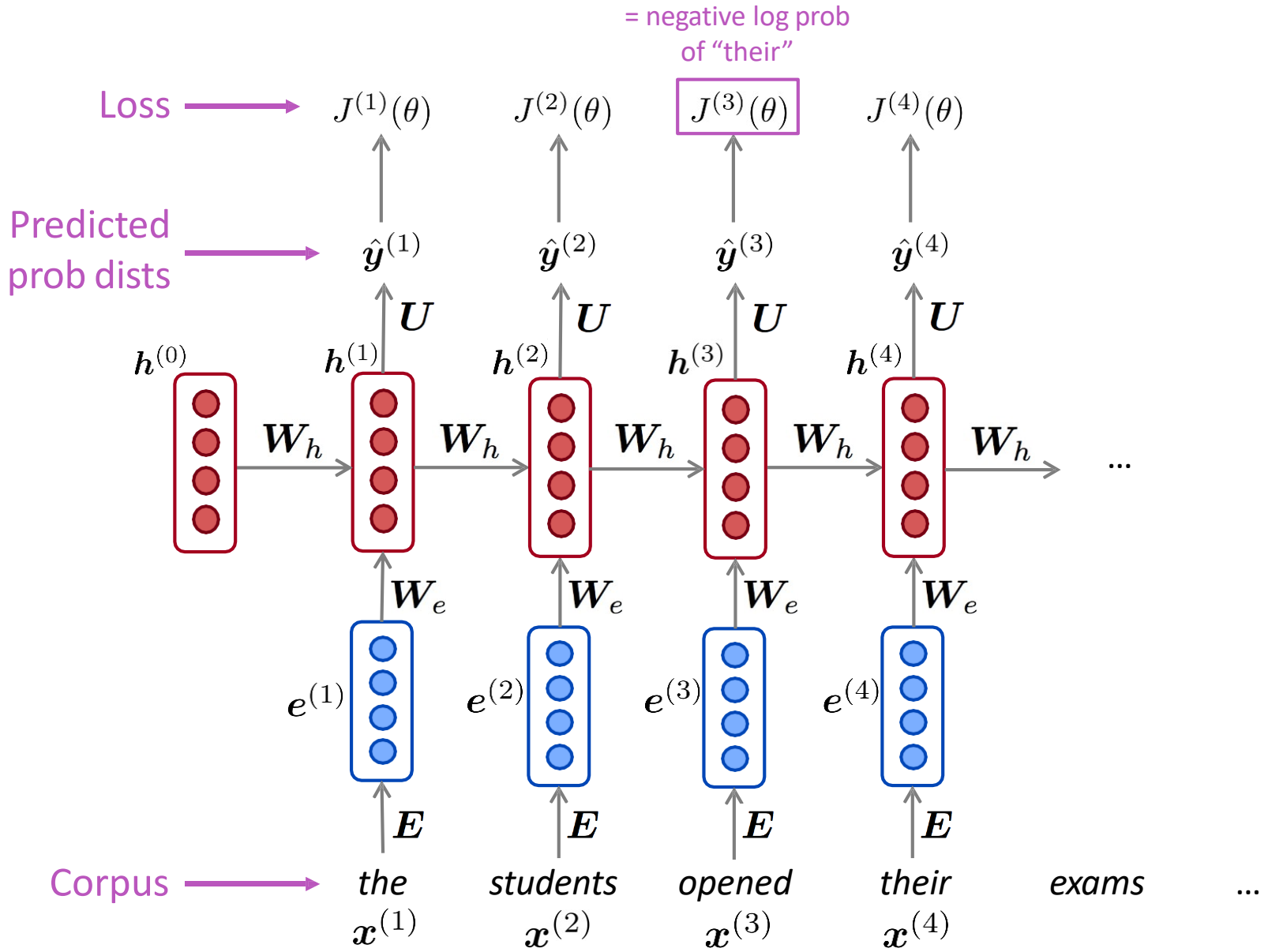


Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Training a RNN Language Model

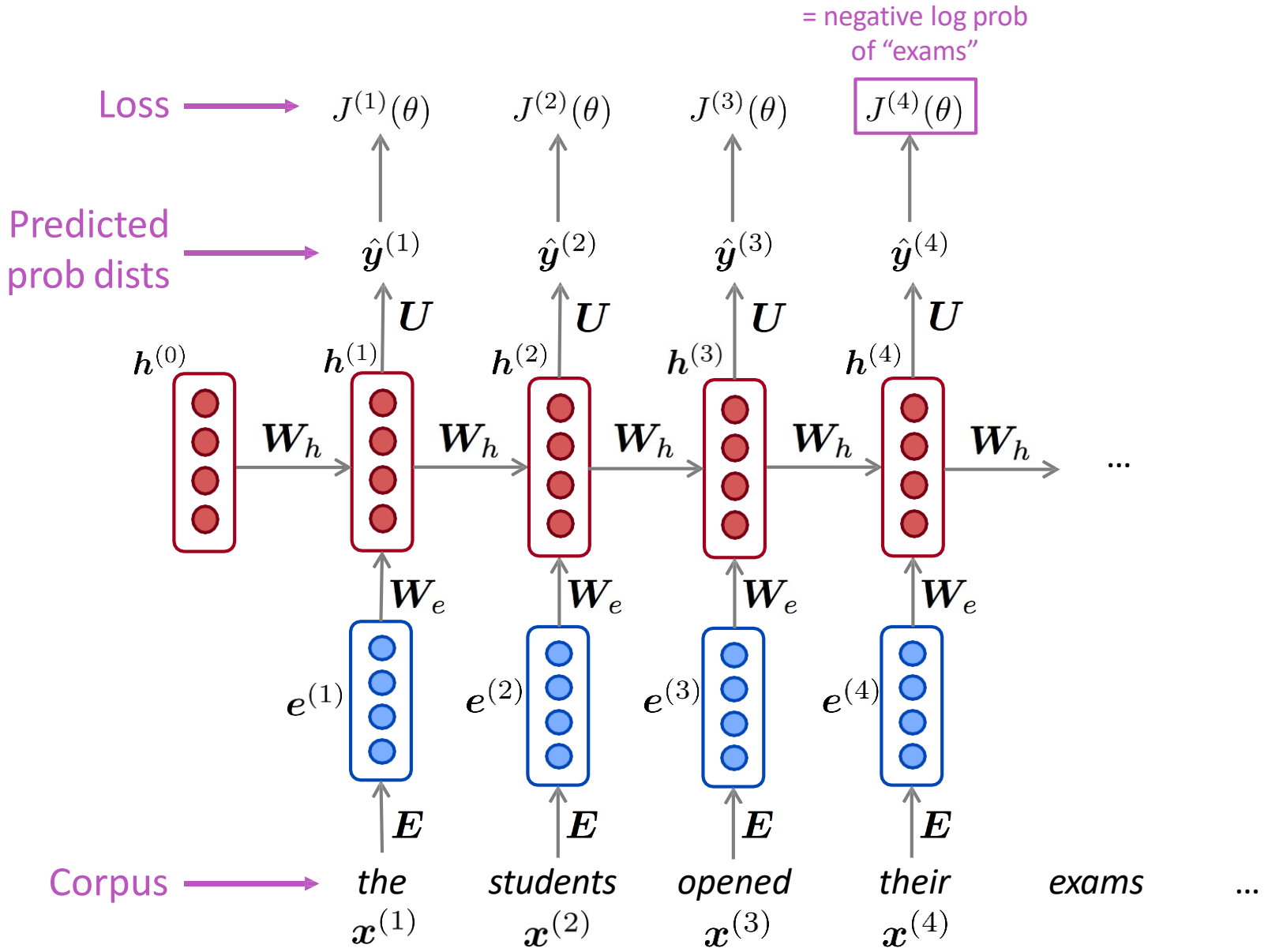


Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Training a RNN Language Model

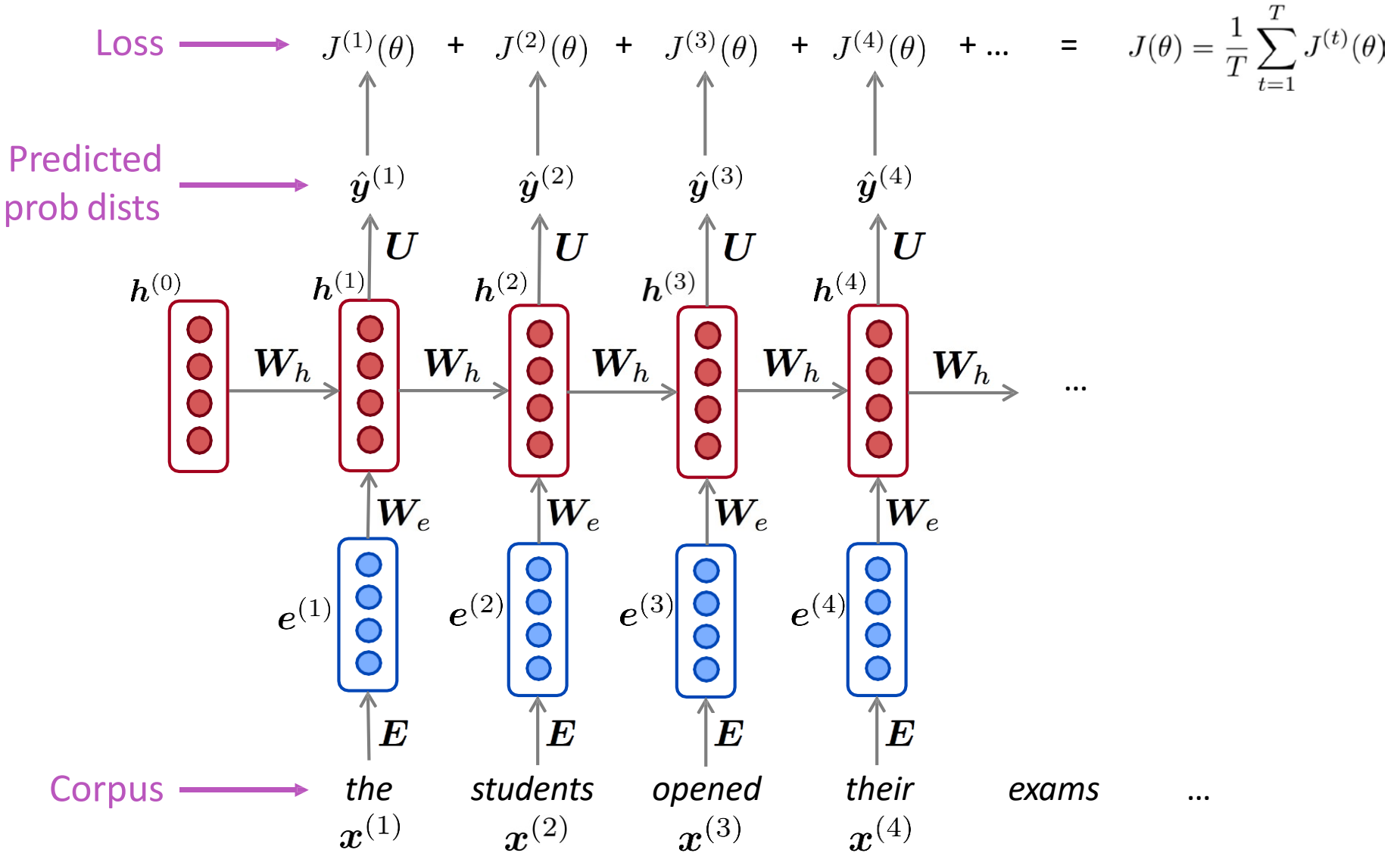


Abigail See

UNIVERSITÉ CÔTE D'AZUR

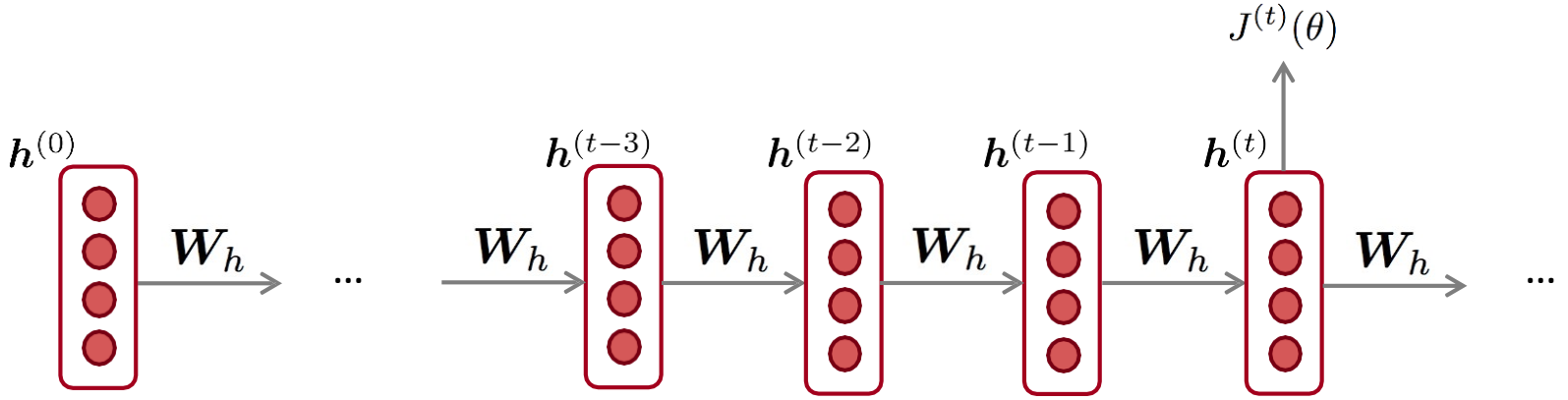
univ-cotedazur.fr

Training a RNN Language Model



Abigail See

Backpropagation for RNNs



Question: What's the derivative of $J^{(t)}(\theta)$ w.r.t. the repeated weight matrix $\boldsymbol{W_h}$?

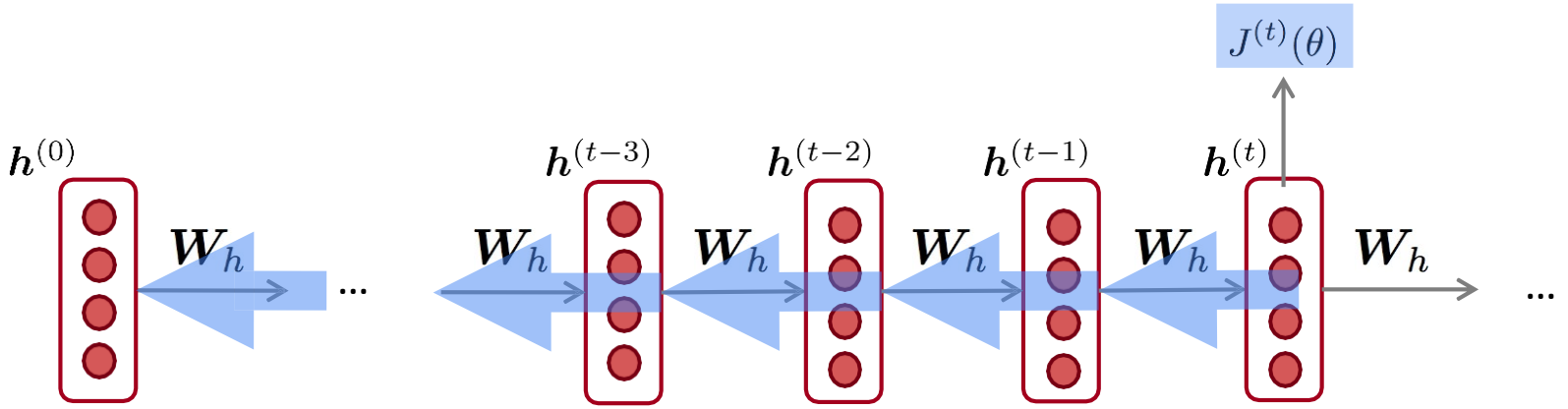
Answer: $$\frac{\partial J^{(t)}}{\partial \boldsymbol{W_h}} = \sum_{i=1}^{t} \left. \frac{\partial J^{(t)}}{\partial \boldsymbol{W_h}} \right|_{(i)}$$

"The gradient w.r.t. a repeated weight is the sum of the gradient w.r.t. each time it appears"

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Backpropagation for RNNs



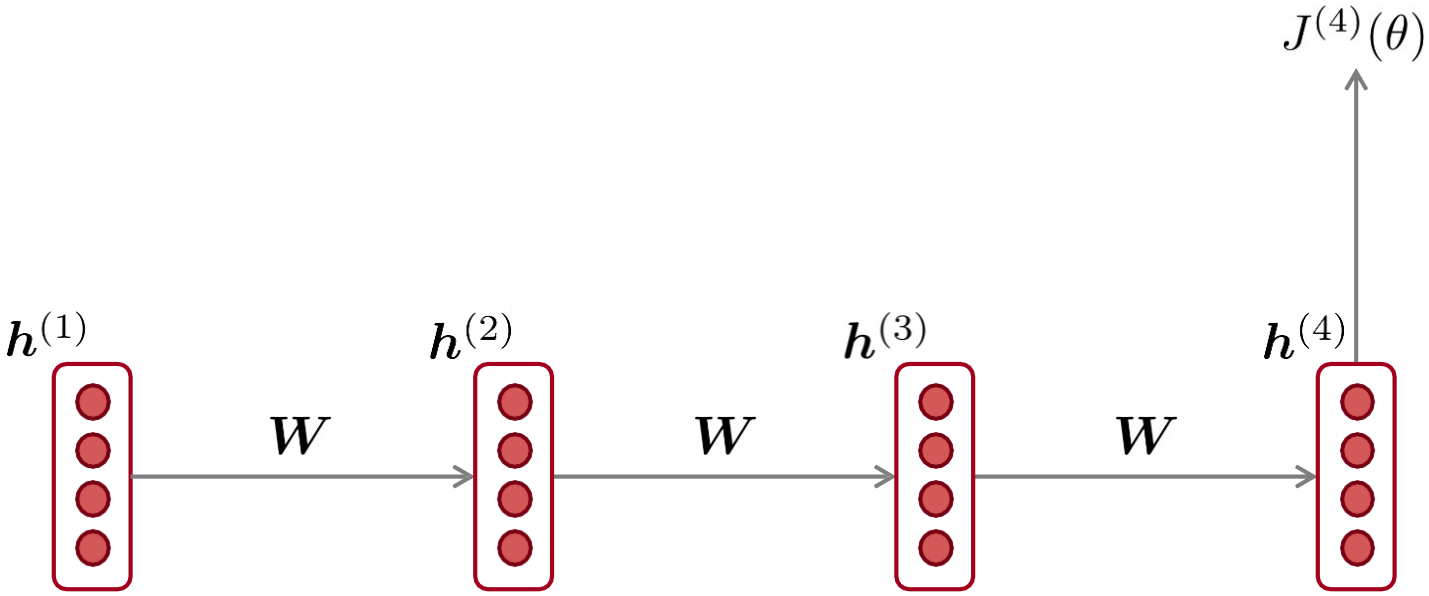
$$\frac{\partial J^{(t)}}{\partial \boldsymbol{W_h}} = \sum_{i=1}^{t} \left. \frac{\partial J^{(t)}}{\partial \boldsymbol{W_h}} \right|_{(i)}$$

Question: How do we calculate this?

Answer: Backpropagate over timesteps *i*=*t*,…,0, summing gradients as you go.
This algorithm is called **"backpropagation through time"**

Abigail See

Vanishing gradient intuition

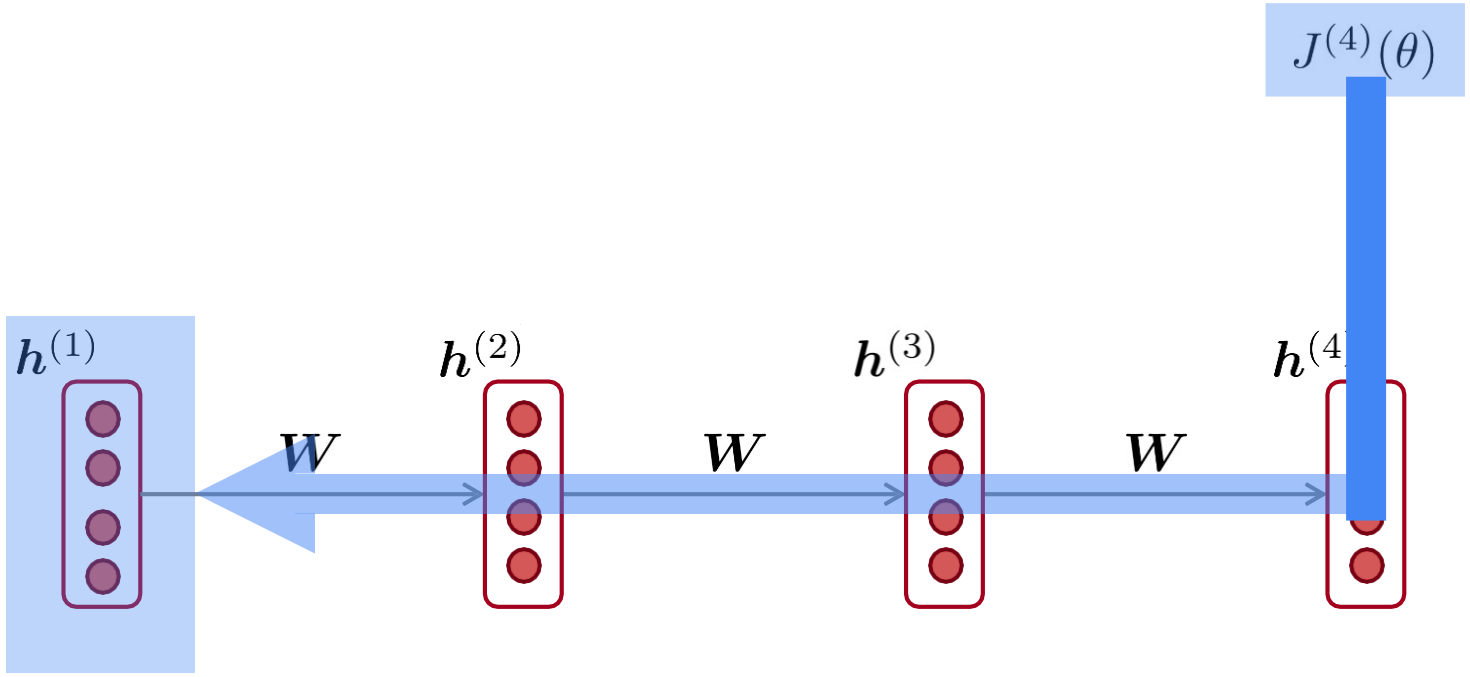


Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Vanishing gradient intuition



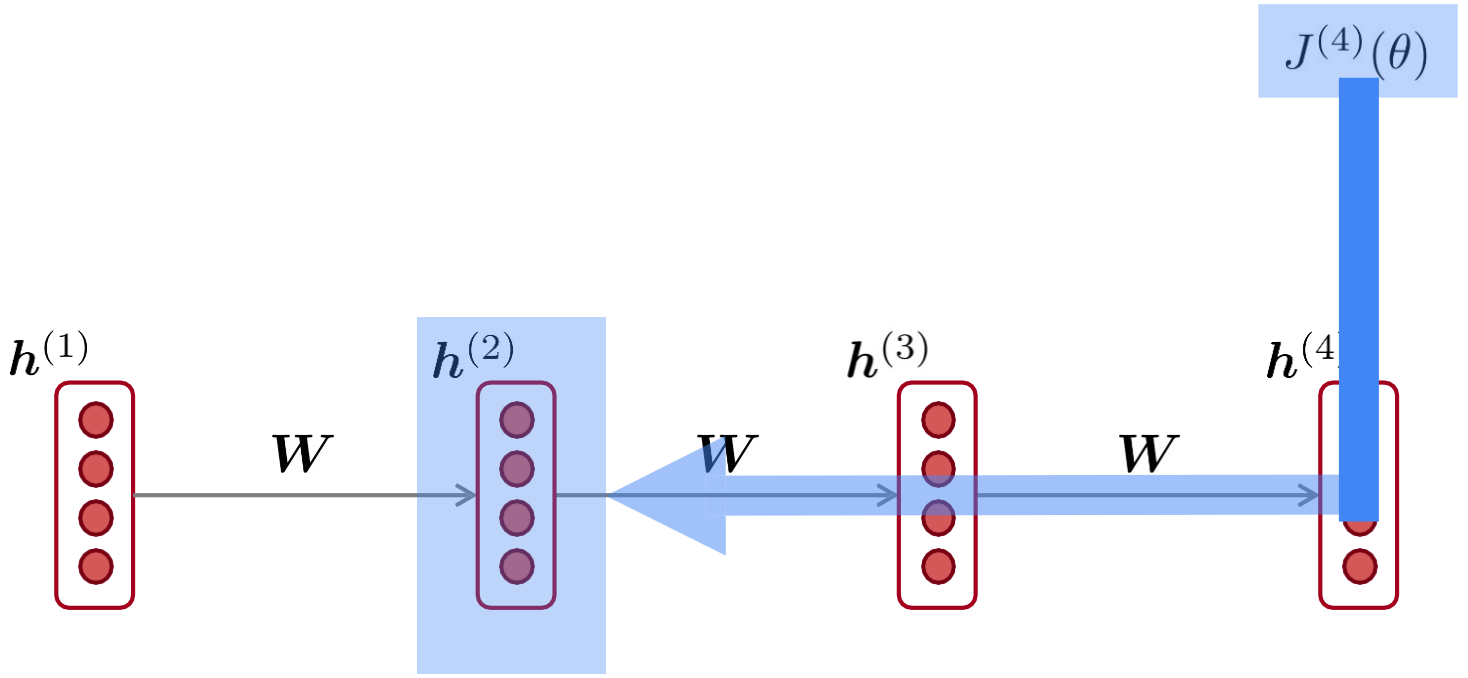
$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = ?$$

Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Vanishing gradient intuition



$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = \frac{\partial h^{(2)}}{\partial h^{(1)}} \times \frac{\partial J^{(4)}}{\partial h^{(2)}}$$

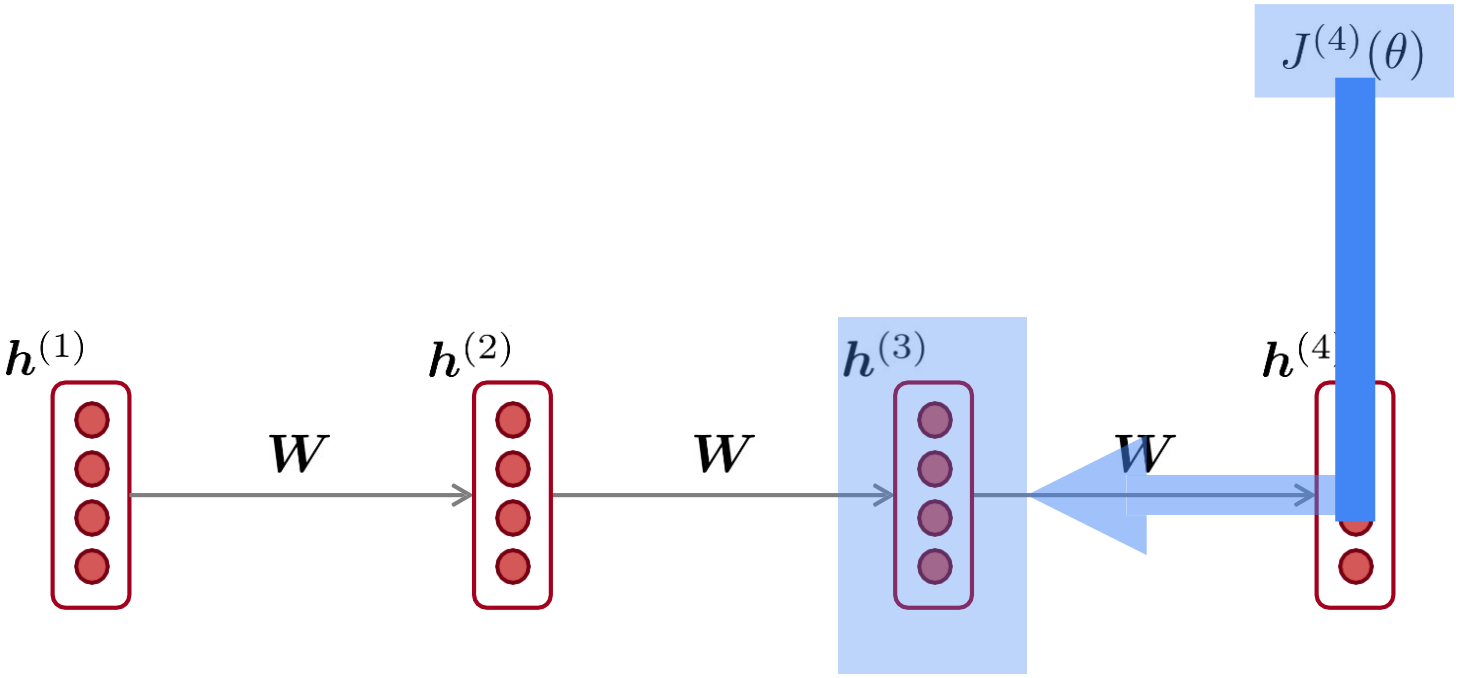
chain rule!

Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Vanishing gradient intuition



$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = \frac{\partial h^{(2)}}{\partial h^{(1)}} \times \frac{\partial h^{(3)}}{\partial h^{(2)}} \times \frac{\partial J^{(4)}}{\partial h^{(3)}}$$

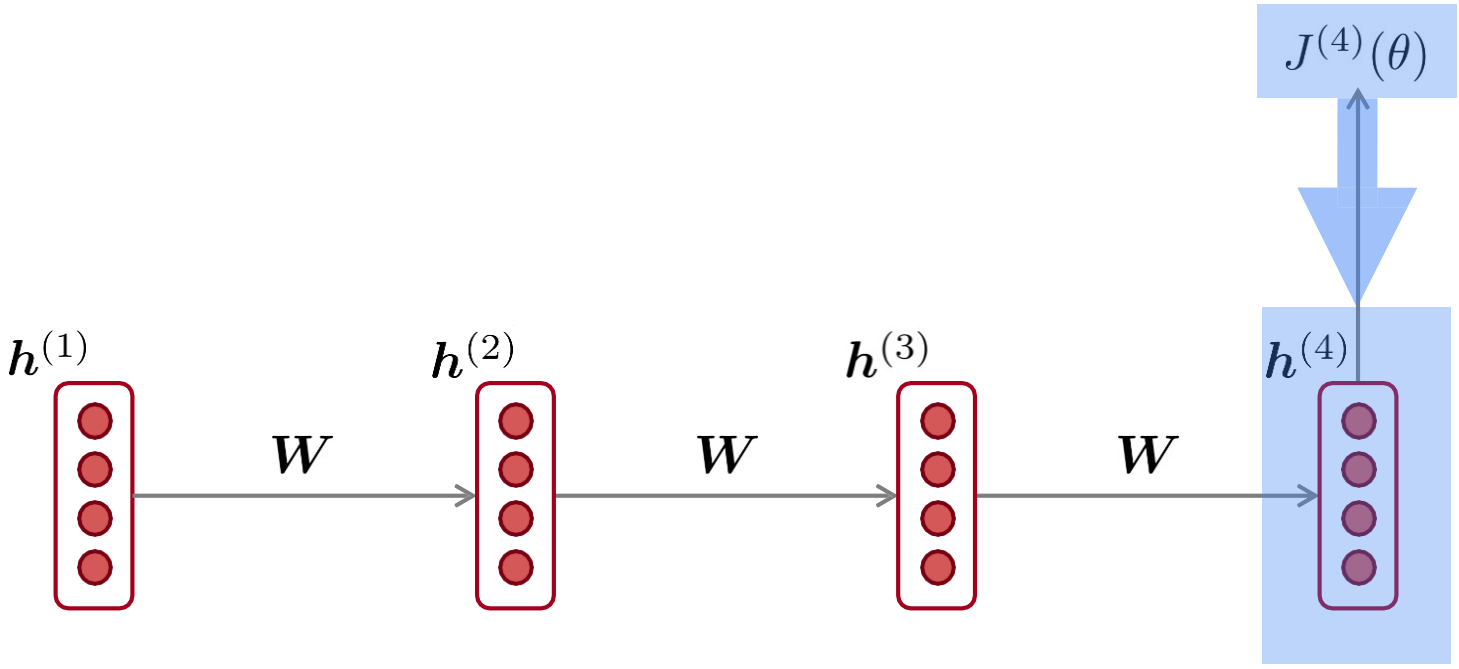
chain rule!

Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Vanishing gradient intuition



$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = \frac{\partial h^{(2)}}{\partial h^{(1)}} \times \frac{\partial h^{(3)}}{\partial h^{(2)}} \times \frac{\partial h^{(4)}}{\partial h^{(3)}} \times \frac{\partial J^{(4)}}{\partial h^{(4)}}$$

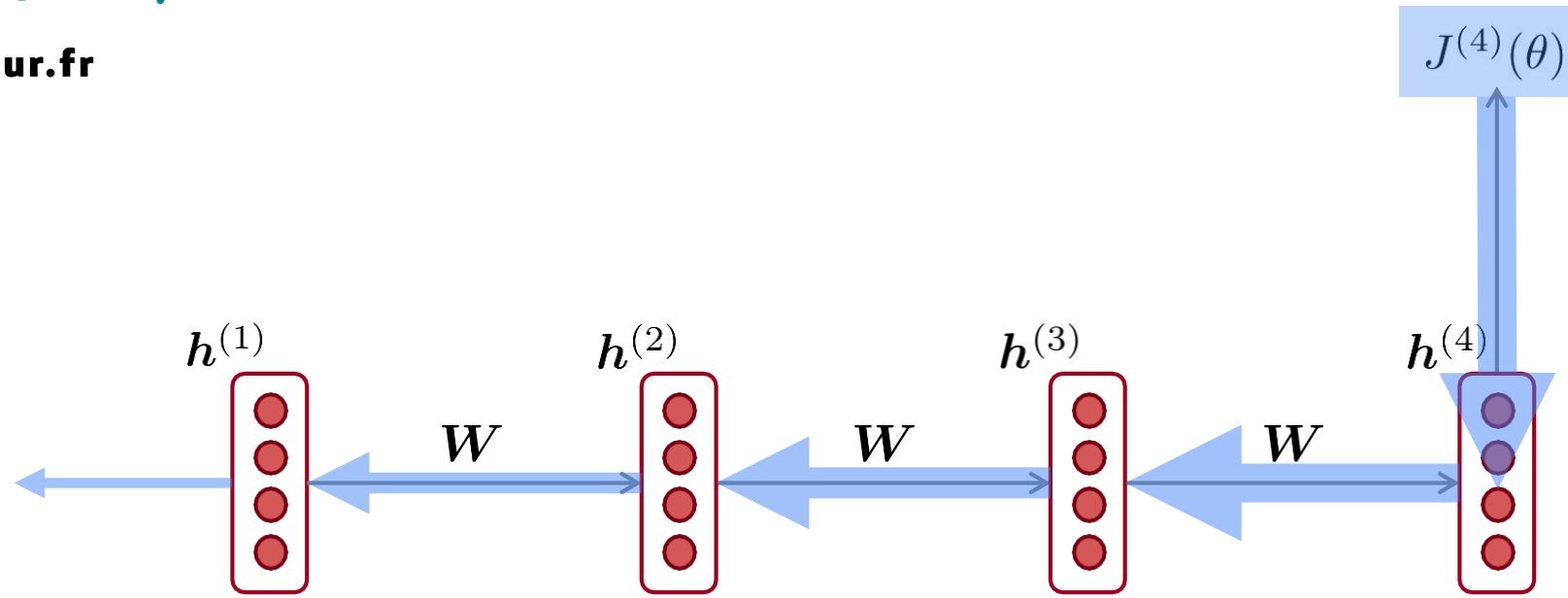
chain rule!

Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Vanishing gradient intuition



$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = \frac{\partial h^{(2)}}{\partial h^{(1)}} \times \frac{\partial h^{(3)}}{\partial h^{(2)}} \times \frac{\partial h^{(4)}}{\partial h^{(3)}} \times \frac{\partial J^{(4)}}{\partial h^{(4)}}$$

What happens if these are small?

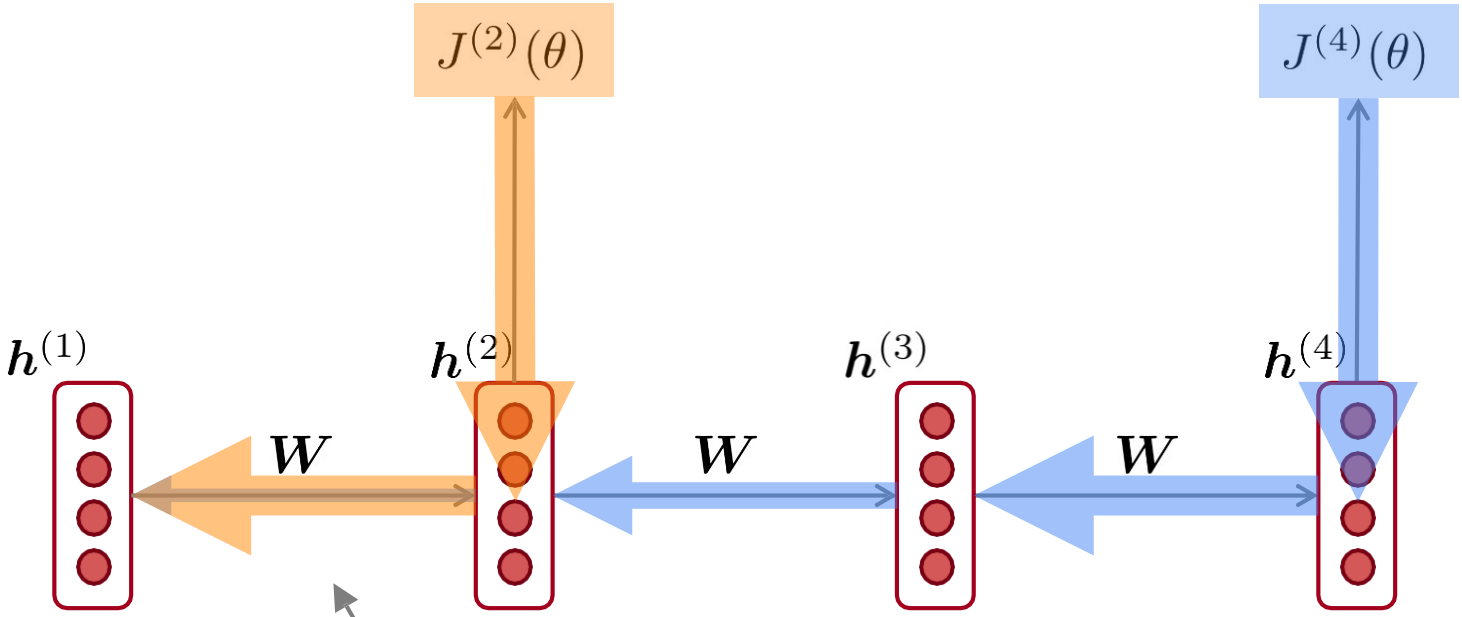
Vanishing gradient problem: When these are small, the gradient signal gets smaller and smaller as it backpropagates further

Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Why is vanishing gradient a problem?



Gradient signal from faraway is lost because it's much smaller than gradient signal from close-by.

So model weights are only updated only with respect to near effects, not long-term effects.

Effect of vanishing gradient on RNN-LM

- **LM task:** *When she tried to print her _____, she found that the printer was out of toner. She went to the stationery store to buy more toner. It was very overpriced. After installing the toner into the printer, she finally printed her tickets.*

- To learn from this training example, the RNN-LM needs to model the dependency between *"tickets"* on the 7th step and the target word *"tickets"* at the end.

- But if gradient is small, the model can't learn this dependency
  - So the model is unable to predict similar long-distance dependencies at test time

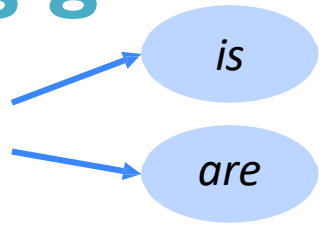
Adapted from Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Effect of vanishing gradient on RNN-LM

- **LM task:** *The writer of the books ____*



- **Correct answer**: *The writer of the books <u>is</u> planning a sequel*

- **Syntactic recency:** *The <u>writer</u> of the books <u>is</u>* (correct)

- **Sequential recency:** *The writer of the <u>books</u> <u>are</u>* (incorrect)

- Due to vanishing gradient, RNN-LMs are better at learning from sequential recency than syntactic recency, so they make this type of error more often than we'd like [Linzen et al 2016]

"Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies", Linzen et al, 2016. https://arxiv.org/pdf/1611.01368.pdf

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Why is exploding gradient a problem?

- If the gradient becomes too big, then the SGD update step becomes too big:

$$\theta^{new} = \theta^{old} - \overbrace{\alpha}^{\text{learning rate}} \underbrace{\nabla_\theta J(\theta)}_{\text{gradient}}$$

- This can cause bad updates: we take too large a step and reach a bad parameter configuration (with large loss)

- In the worst case, this will result in Inf or NaN in your network (then you have to restart training from an earlier checkpoint)

Gradient clipping: solution for exploding gradient

- Gradient clipping: if the norm of the gradient is greater than some threshold, scale it down before applying SGD update

Algorithm 1 Pseudo-code for norm clipping

$\hat{\mathbf{g}} \leftarrow \frac{\partial \mathcal{E}}{\partial \theta}$
if $\|\hat{\mathbf{g}}\| \geq threshold$ **then**
  $\hat{\mathbf{g}} \leftarrow \frac{threshold}{\|\hat{\mathbf{g}}\|}\hat{\mathbf{g}}$
end if

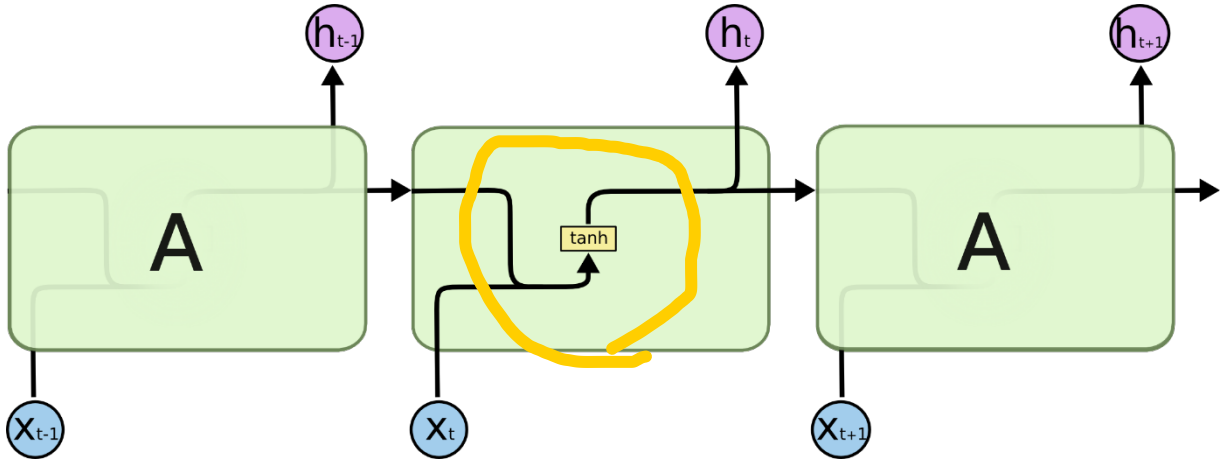
- Intuition: take a step in the same direction, but a smaller step

Source: "On the difficulty of training recurrent neural networks", Pascanu et al, 2013. http://proceedings.mlr.press/v28/pascanu13.pdf

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The repeating module in a standard RNN contains a single layer.



Unfortunately, as that gap grows, RNNs become unable to learn to connect the information (cf. vanishing gradients)

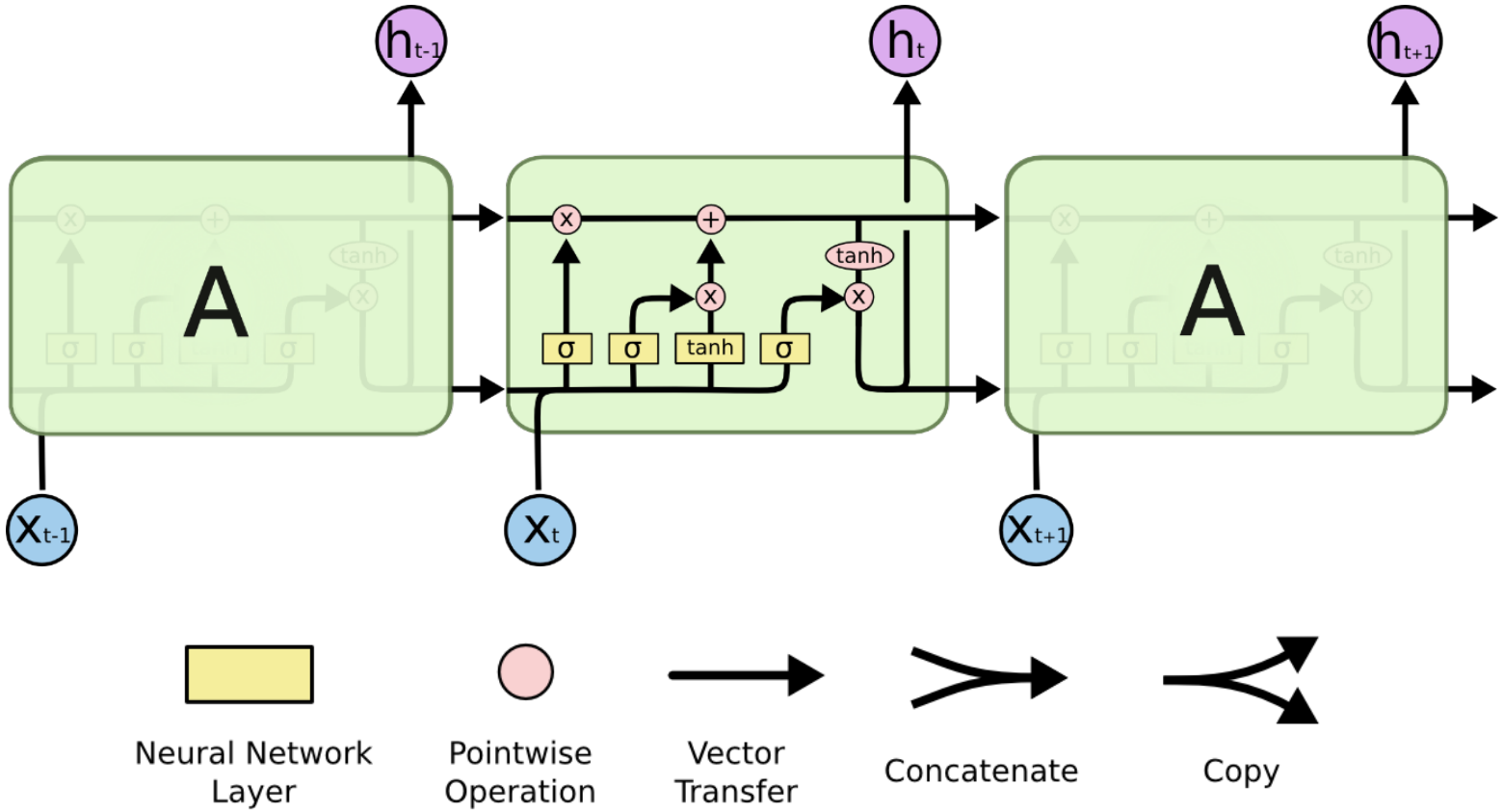
The problem was explored in depth by Hochreiter (1991) and Bengio, et al. (1994), who found some pretty fundamental reasons why it might be difficult.

Thankfully, LSTMs do not have this problem! (Hochreiter & Schmidhuber, 1997)

UNIVERSITÉ CÔTE D'AZUR

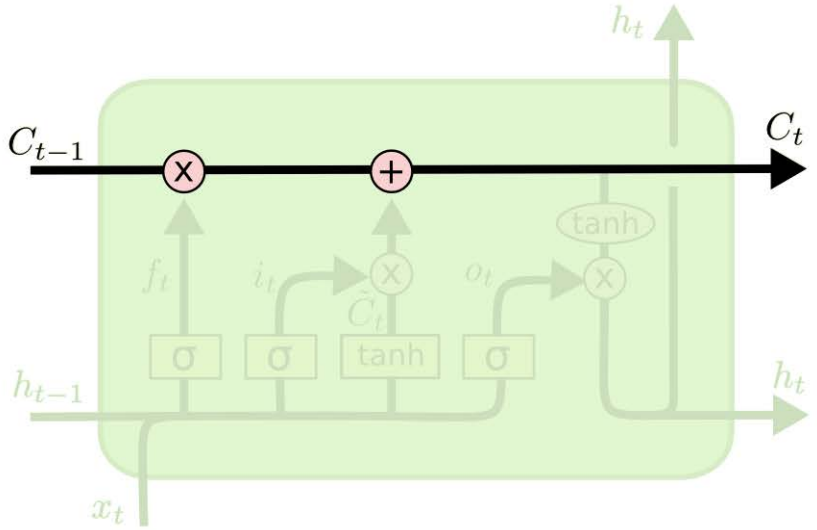
univ-cotedazur.fr

The repeating module in an LSTM contains four interacting layers.



The Core Idea Behind LSTMs

The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to just flow along it unchanged.



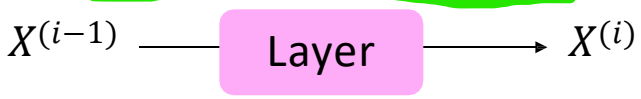
The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates.

UNIVERSITÉ CÔTE D'AZUR

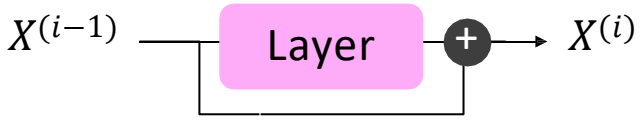
univ-cotedazur.fr

Residual connections [He et al., 2016]

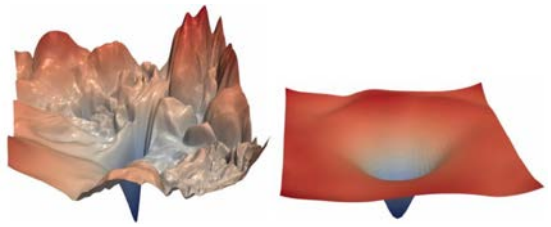
- **Residual connections** are a trick to help models train better.
  - Instead of $X^{(i)} = \text{Layer}(X^{(i-1)})$ (where i represents the layer)



  - We let $X^{(i)} = X^{(i-1)} + \text{Layer}(X^{(i-1)})$ (so we only have to learn "the residual" from the previous layer)



  - Residual connections are thought to make the loss landscape considerably smoother (thus easier training!)



[no residuals] [residuals]

[Loss landscape visualization, Li et al., 2018, on a ResNet]

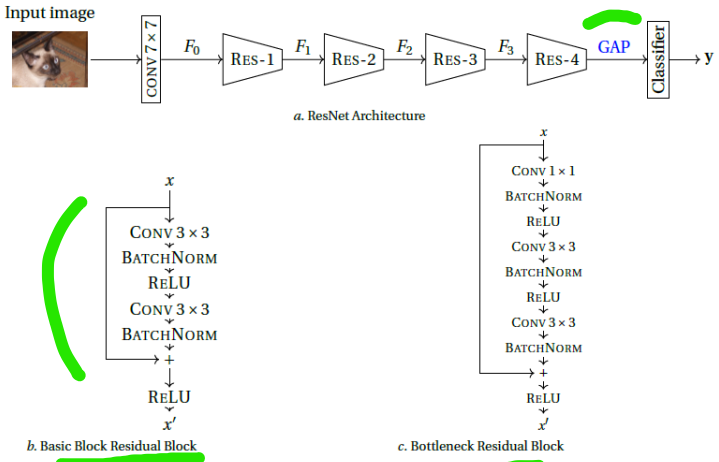


Figure 1.5: **Generalities** of the ResNet architecture.

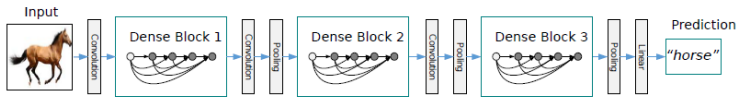


Figure 1.6: **Illustration of the DenseNet architecture** (G. Huang et al. 2017)

Examples with CNNs: ResNet, DenseNet

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

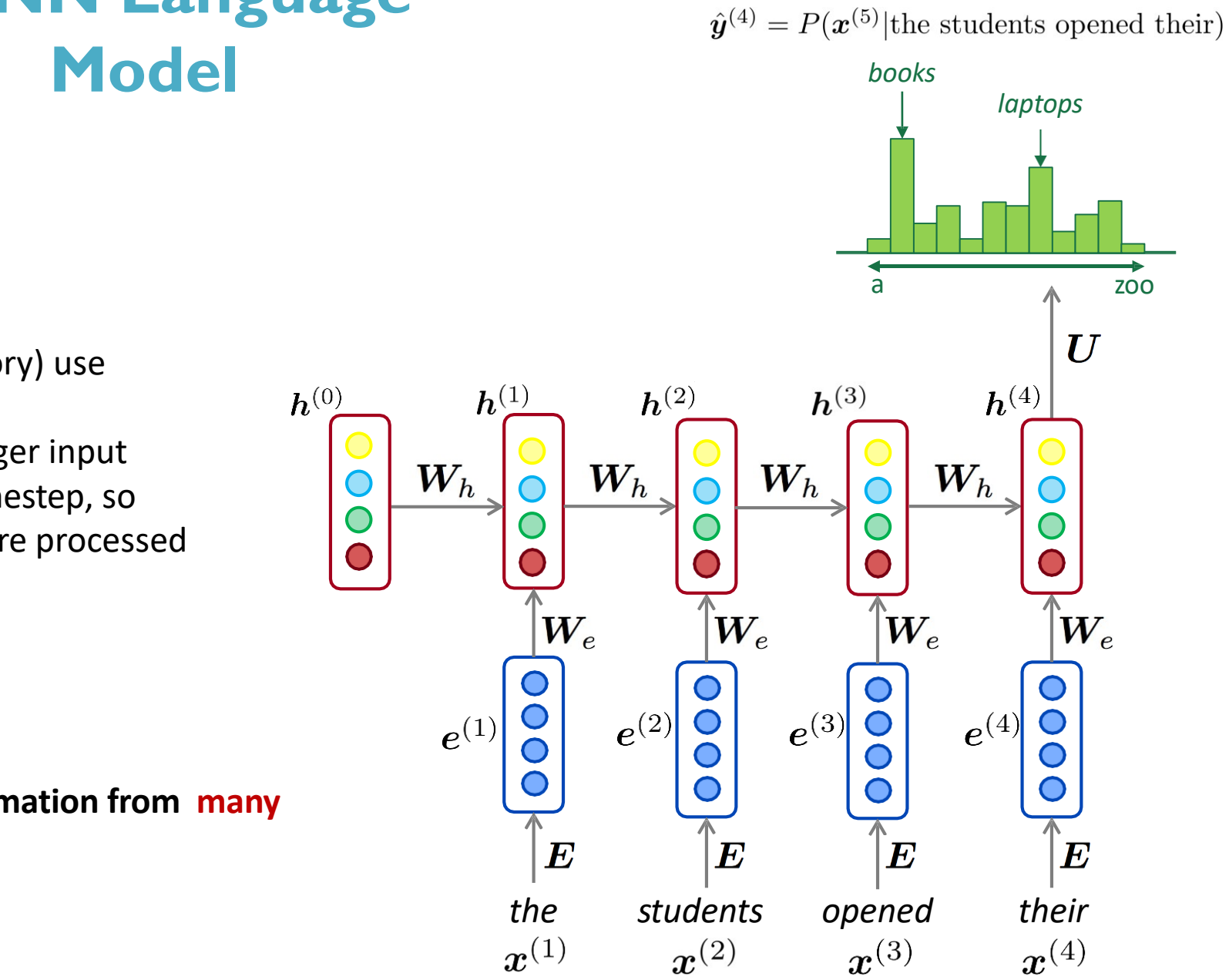
A RNN Language Model

RNN **Advantages**:

- Can process any length input
- Computation for step *t* can (in theory) use information from many steps back
- Model size doesn't increase for longer input
- Same weights applied on every timestep, so there is symmetry in how inputs are processed

RNN **Disadvantages**:

- **Recurrent computation is slow**
- **In practice, difficult to access information from many steps back**



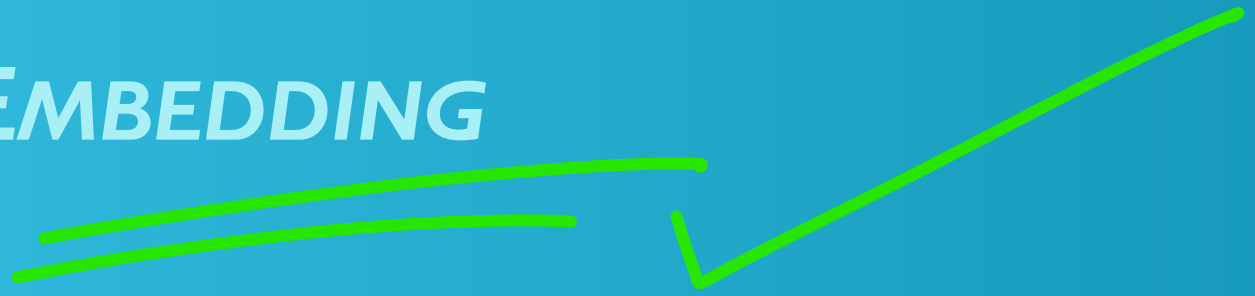
Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Local correlations

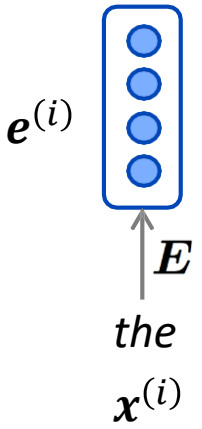
Embedding



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

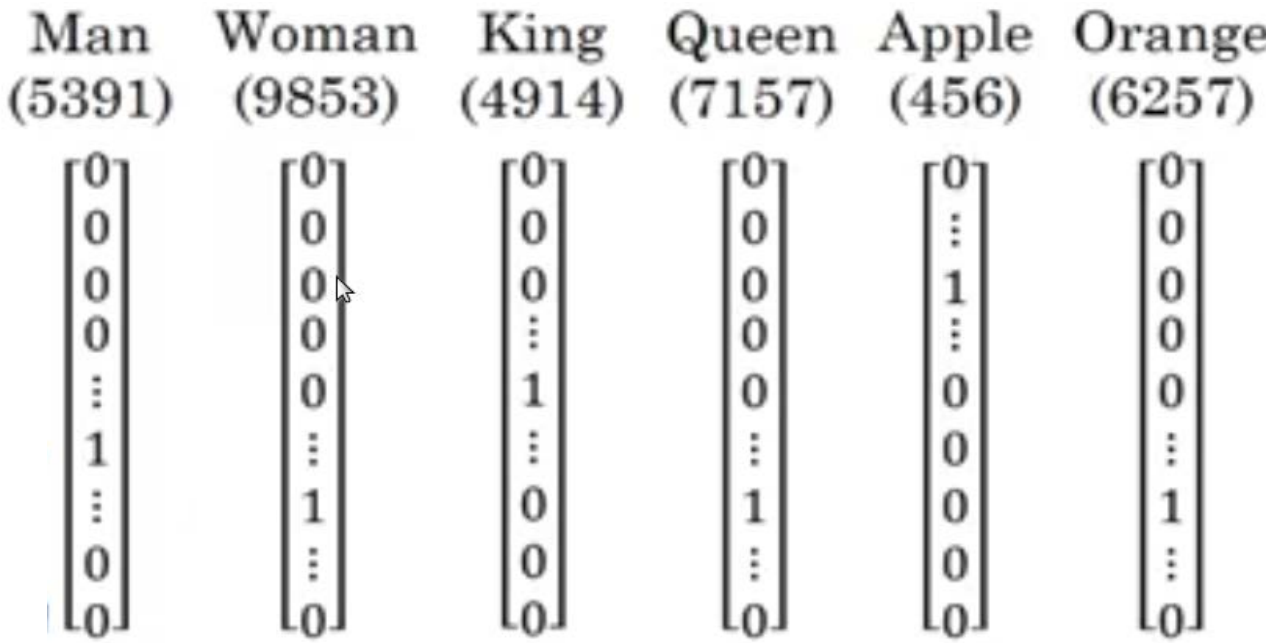
First let us now detail how to represent words as vectors: *word embedding*



UNIVERSITÉ CÔTE D'AZUR
univ-cotedazur.fr

Word representation

- Vocabulary V = [a, aaron, …, zulu, <UNK>]
- 1-hot representation of words



I want a glass of orange _____.

I want a glass of apple _____.

Problem with words as discrete symbols

Example: in web search, if user searches for "Seattle motel", we would like to match documents containing "Seattle hotel".

But:

motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]

These two vectors are orthogonal.

There is no natural notion of **similarity** for one-hot vectors!

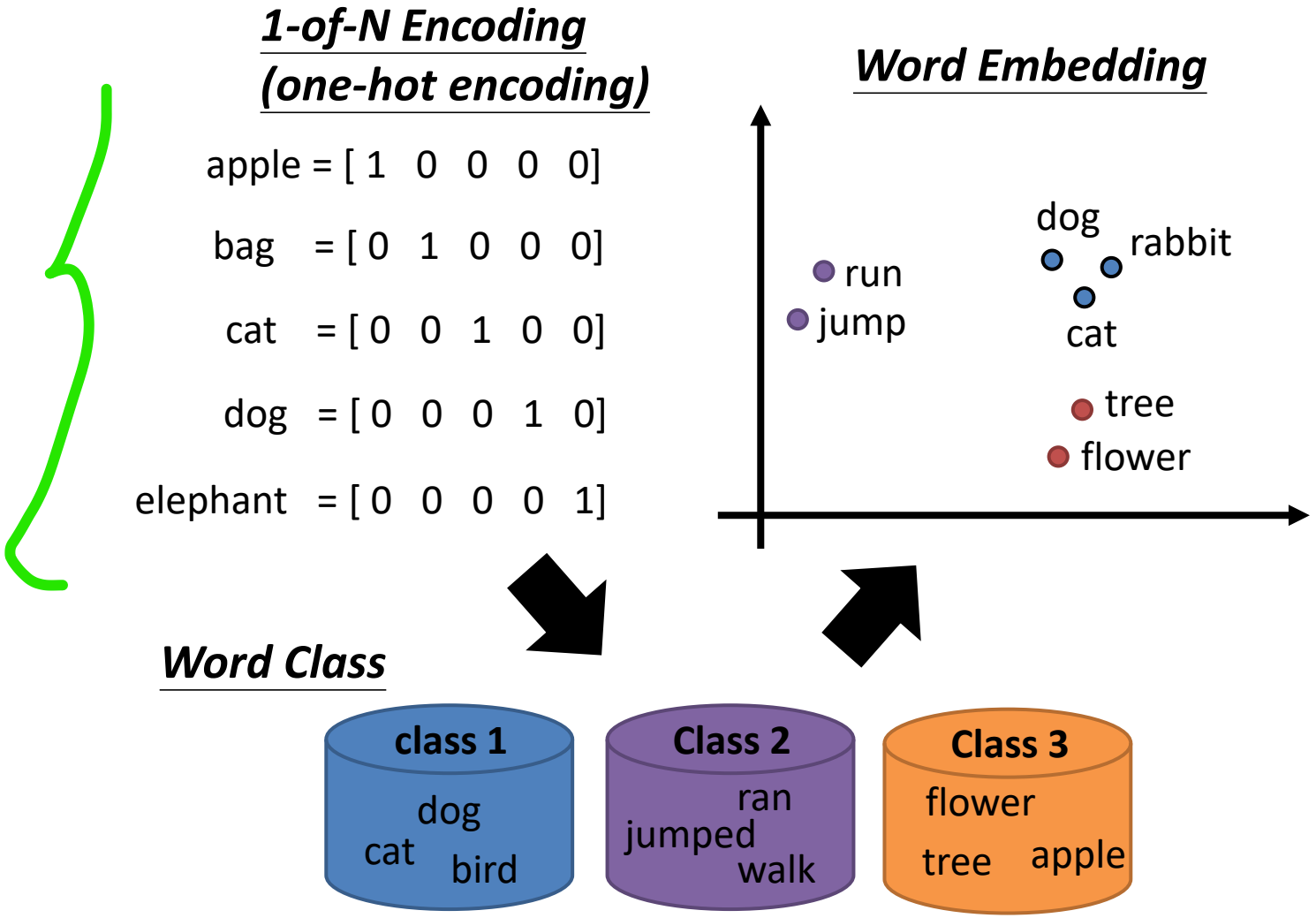
Solution:

- Could try to rely on WordNet's list of synonyms to get similarity?
  - But it is well-known to fail badly: incompleteness, subjective, missing nuance, etc.
- **Instead: learn to encode similarity in the vectors themselves**

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Learn to encode similarity in the vectors themselves



Featurized representation: word embedding

- The model is trained to complete the missing word:
  - The model gets I want a glass of orange juice . , with the associated answer "juice".
- At test time, when the model receives new sentences, it should predict the missing word:
  - I want a glass of apple _____.
  - To correctly guess, it must rely on some proximity between apple and orange.

→ We must find a way to numerically encode this proximity of words:

- Rather than representing every word with its index in the dictionary, represent it with a score on **common characteristics**:

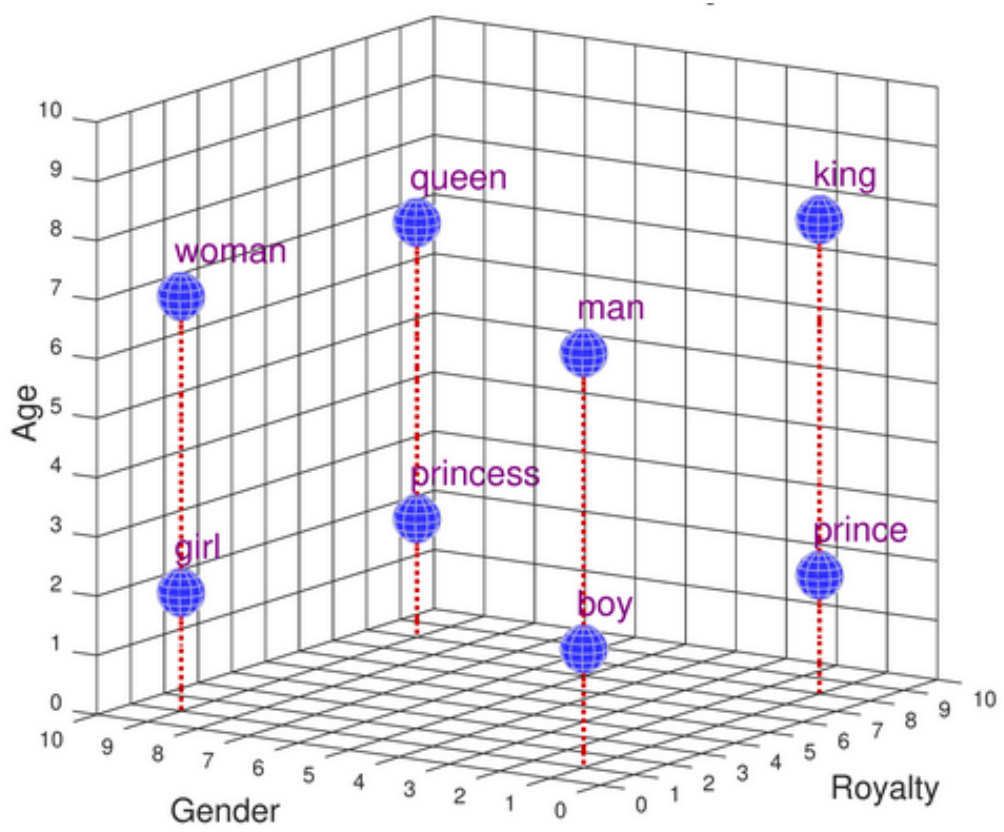
	Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
Gender	-1	1	-0.95	0.97	0.00	0.01
Royal	0.01	0.02	0.93	0.95	-0.01	0.02
Age	0.03	0.02	0.70	0.69	0.03	-0.02
Food	0.09	0.01	0.02	0.01	0.95	0.97

e_{5391}

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Visualizing word embeddings



Word Coordinates	Gender	Age	Royalty
man	[1,	7,	1]
woman	[9,	7,	1]
boy	[1,	2,	1]
girl	[9,	2,	1]
king	[1,	8,	8]
queen	[9,	7,	8]
prince	[1,	2,	8]
princess	[9,	2,	8]

© https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/tutorial.html

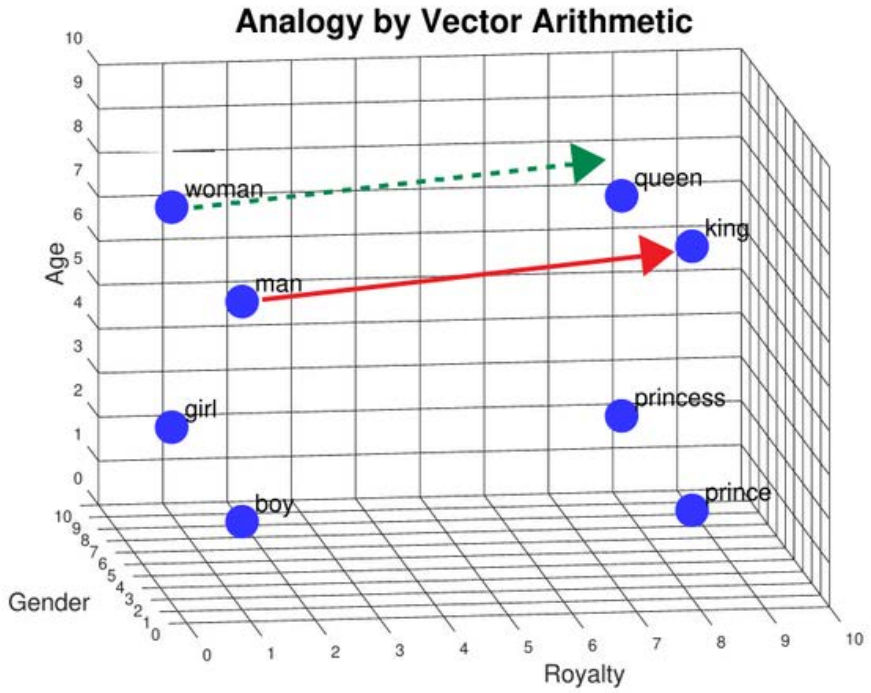
UNIVERSITÉ CÔTE D'AZUR
univ-cotedazur.fr

Analogies using word vectors

- To answer "Man is to King" what "Woman is to ?", we look for the word whose embedding is closest to:

$e_{woman} + (e_{king} - e_{man})$

So we can build a new embedding vector by adding to e_{woman} the displacement between e_{man} and e_{king}. Then we check what are the closest words in the neighborhood of the resulting vector.



$$e_{hotter} - e_{hot} \approx e_{bigger} - e_{big}$$
$$e_{Rome} - e_{Italy} \approx e_{Berlin} - e_{Germany}$$
$$e_{king} - e_{queen} \approx e_{uncle} - e_{aunt}$$

© https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/tutorial.html

UNIVERSITÉ CÔTE D'AZUR
univ-cotedazur.fr

How to learn this automatically?

- Given two words w_i and w_o, we want them to be as close as possible in the projected space if they are *similar*.
- **Deliberate choice:** *similar* in terms of the context in which they appear

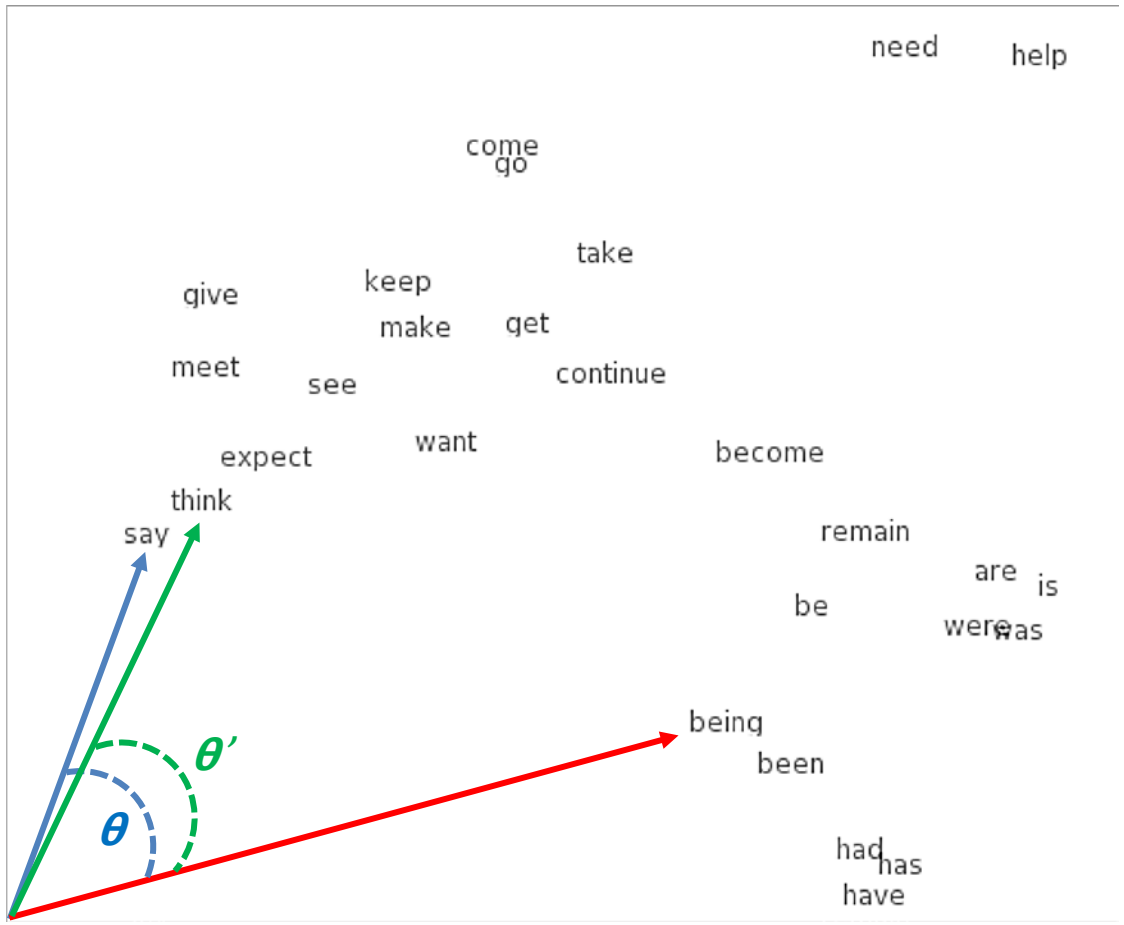
The model is given a sequence of words with the goal of predicting the next word.	The model is given a sequence of words with the goal of predicting a 'masked' word in the middle.
Example: Hannah is a ____	Example Jacob [mask] reading
Hannah is a *sister* Hannah is a *friend* Hannah is a *marketer* Hannah is a *comedian*	Jacob *fears* reading Jacob *loves* reading Jacob *enjoys* reading Jacob *hates* reading

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Word meaning as a neural word vector – visualization

$$\textit{say} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \\ 0.487 \end{pmatrix}$$



$\theta' \cong \theta \Leftrightarrow$ think $\leftrightarrow$ say

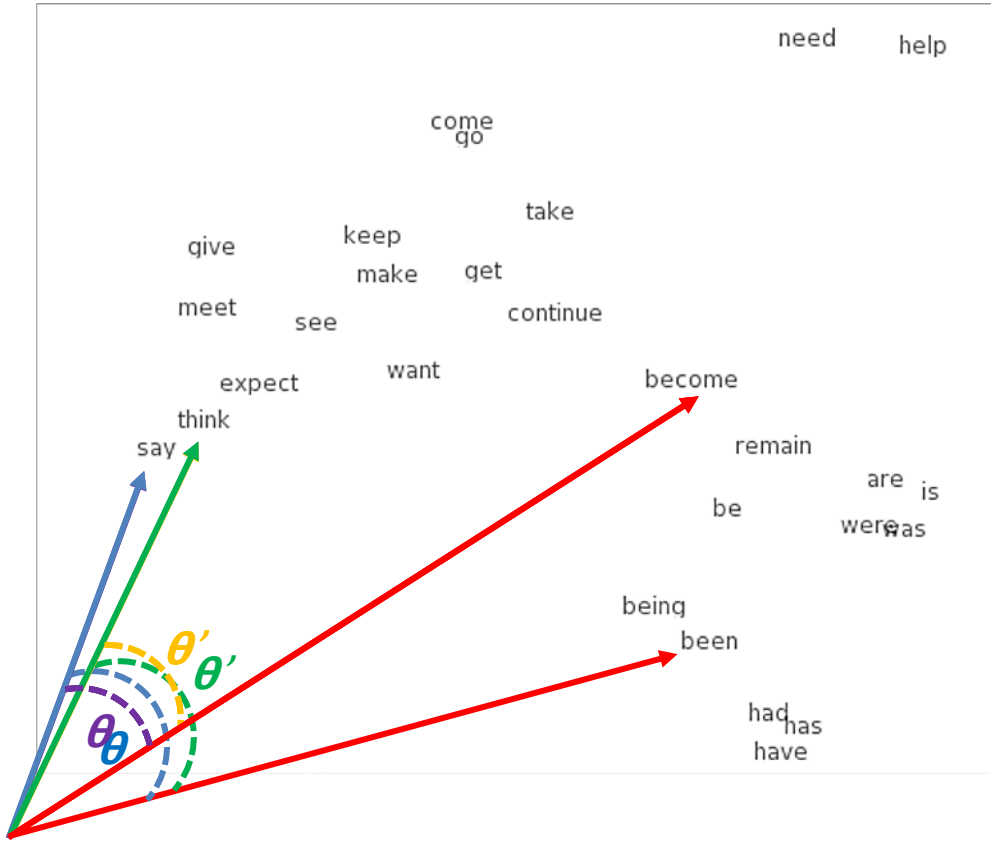
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Word meaning as a neural word vector – visualization

$\theta' \cong \theta \Leftrightarrow$ think $\cong$ say

$\theta' \cong \theta \Leftrightarrow$ think $\cong$ say



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Word2Vec Overview

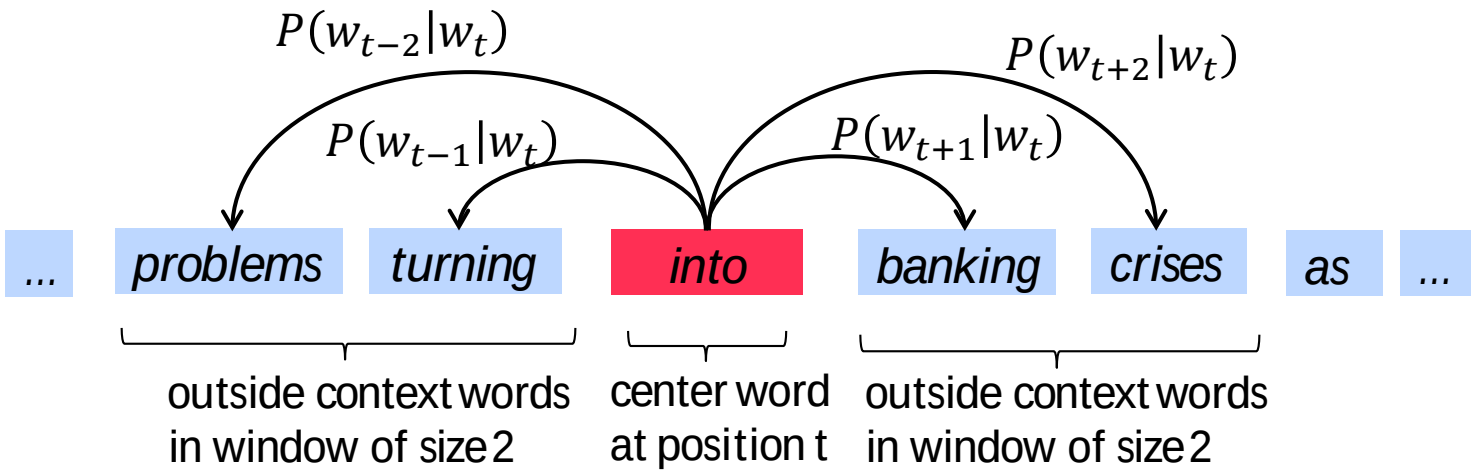
Word2vec (Mikolov et al. 2013) is a framework for learning word vectors

Idea:

- We have a large corpus of text
- Every word in a fixed vocabulary is represented by a vector
- **Go through each position** *t* in the text, **which has a center word** *c* and **context ("outside") words** *o*
- Use the (cosine) similarity of the word vectors for *c* and *o* to calculate the probability of *o* given *c* (or vice versa)
- Keep adjusting the word vectors to maximize this probability

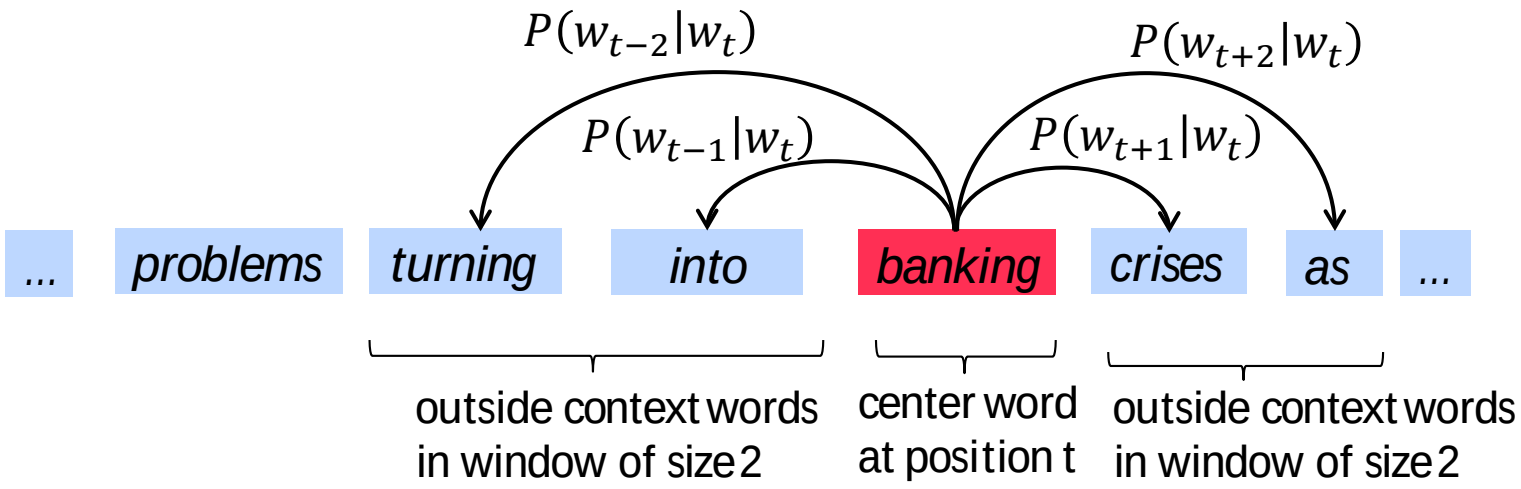
Word2Vec Overview

- Example windows and process for computing $P(w_{t+j}|w_t)$



Word2Vec Overview

- Example windows and process for computing $P(w_{t+j}|w_t)$



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Word2Vec: objective function

- We want to minimize the objective function, the (average) negative log likelihood :

$$J(\theta) = -\frac{1}{T}\log L(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m \\ j\neq 0}} \log P\left(w_{t+j} \mid w_t; \theta\right)$$

Minimizing objective function ⟺ Maximizing predictive accuracy

- Question: How to calculate $P(w_{t+j}|w_t; \theta)$?
- Answer: We will *use two* vectors per word *w*:
  - V_w when *w* is a ***center*** word
  - u_w when *w* is an ***outside*** word (i.e. ***contextual*** word)
- Then for a center word *c* and an outsite word *o*:

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w\in V}\exp(u_w^T v_c)}$$

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Word2Vec: prediction function

Exponentiation makes anything positive

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

Dot product compares similarity of *o* and *c*.

$u^T v = u.v = \sum_{i=1}^{n} u_i v_i$

Larger dot product = larger probability

Normalize over entire vocabulary to give probability distribution

- This is an example of the **softmax function** $\mathbb{R}^n \rightarrow \mathbb{R}^n$

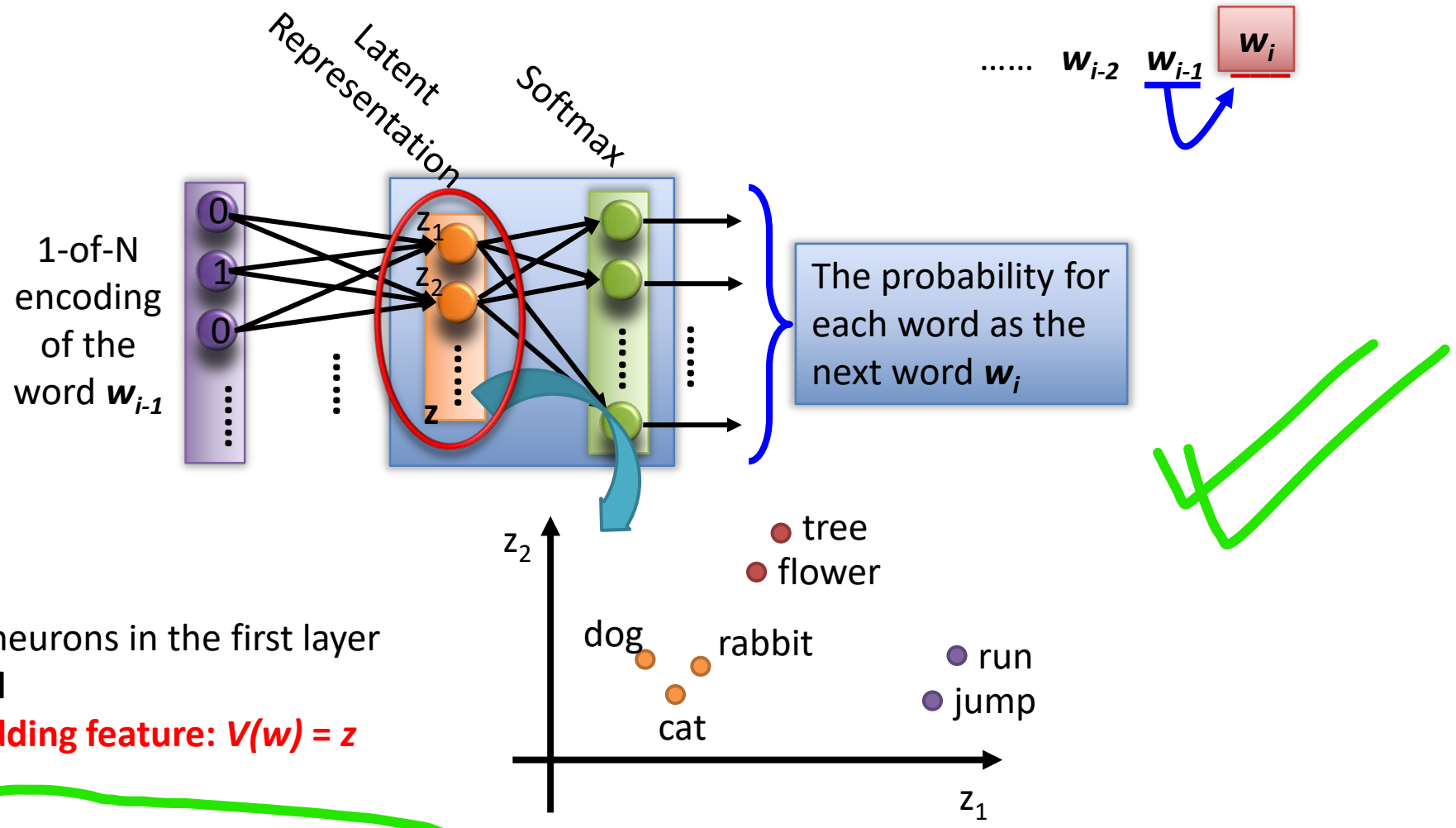
$$\text{softmax}(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^{n} \exp(x_j)} = p_i$$

- The softmax function maps arbitrary values to a probability distribution p_i
  - "max" because amplifies probability of largest x_i
  - "soft" because still assigns some probability to smaller x_i

UNIVERSITÉ CÔTE D'AZUR

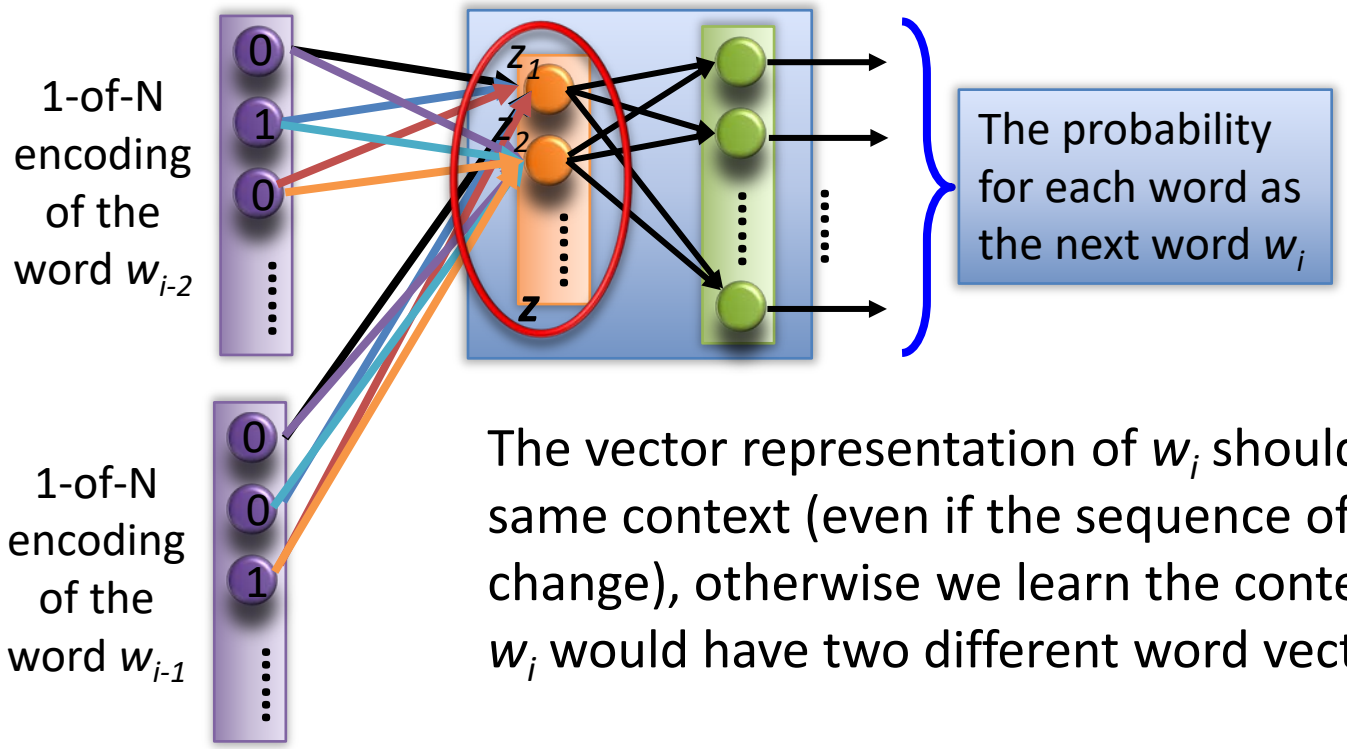
univ-cotedazur.fr

Prediction-based – Language Modeling



- Take out the **input** of the neurons in the first layer
- **Use it to represent a word**
- **Word vector, word embedding feature: *V(w) = z***

Prediction-based – Sharing Parameters



The vector representation of w_i should not differ from the same context (even if the sequence of outside words change), otherwise we learn the context, not the word w_i. w_i would have two different word vector representations.

The weights with the same color should thus be the same.

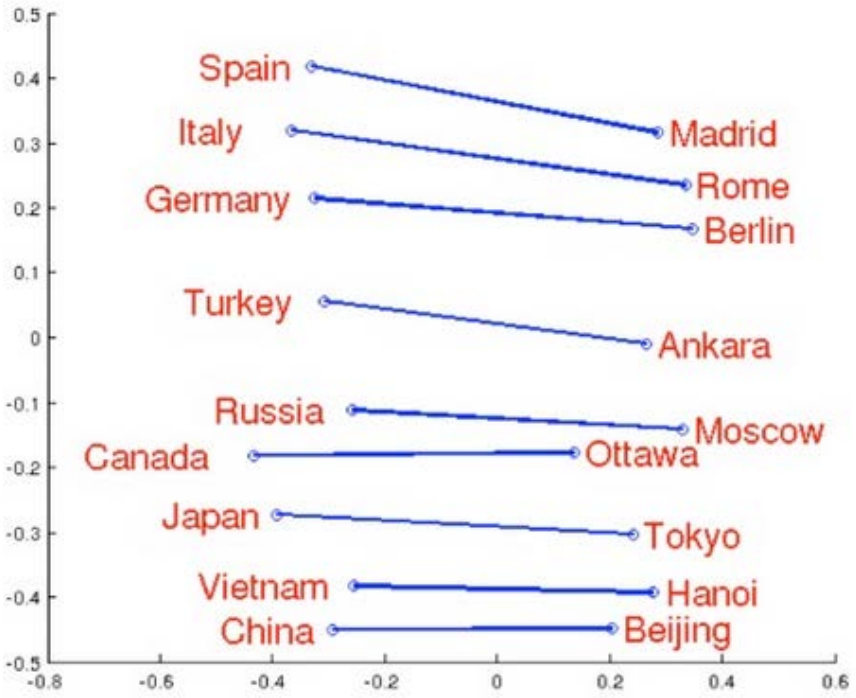
$$W_1 = W_2 = W \Rightarrow z = W(w_{i-2} + w_{i-1})$$

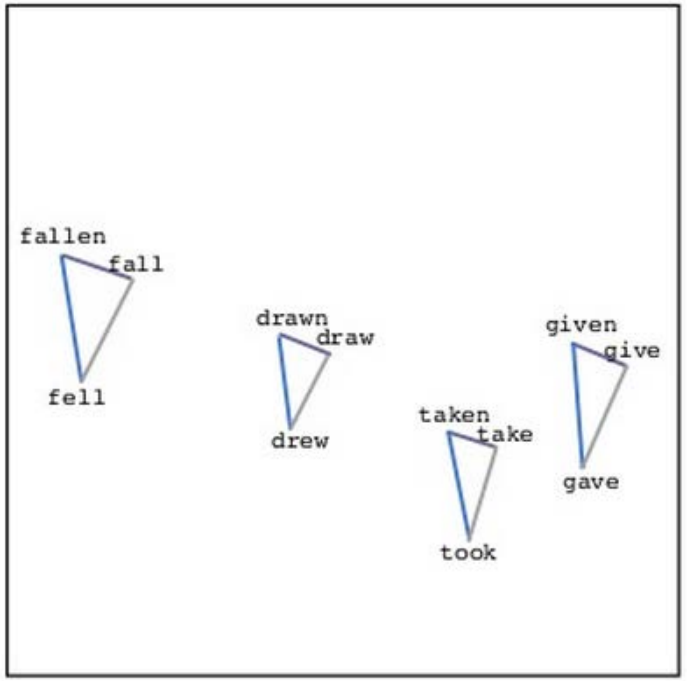
It is the same idea as the shared connections in CNN, because the edge or corner extractor is actually defined by the weights

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Word Embedding





Source: http://www.slideshare.net/hustwj/cikm-keynotenov2014

Word Embedding

- Characteristics

$$V(Germany) \approx V(Berlin) - V(Rome) + V(Italy)$$

- Solving analogies

$$V(hotter) - V(hot) \approx V(bigger) - V(big)$$
$$V(Rome) - V(Italy) \approx V(Berlin) - V(Germany)$$
$$V(king) - V(queen) \approx V(uncle) - V(aunt)$$

Rome : Italy = Berlin : ?

Compute $V(Berlin) - V(Rome) + V(Italy)$

Find the word w with the closest V(w)

Word Embedding

- Word2vec is known to be good at certain kinds of analogies:

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

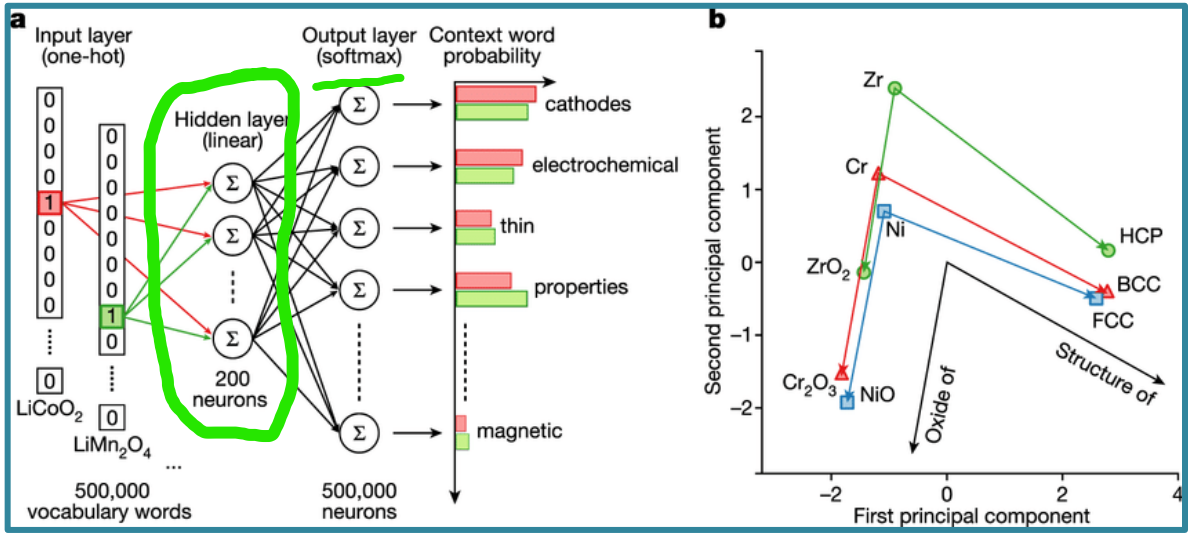
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Word2Vec not for language (Nature 2019)

a) Target words 'LiCoO2' and 'LiMn2O4' are represented as vectors with ones at their corresponding vocabulary indices (for example, 5 and 8 in the schematic) and zeros everywhere else (one-hot encoding).

These one-hot encoded vectors are used as inputs for a neural network with a single linear hidden layer (for example, 200 neurons), which is trained to predict all words mentioned within a certain distance (context words) from the given target word.



- **For similar battery cathode materials such as LiCoO2 and LiMn2O4, the context words that occur in the text are mostly the same** (for example, 'cathodes', 'electrochemical', and so on), which leads to similar hidden layer weights after the training is complete.
- These hidden layer weights are the actual word embeddings.
- The softmax function is used at the output layer to normalize the probabilities.
- **b)** Word embeddings for Zr, Cr and Ni, their principal oxides and crystal symmetries (at standard conditions) projected onto two dimensions using principal component analysis and represented as points in space. The relative positioning of the words encodes materials science relationships, such that there exist consistent vector operations between words that represent concepts such as 'oxide of' and 'structure of'.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Word2Vec not for language (Nature 2019)

Relationship	Example vector operation	Answer	Validation pairs	Accuracy (%)
Chemical element names	helium - He + Fe	= iron	8372	71.4
Crystal symmetries	cubic - GaAs + CdSe	= hexagonal	2034	35.4
Crystal structure names	zincblende - GaP + GaN	= wurtzite	556	18.7
Elemental crystal structures	dhcp - La + Cr	= bcc	1198	48.6
Principal oxides	Al_2O_3 - Al + Si	= SiO_2	650	48.8
Units	pressure - Pa + Hz	= frequency	452	35.4
Magnetic properties	ferromagnetic - NiCo + IrMn	= antiferromagnetic	622	41.0
Applications	thermoelectric - PbTe + LiFePO4	= cathode materials	-	-
Grammar	structures - structure + energy	= energies	15162	61.6
Total			29046	60.1

Materials science analogies

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

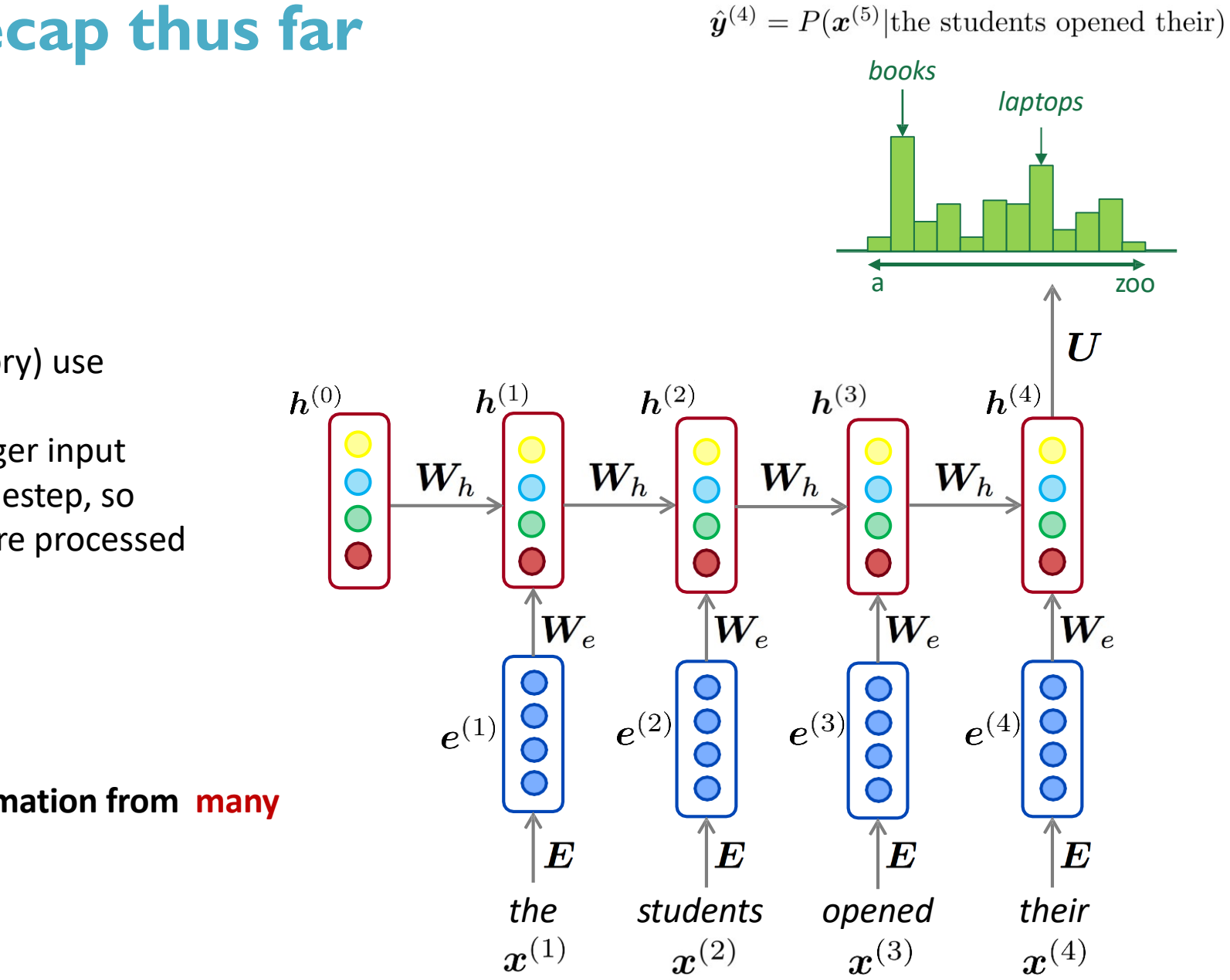
Recap thus far

RNN **Advantages**:

- Can process any length input
- Computation for step *t* can (in theory) use information from many steps back
- Model size doesn't increase for longer input
- Same weights applied on every timestep, so there is symmetry in how inputs are processed

RNN **Disadvantages**:

- **Recurrent computation is slow**
- **In practice, difficult to access information from many steps back**



Abigail See

Recap thus far

- **Language Model**: A system that predicts the next word
- **Recurrent Neural Network**: A family of neural networks that:
  - Take sequential input of any length
  - Apply the same weights on each step
  - Can optionally produce output on each step
- Vanishing gradient problem: what it is, why it happens, and why it's bad for RNNs
- LSTMs and GRUs: more complicated RNNs that use gates to control information flow; more resilient to vanishing gradients

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Why should we care about Language Modeling?

- Language Modeling is a benchmark task that helps us measure our progress on understanding language

- Language Modeling is a subcomponent of many NLP tasks, especially those involving generating text or estimating the probability of text:
  - Predictive typing
  - Speech recognition
  - Handwriting recognition
  - Spelling/grammar correction
  - Authorship identification
  - Machine translation
  - Summarization
  - Dialogue
  - etc.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Neural Machine Translation

- Neural Machine Translation (NMT) is a way to do Machine Translation with a *single neural network*

- The neural network architecture is called sequence-to-sequence (aka seq2seq) and it involves *two* RNNs.

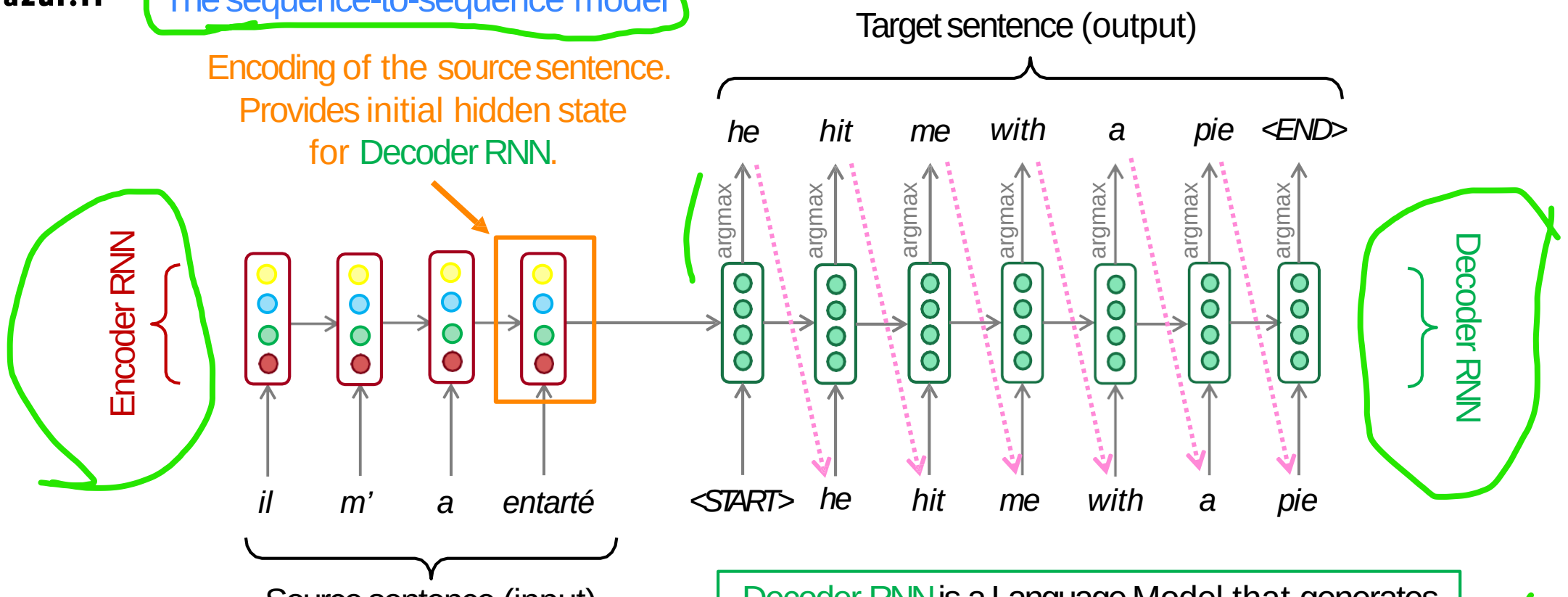
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Neural Machine Translation (NMT)

The sequence-to-sequence model

Encoding of the source sentence. Provides initial hidden state for Decoder RNN.



Encoder RNN produces an encoding of the source sentence.

Decoder RNN is a Language Model that generates target sentence, *conditioned on encoding*.

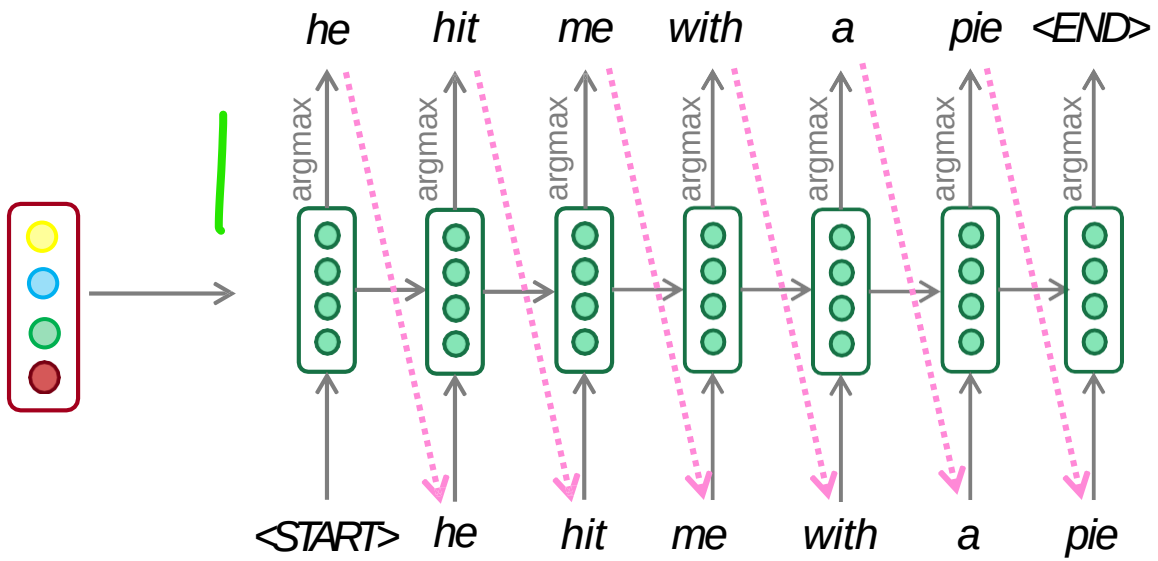
Note: This diagram shows **test time** behavior: decoder output is fed in ⇢ as next step's input

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Greedy decoding

- We saw how to generate (or "decode") the target sentence by taking argmax on each step of the decoder



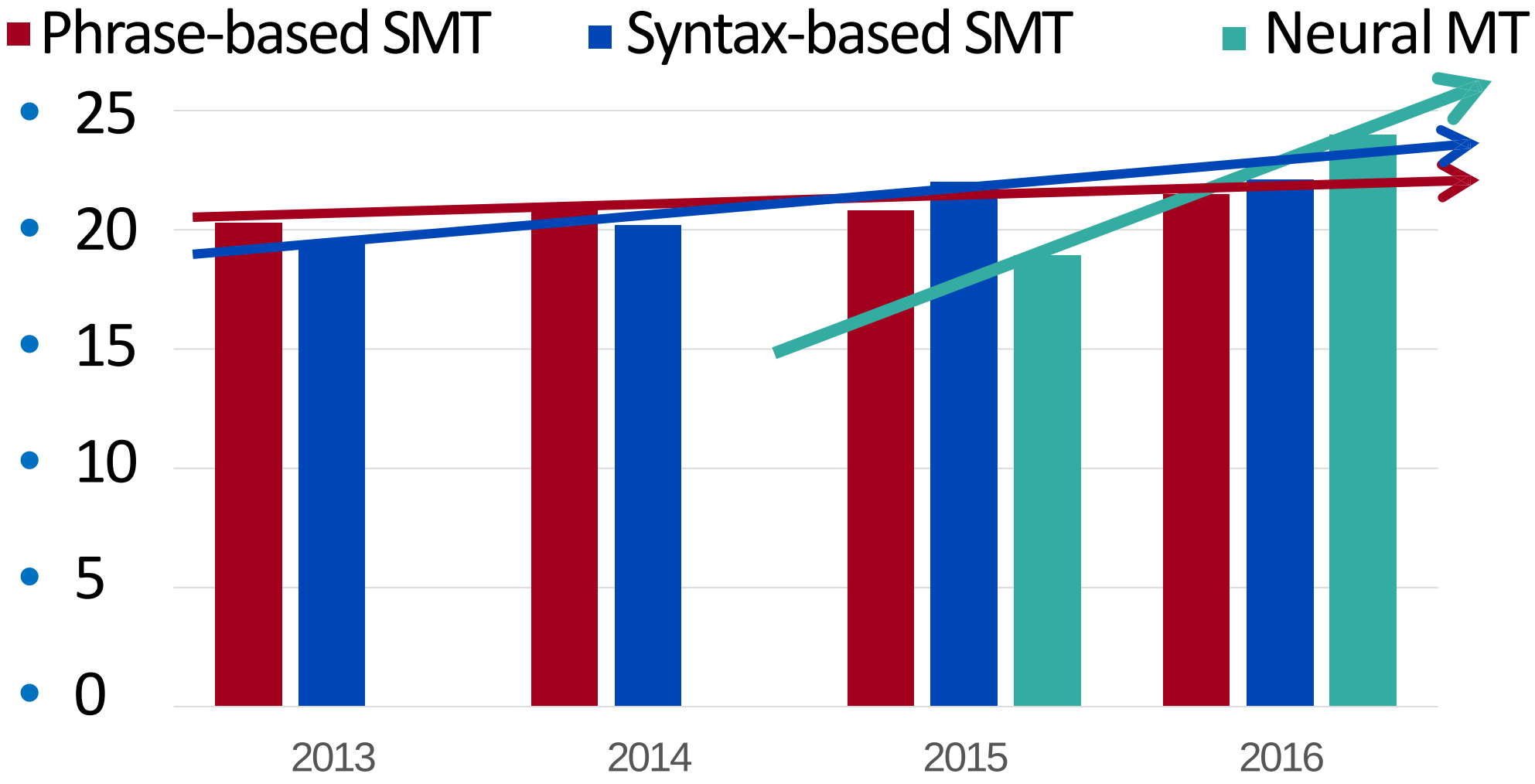
- This is greedy decoding (take most probable word on eachstep)
- **Problems with this method?**

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

MT progress over time

[Edinburgh En-De WMT newstest2013 Cased BLEU; NMT 2015 from U. Montréal]



Source: http://www.meta-net.eu/events/meta-forum-2016/slides/09_sennrich.pdf

NMT: the biggest success story of NLP Deep Learning

Neural Machine Translation went from a fringe research activity in **2014** to the leading standard method in **2016**

- **2014**: First seq2seq paper published
- **2016**: Google Translate switches from SMT to NMT
- This is amazing!
  - **SMT** systems, built by hundreds of engineers over many years, outperformed by NMT systems trained by a handful of engineers in a few months

Abigail See

NMT research continues

NMT is the **flagship task** for NLP Deep Learning

- NMT research has pioneered many of the recent innovations of NLP Deep Learning

- In **2019**: NMT research continues to thrive
  - Researchers have found *many, many* improvements to the "vanilla" seq2seq NMT system
  - But one improvement is so integral that it is the new vanilla…

ATTENTION

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Local correlations

ATTENTION

To try to solve issues with reccurent models

UNIVERSITÉ CÔTE D'AZUR

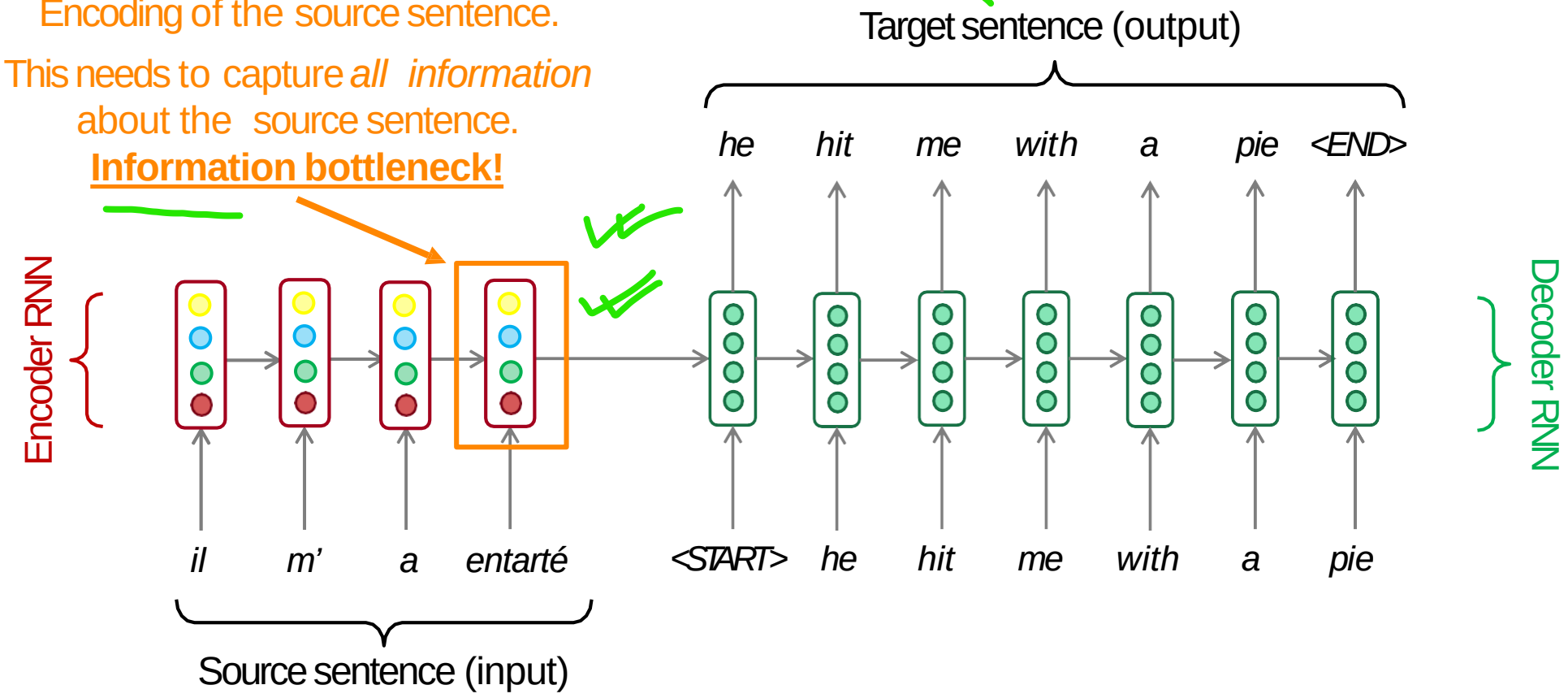
univ-cotedazur.fr

Sequence-to-sequence: the bottleneck problem

Encoding of the source sentence.

This needs to capture *all information* about the source sentence.

Information bottleneck!



Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

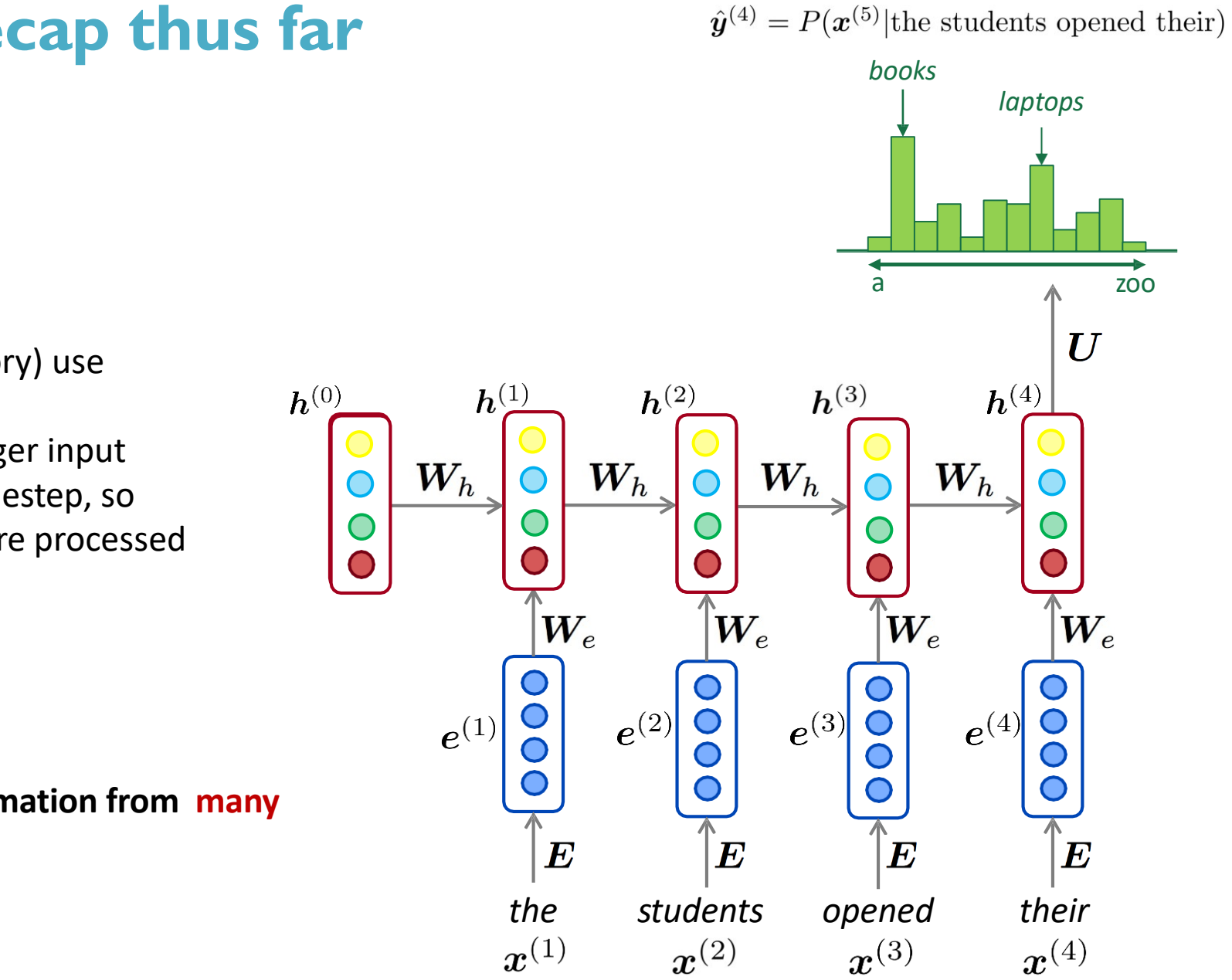
Recap thus far

RNN **Advantages**:

- Can process any length input
- Computation for step *t* can (in theory) use information from many steps back
- Model size doesn't increase for longer input
- Same weights applied on every timestep, so there is symmetry in how inputs are processed

RNN **Disadvantages**:

- **Recurrent computation is slow**
- **In practice, difficult to access information from many steps back**



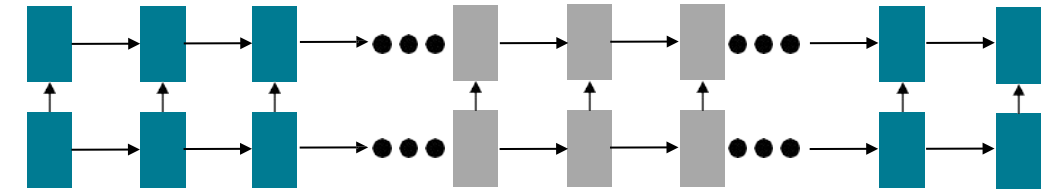
Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Issues with recurrent models: Linear interaction distance

- **O(sequence length)** steps for distant word pairs to interact means:
  - Hard to learn long-distance dependencies (because of gradient problems! Even if LSTM and GRU have mainly solved this)
  - Linear order of words is "baked in" (RNNs); not necessarily the right way to think about sentences…



Info of ***chef*** has gone through O(sequence length) many layers!

Adapted from John Hewitt

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Attention

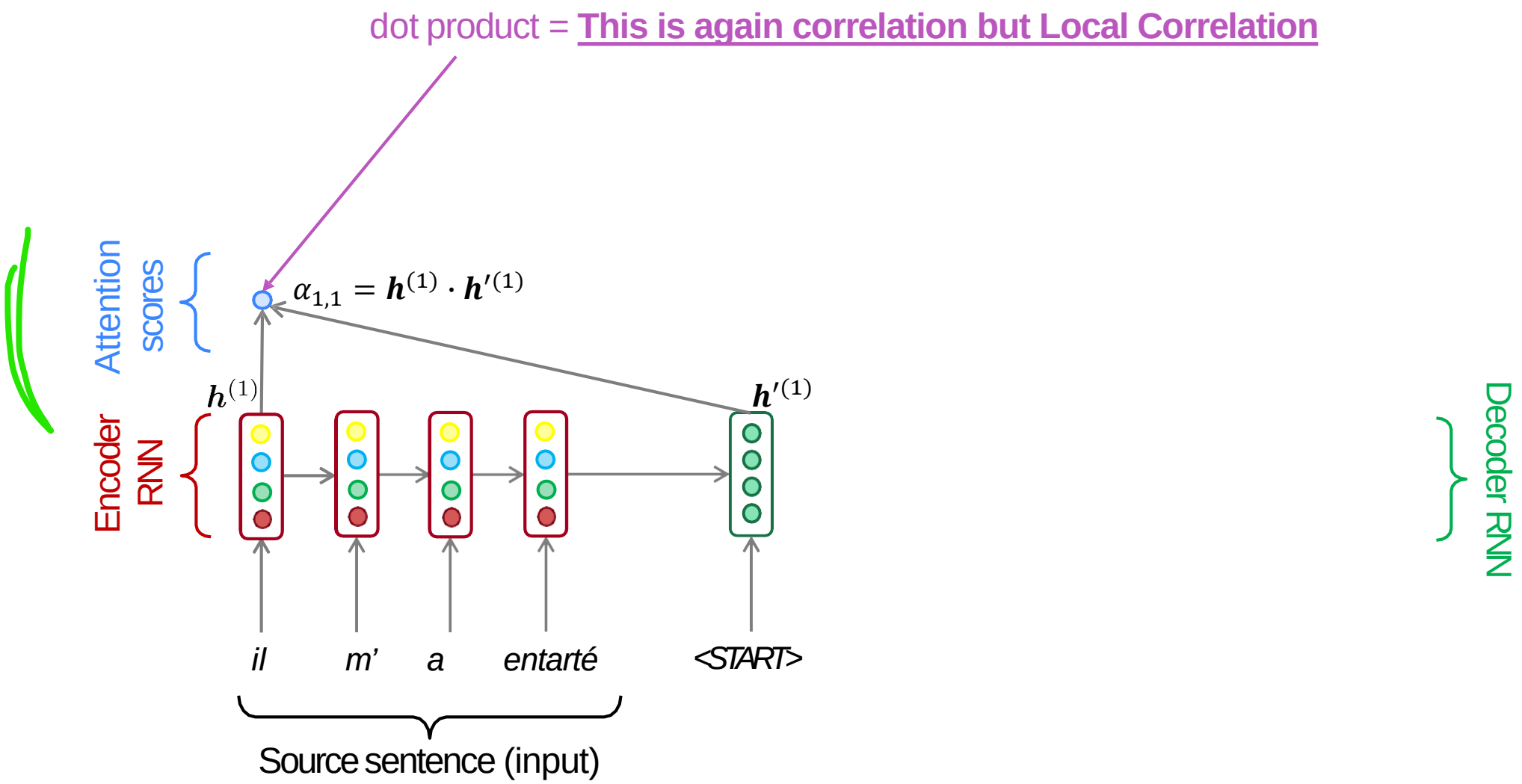
- **Attention** provides a solution to the bottleneck problem.
- Core idea: on each step of the decoder, use *direct connection to the encoder* to *focus on a particular part* of the source sequence

- First we will show via diagram (no equations), then we will show with equations

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Sequence-to-sequence with attention

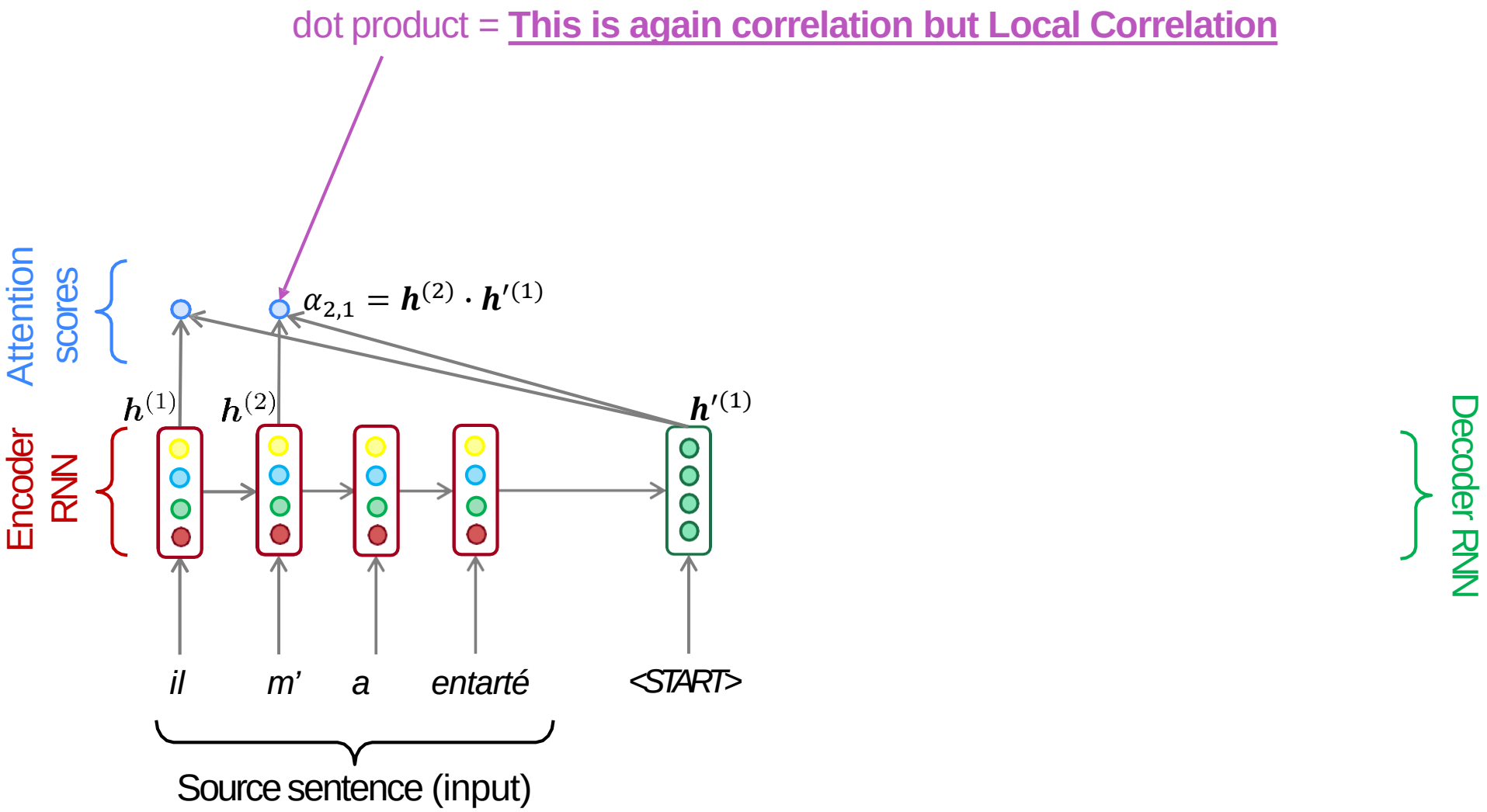


Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Sequence-to-sequence with attention



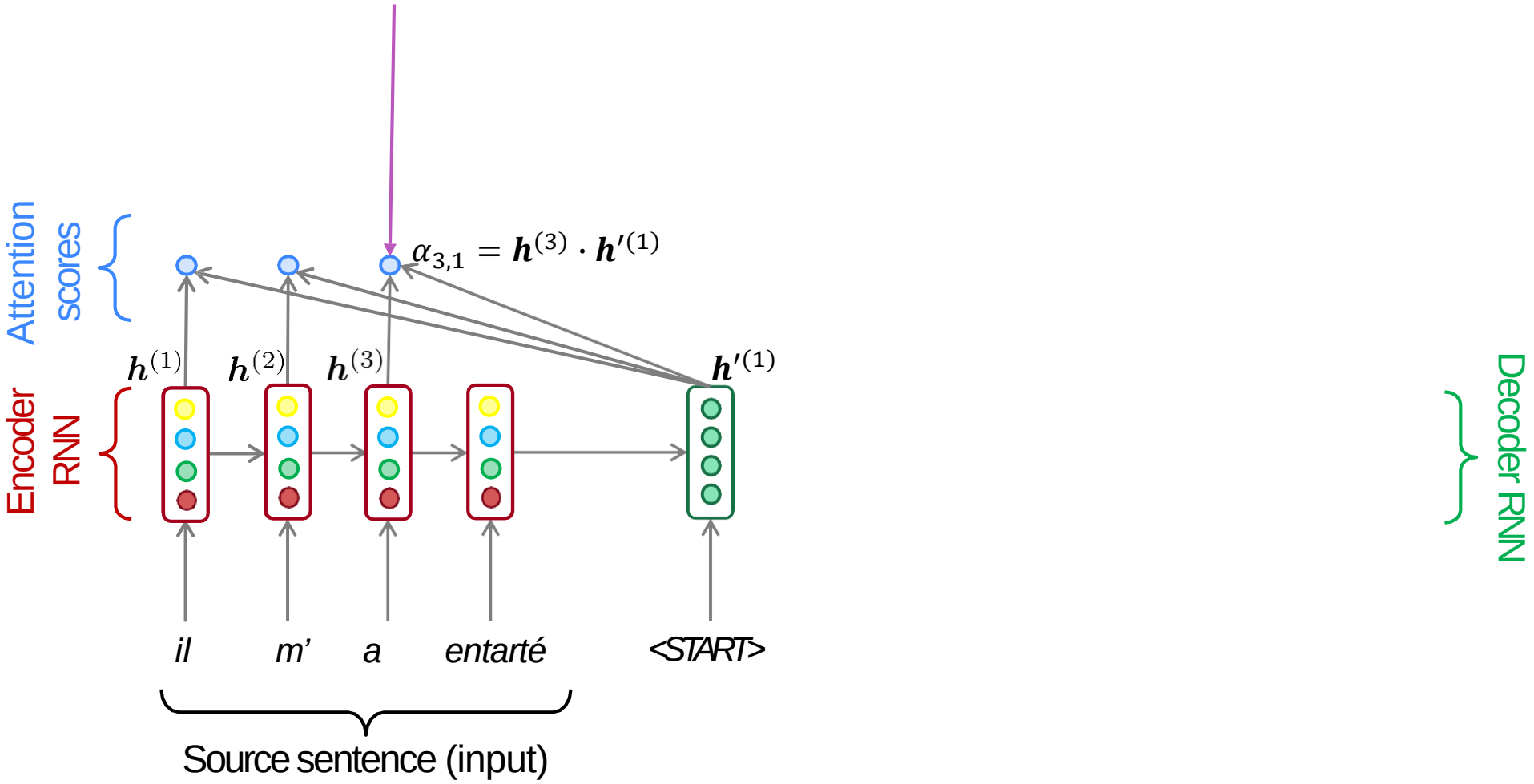
Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Sequence-to-sequence with attention

dot product = **This is again correlation but Local Correlation**



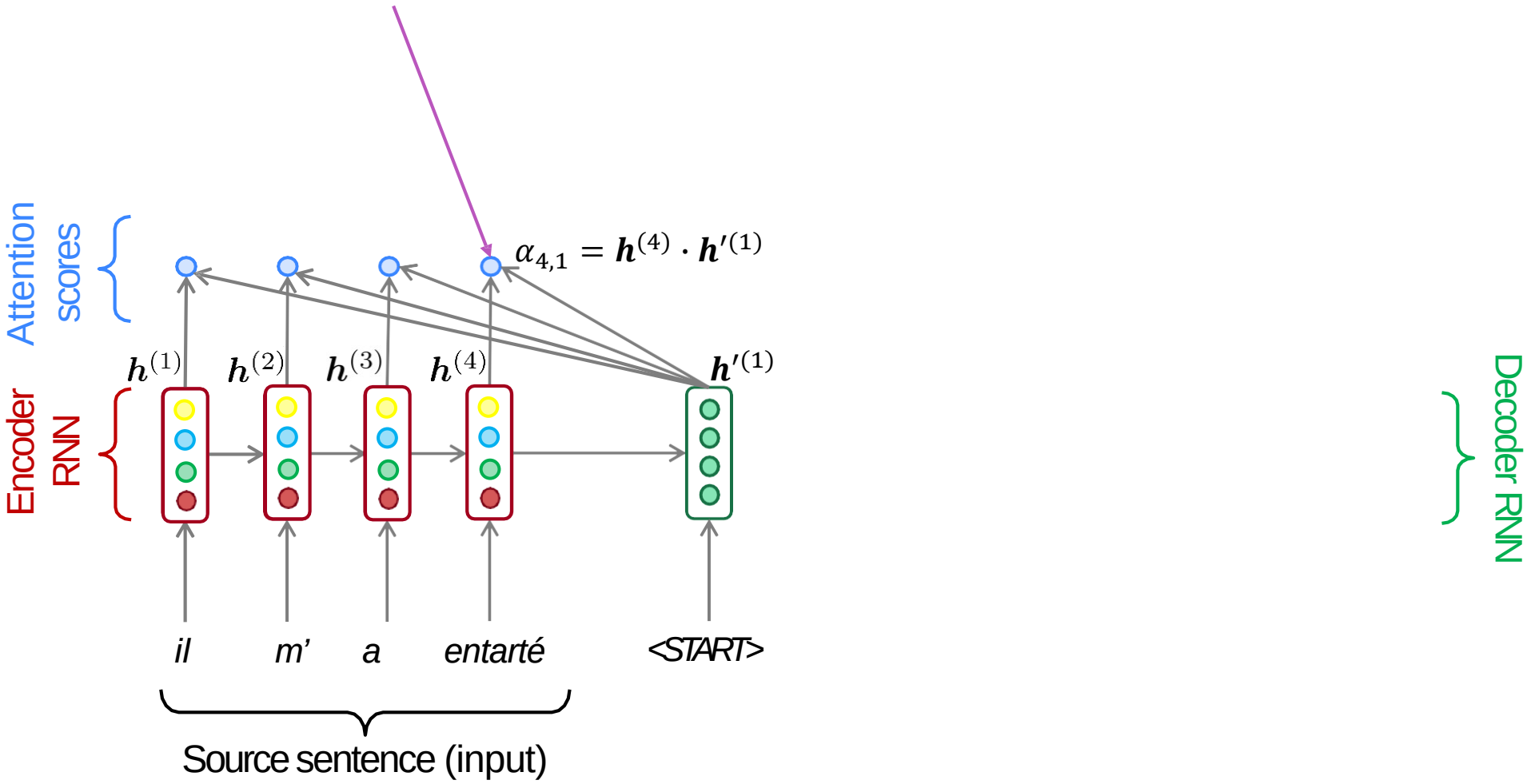
Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Sequence-to-sequence with attention

dot product = **This is again correlation but Local Correlation**



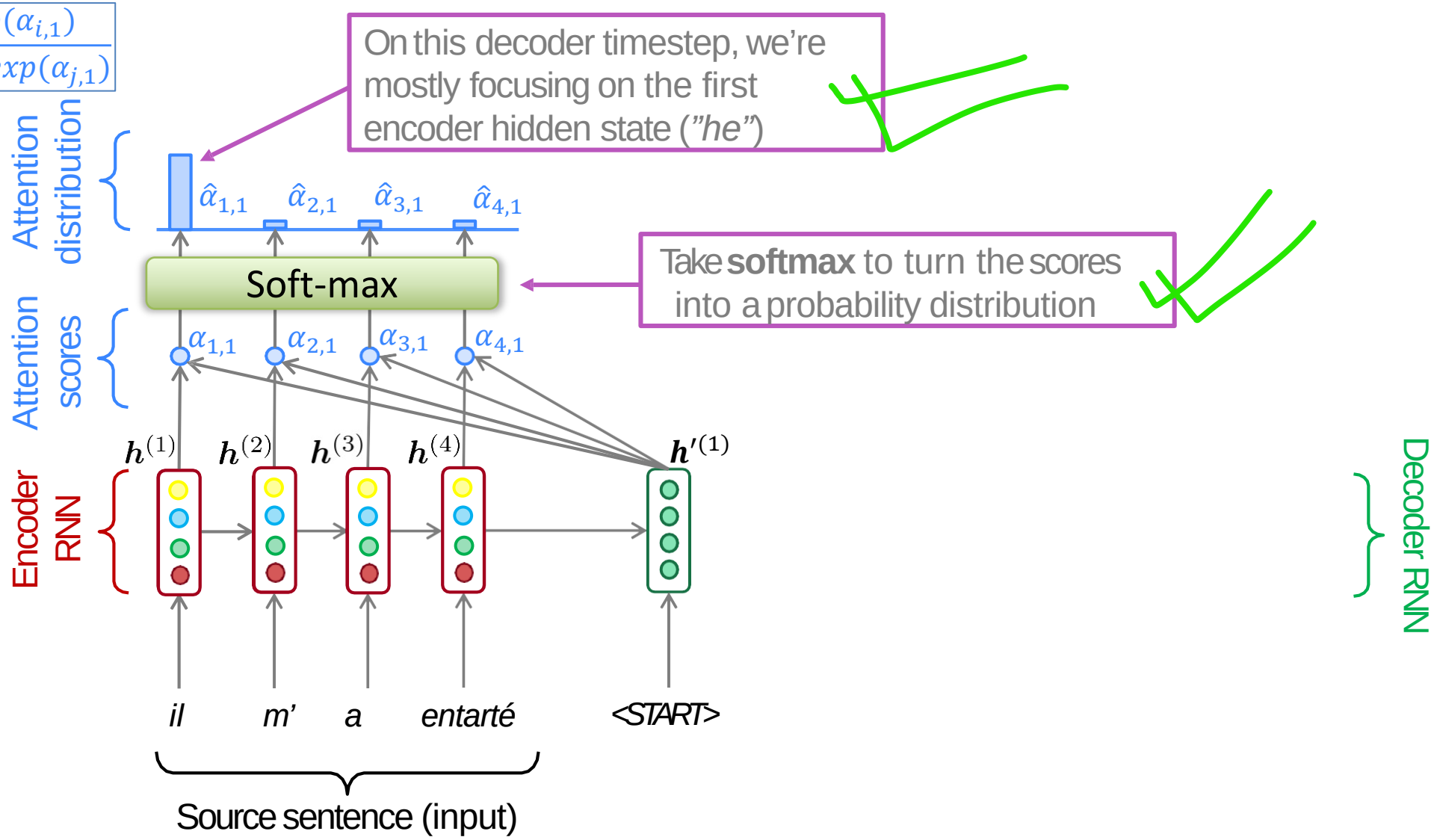
Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Sequence-to-sequence with attention

$$\hat{\alpha}_{i,1} = \frac{exp(\alpha_{i,1})}{\sum_{j=1}^{4} exp(\alpha_{j,1})}$$



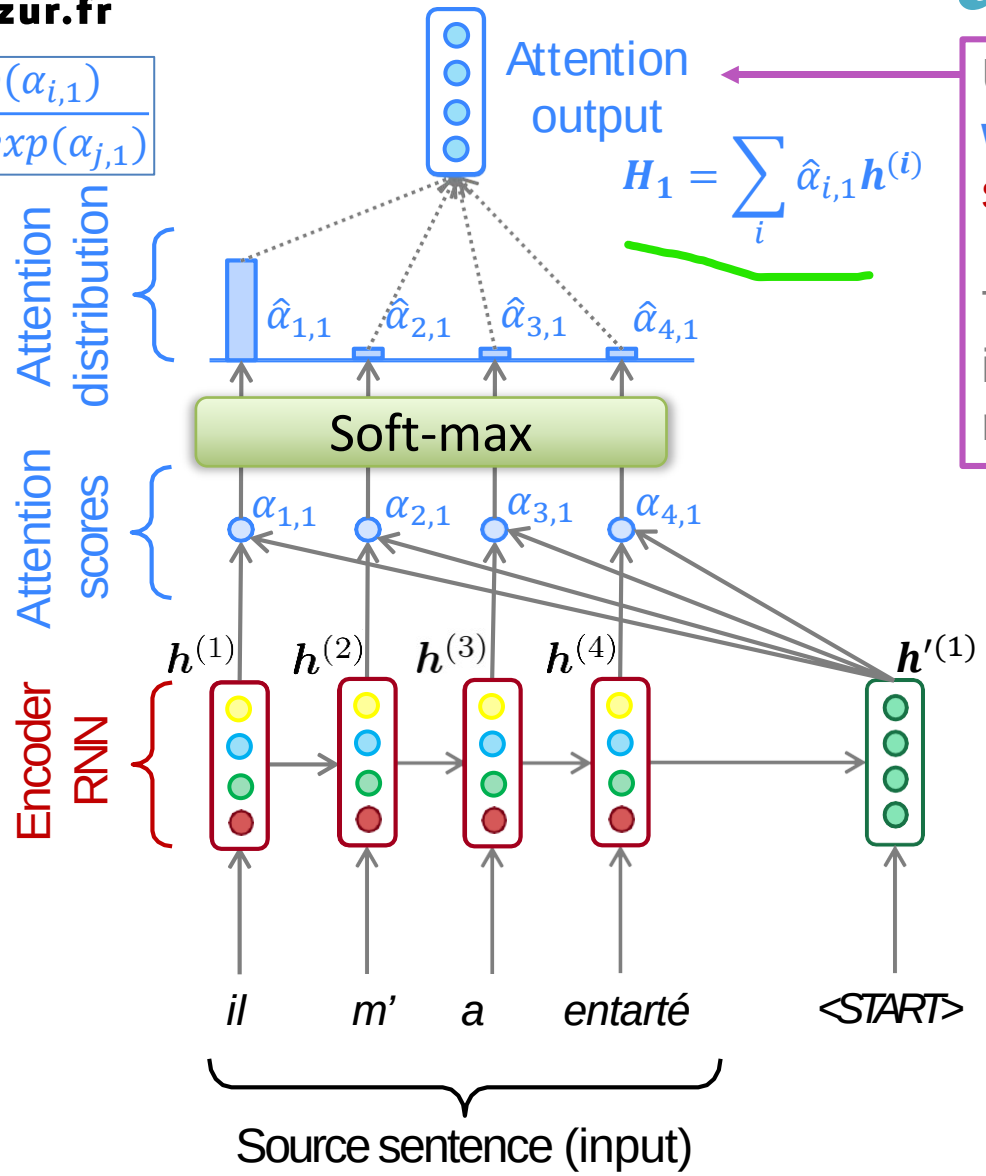
Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Sequence-to-sequence with attention

$$\hat{\alpha}_{i,1} = \frac{exp(\alpha_{i,1})}{\sum_{j=1}^{4} exp(\alpha_{j,1})}$$



Use the attention distribution to take a **weighted sum** of the encoder hidden states.

The attention output mostly contains information from the hidden states that received high attention.

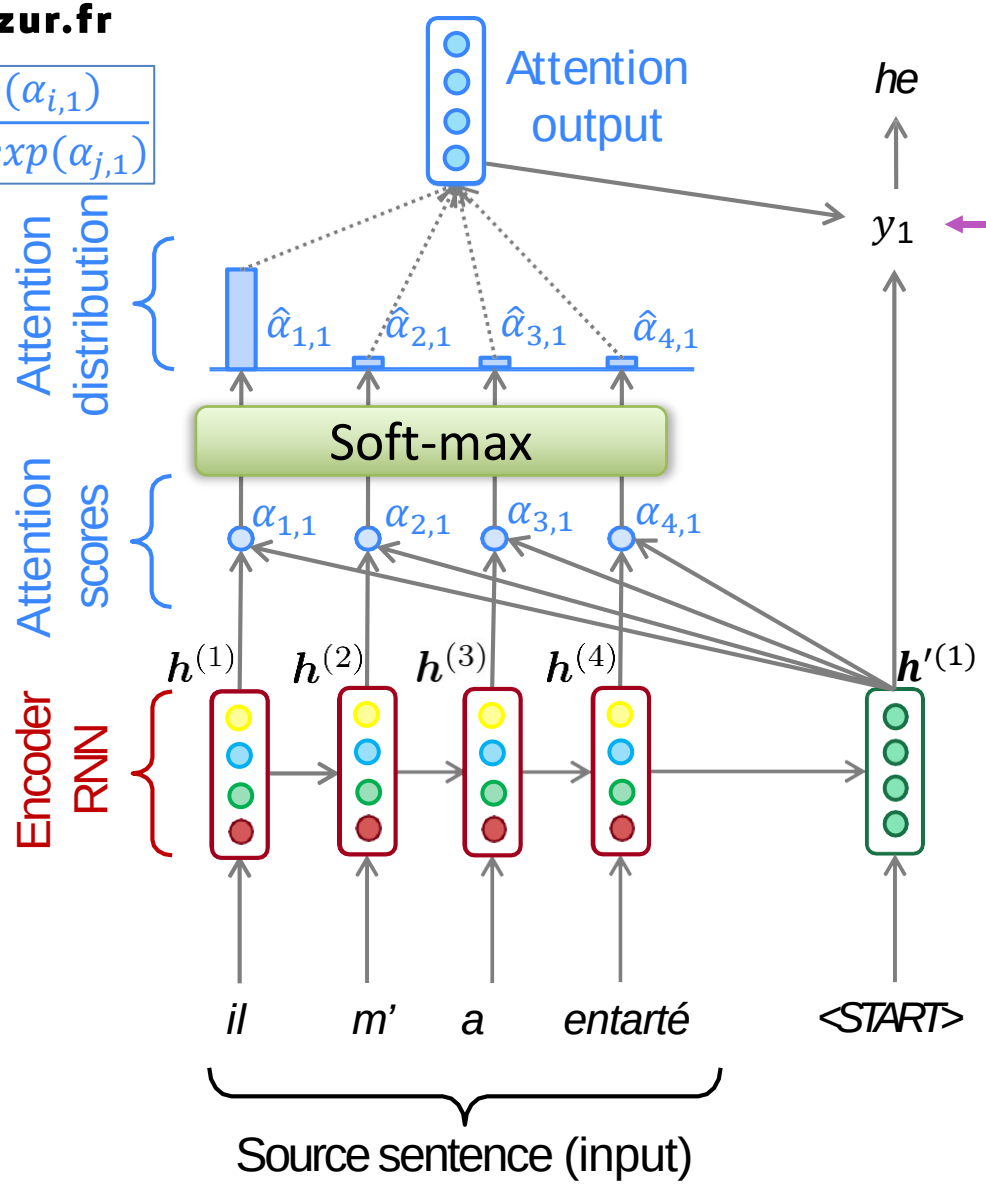
Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Sequence-to-sequence with attention

$$\hat{\alpha}_{i,1} = \frac{exp(\alpha_{i,1})}{\sum_{j=1}^{4} exp(\alpha_{j,1})}$$



Concatenate attention output with decoder hidden state, then use to compute y_1 as before

$$y_1 = U(H_1, h'^{(1)})$$

Decoder RNN

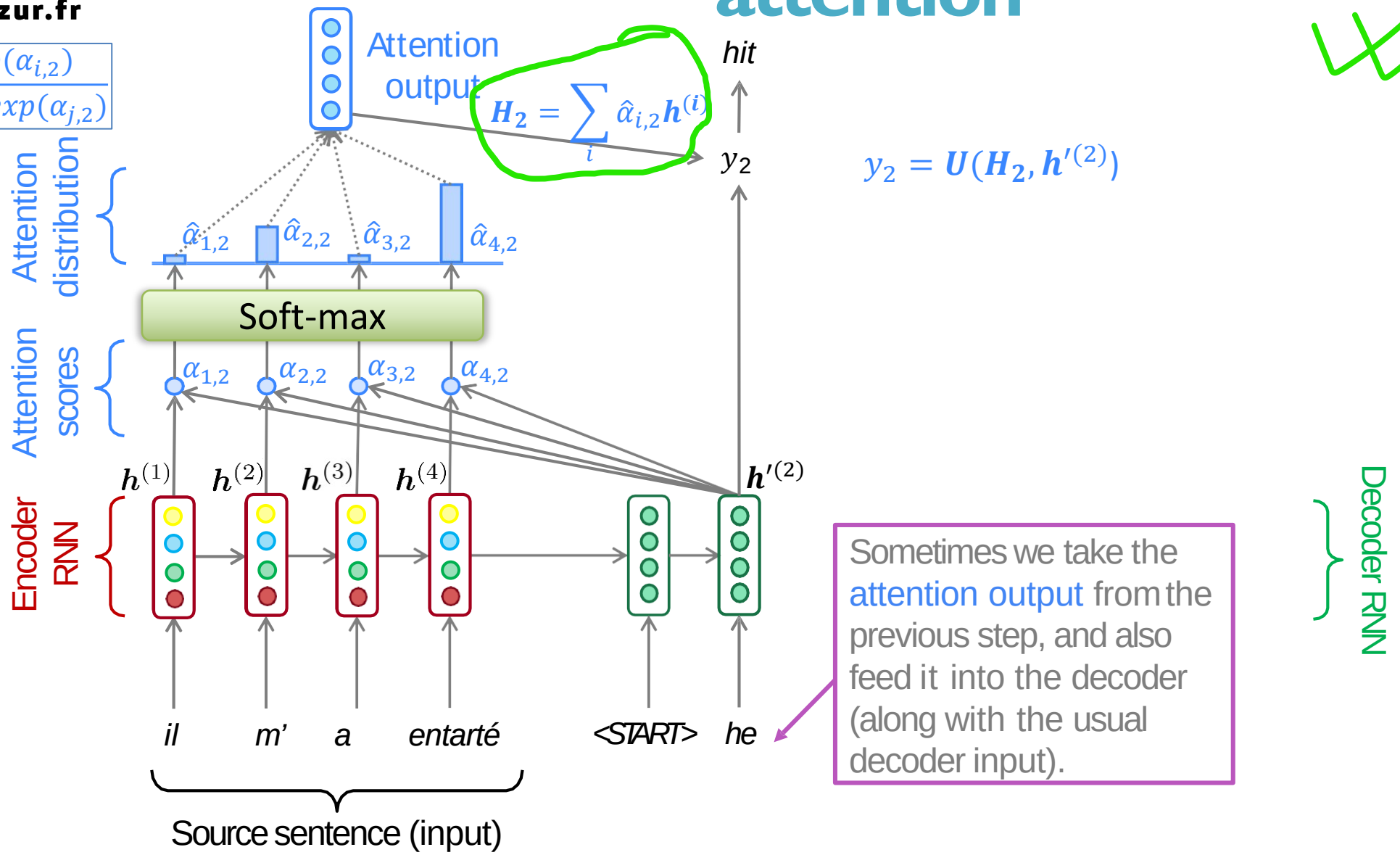
Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Sequence-to-sequence with attention

$$\hat{\alpha}_{i,2} = \frac{exp(\alpha_{i,2})}{\sum_{j=1}^{4} exp(\alpha_{j,2})}$$



Sometimes we take the attention output from the previous step, and also feed it into the decoder (along with the usual decoder input).

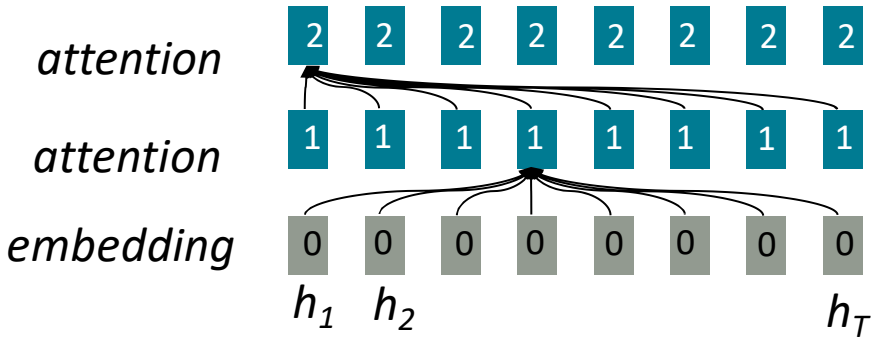
Abigail See

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

If not recurrence, then what?
How about attention?

- **Attention** treats each word's representation as a **query** to access and incorporate information from **a set of values** *(in the previous example of translation, the values are the words from the initial sentence, the encoded one).*
  - We saw attention from the **decoder** to the **encoder**; here let's think about attention **within a single sentence**.
  - If **attention** gives us access to any state… maybe we can just use attention and don't need the RNN?
- Number of unparallelizable operations not linked to sequence length.
- **All words interact at every layer!**



All words attend to all words in previous layer.

Most arrows here are omitted in the figure.

Adapted from John Hewitt

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Multimodal correlations

Self-Attention, Transformers

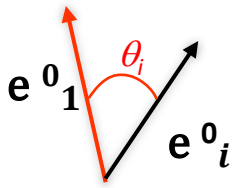
Principles of Transformers
Heart of LLMs: f(.)

- Generate a successive representation of the words in the form of recombination of the representations of the other words:

Attention scores = Correlation scores

Calculation of the similarity/correlation of word 1 with the i-th word in the analysis window:

$$\alpha_{1,i} = \boldsymbol{e^0}_1 \cdot \boldsymbol{e^0}_i / \sqrt{d} \approx \cos(\boldsymbol{e^0}_1, \boldsymbol{e^0}_i) = \cos(\theta_i)$$

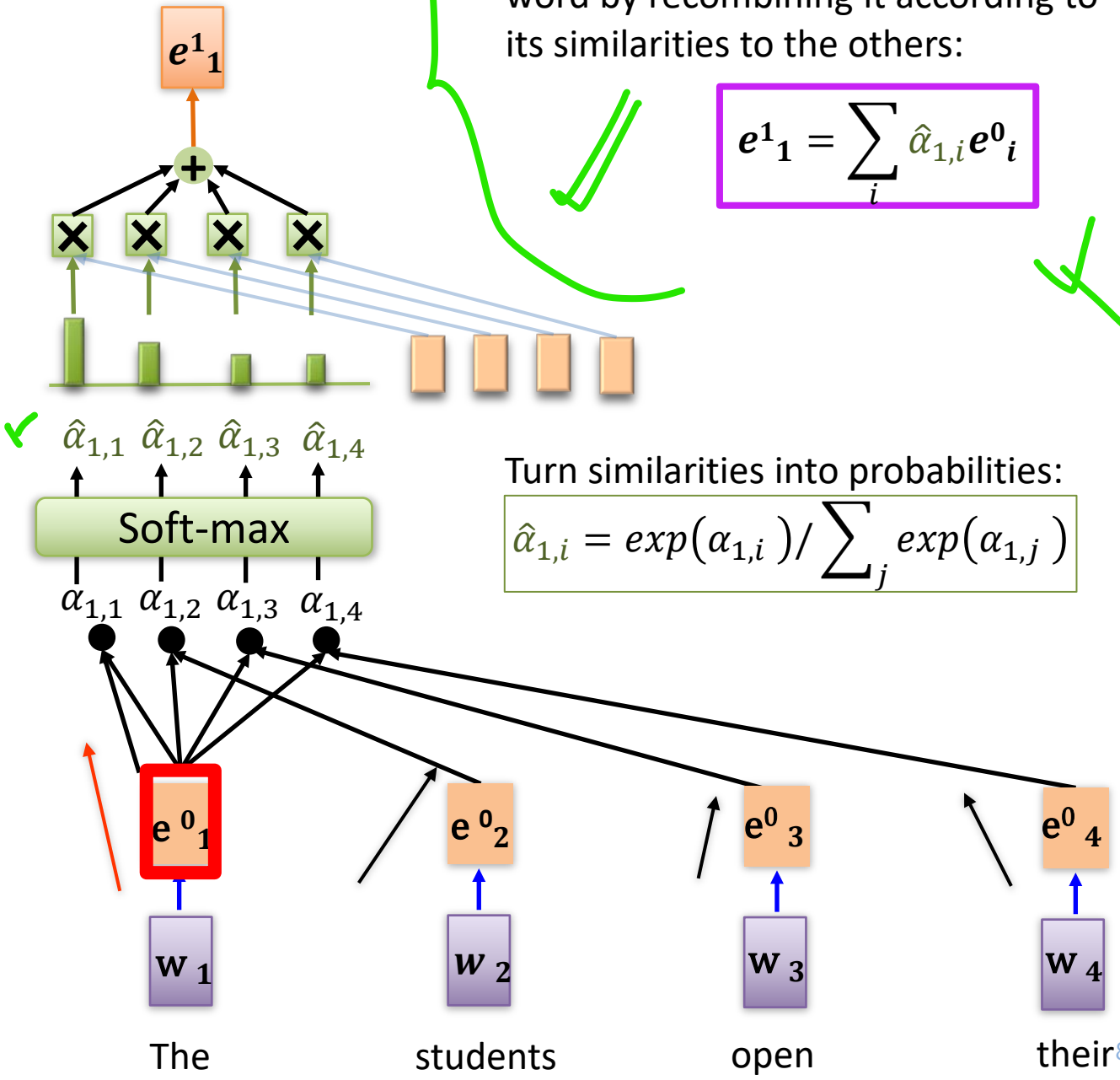


Generate a new representation of this word by recombining it according to its similarities to the others:

$$\boldsymbol{e^1}_1 = \sum_i \hat{\alpha}_{1,i} \boldsymbol{e^0}_i$$

Turn similarities into probabilities:

$$\hat{\alpha}_{1,i} = exp(\alpha_{1,i}) / \sum_j exp(\alpha_{1,j})$$



Credits: L. Sassatelli

Principles of Transformers
Heart of LLMs: f(.)

- Generate a successive representation of the words in the form of recombination of the representations of the other words:

Turn similarities into probabilities:

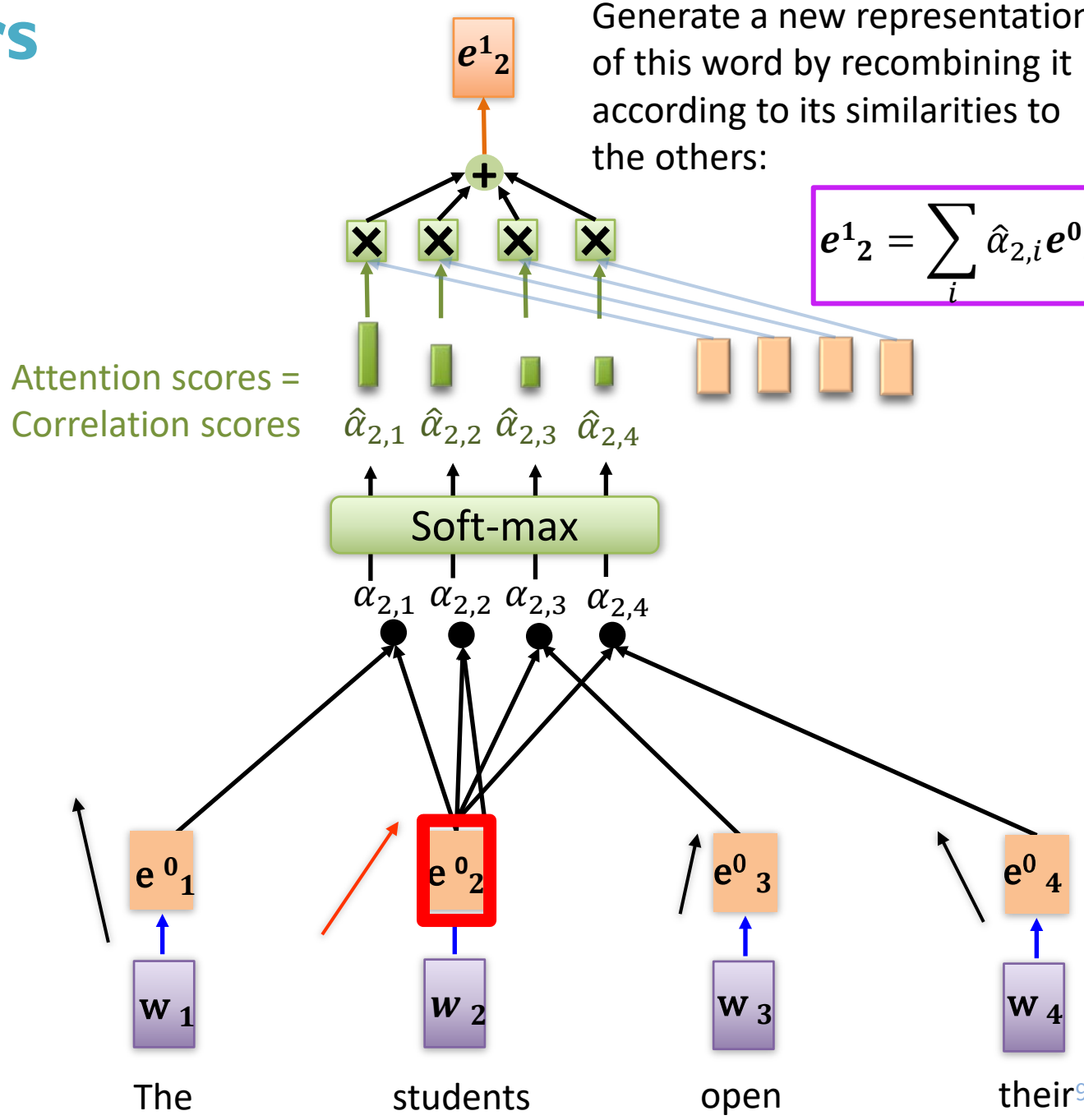
$$\hat{\alpha}_{2,i} = exp(\alpha_{2,i}) / \sum_j exp(\alpha_{2,j})$$

Calculation of the similarity/correlation of word 2 with the i-th word in the analysis window:

$$\alpha_{2,i} = \boldsymbol{e^0}_2 \cdot \boldsymbol{e^0}_i / \sqrt{d}$$

Generate a new representation of this word by recombining it according to its similarities to the others:

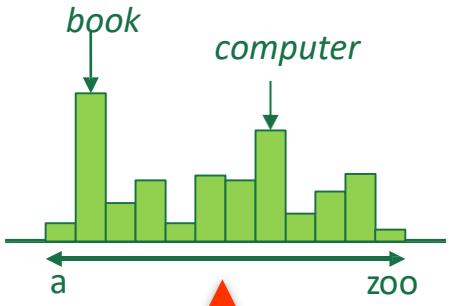
$$\boldsymbol{e^1}_2 = \sum_i \hat{\alpha}_{2,i} \boldsymbol{e^0}_i$$



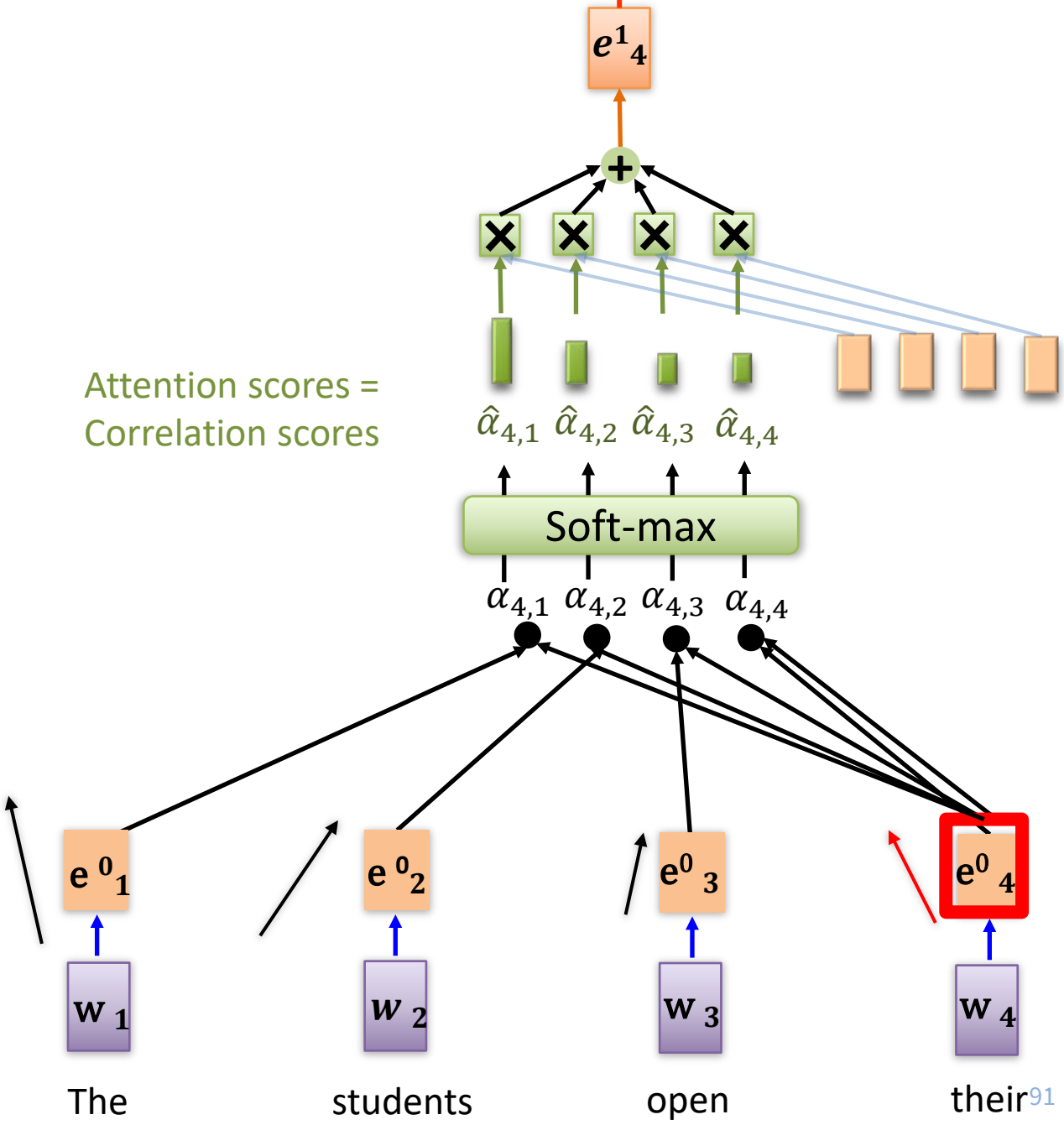
Credits: L. Sassatelli

Principles of Transformers
Heart of LLMs: g(.)

- Generate a successive representation of the words in the form of recombination of the representations of the other words:



Probability prediction g(.)



Credits: L. Sassatelli

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

https://arxiv.org/abs/1706.03762

Attention is all you need.

Make Self-Attention More Flexible: Different representations for the roles of query, key, value

$\boldsymbol{q}$: query (to match others)

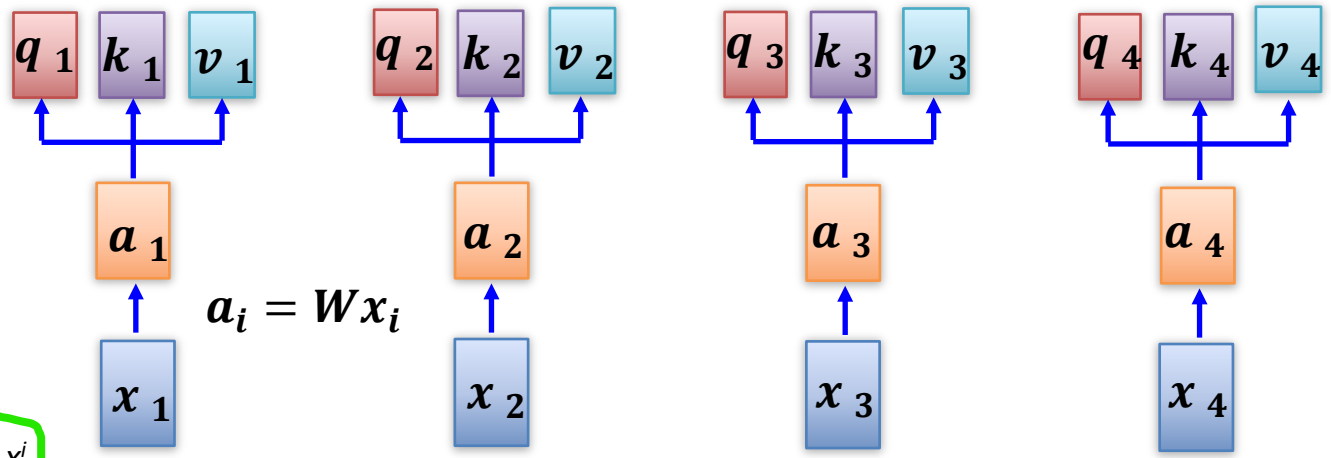
$$\boldsymbol{q_i} = \boldsymbol{W^q a_i}$$

$\boldsymbol{k}$: key (to be matched)

$$\boldsymbol{k_i} = \boldsymbol{W^k a_i}$$

$\boldsymbol{v}$: information to be extracted

$$\boldsymbol{v_i} = \boldsymbol{W^v a_i}$$



$$\boldsymbol{a_i} = \boldsymbol{W x_i}$$

a^i is the embedding of word x^i

UNIVERSITÉ CÔTE D'AZUR

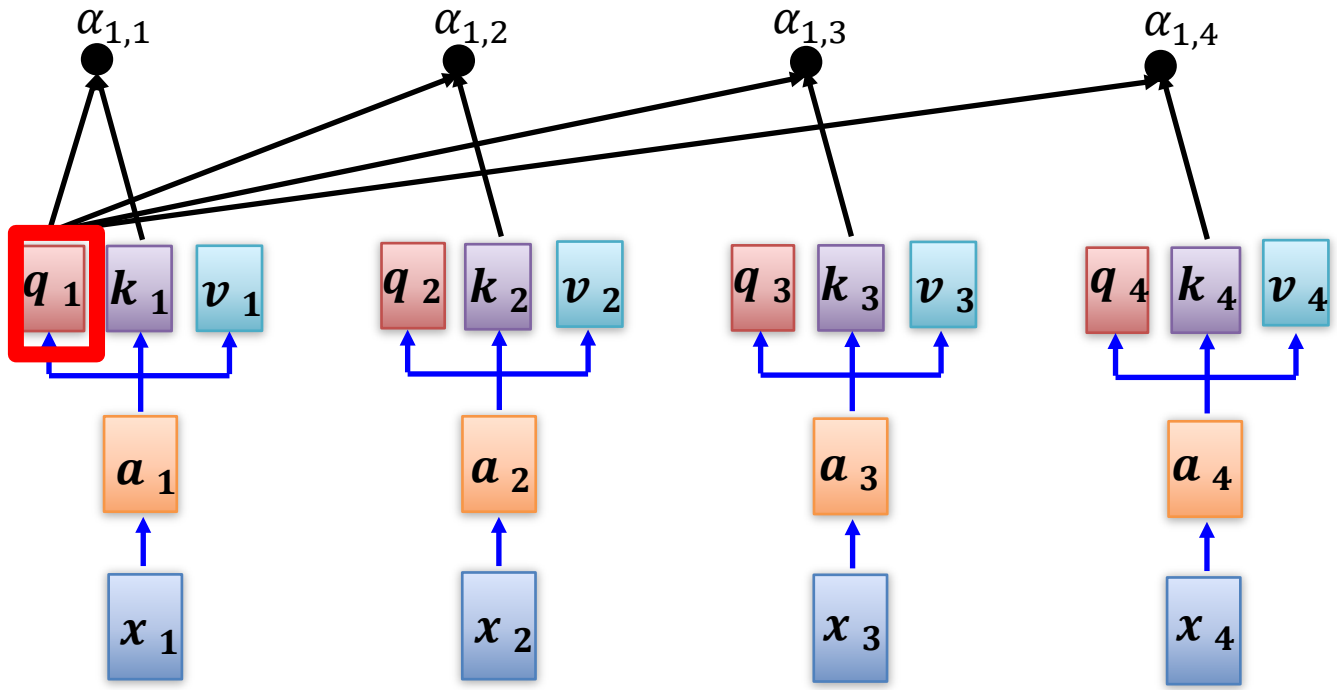
univ-cotedazur.fr

Make Self-Attention More Flexible: Different representations for the roles of query, key, value

Take each query q to pay attention to each key k

d is the dim of $\boldsymbol{q}$ and $\boldsymbol{k}$

Scaled Dot-Product Attention: $\alpha_{1,i} = \underbrace{\boldsymbol{q_1} \cdot \boldsymbol{k_i}}_{\text{dot product}} / \sqrt{d}$

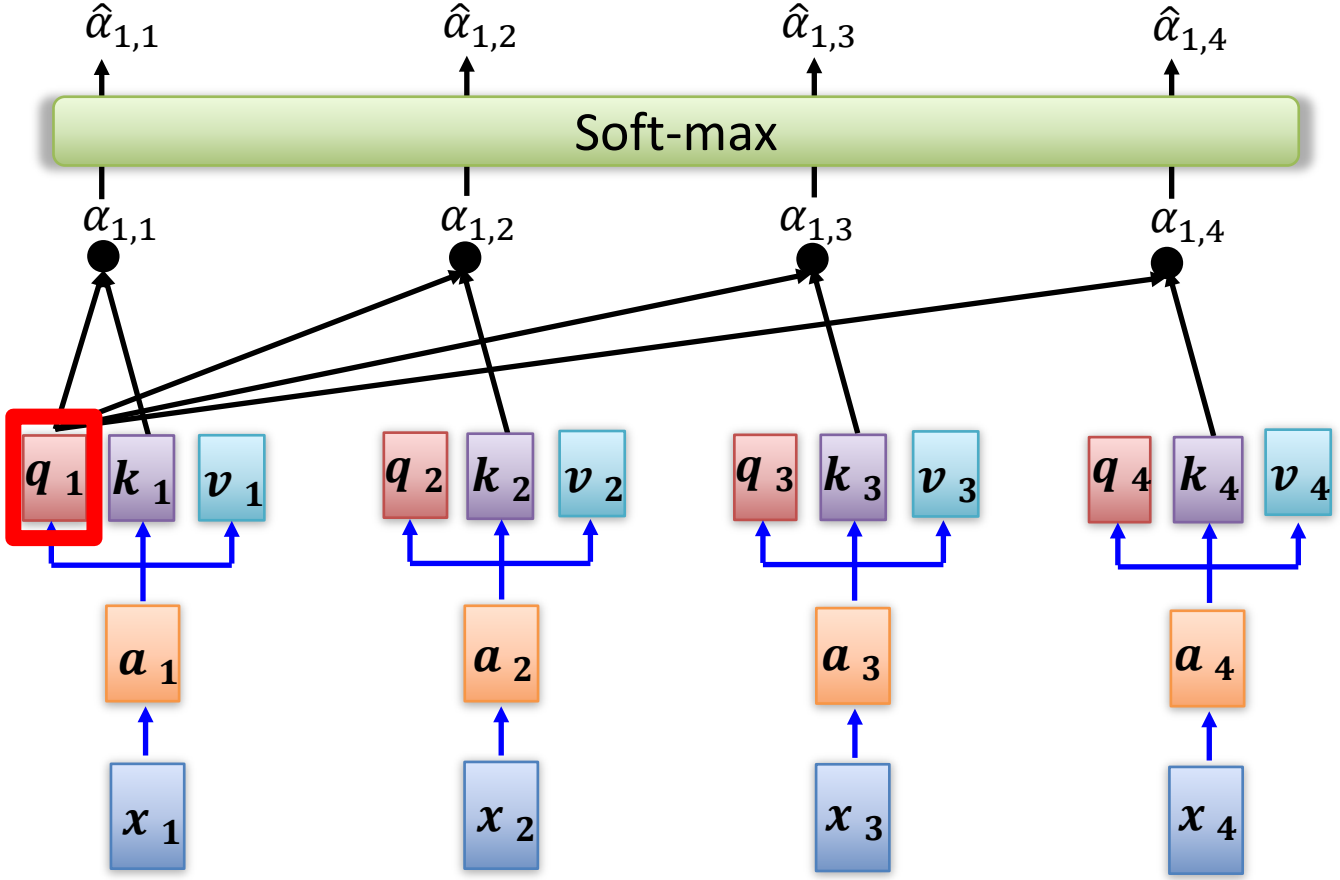


UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Make Self-Attention More Flexible: Different representations for the roles of query, key, value

$$\hat{\alpha}_{1,i} = exp(\alpha_{1,i}) / \sum_j exp(\alpha_{1,j})$$



Make Self-Attention More Flexible: Different representations for the roles of query, key, value

Scores d'attention =
Scores de corrélation

$$\alpha_{1,i} = \boldsymbol{q}^0{}_1 \cdot \boldsymbol{k}^0{}_i / \sqrt{d}$$

$\boldsymbol{q}$: query (to match with others)

$$\boldsymbol{q}_i = \boldsymbol{W}^q \boldsymbol{e}_i$$

$\boldsymbol{k}$: key (to be matched)

$$\boldsymbol{k}_i = \boldsymbol{W}^k \boldsymbol{e}_i$$

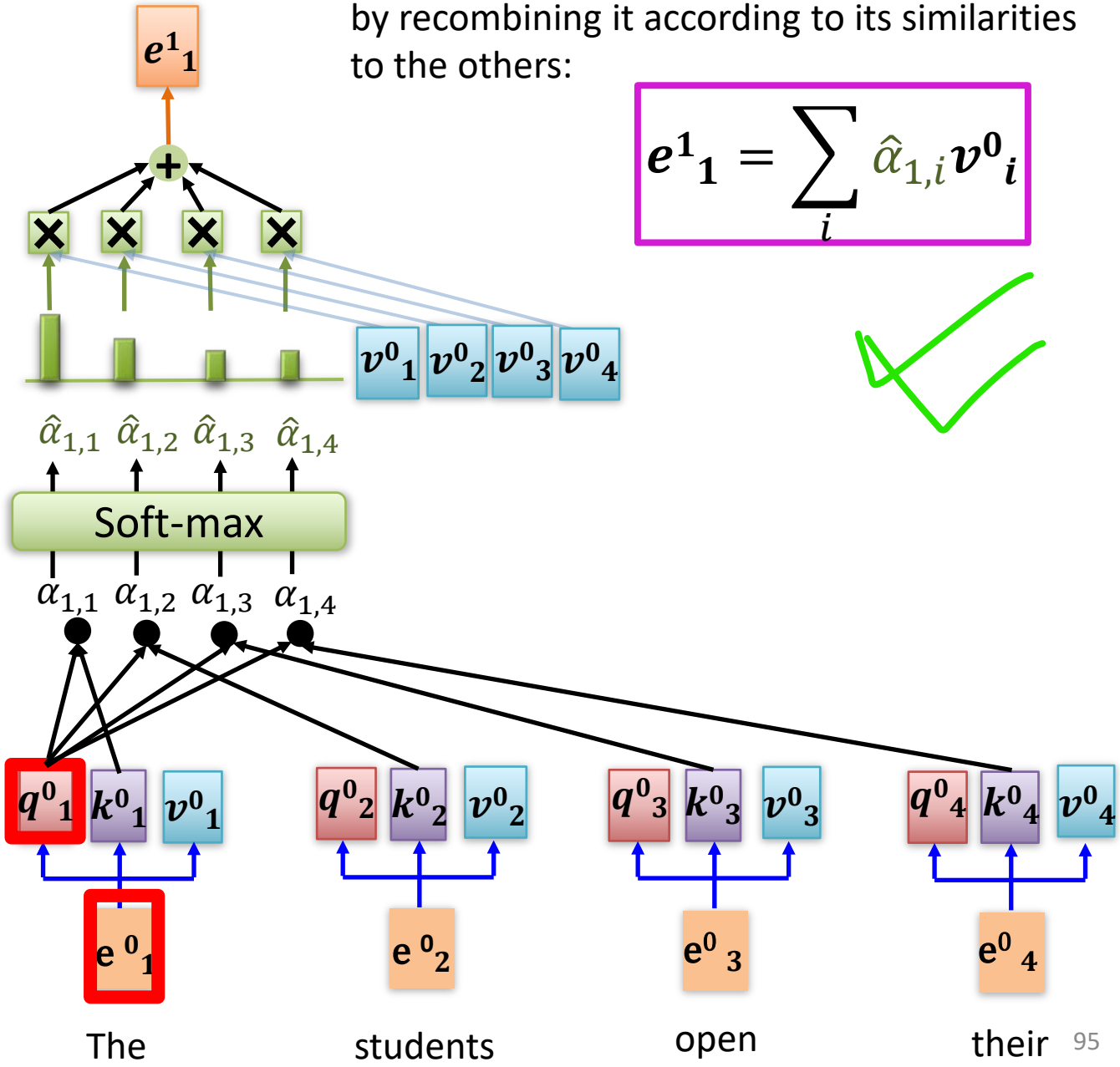
$\boldsymbol{v}$: representation to be extracted

$$\boldsymbol{v}_i = \boldsymbol{W}^v \boldsymbol{e}_i$$

(one can also take $\boldsymbol{q}^0{}_i = \boldsymbol{k}^0{}_i = \boldsymbol{v}^0{}_i = \boldsymbol{e}^0{}_i$)

Generate a new representation of this word by recombining it according to its similarities to the others:

$$\boldsymbol{e}^1{}_1 = \sum_i \hat{\alpha}_{1,i} \boldsymbol{v}^0{}_i$$



Credits: L. Sassatelli

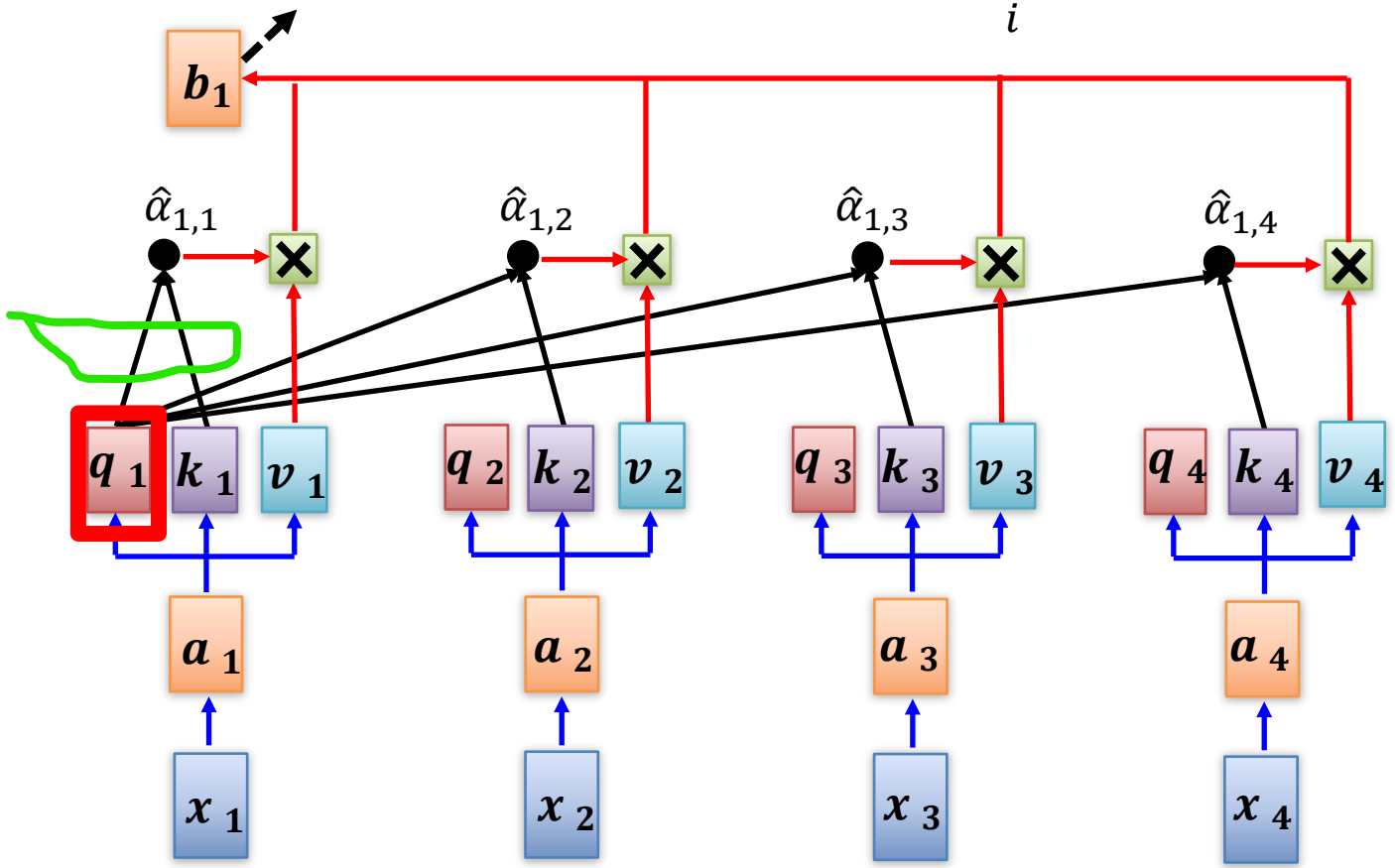
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Self-attention

Considering the whole sequence

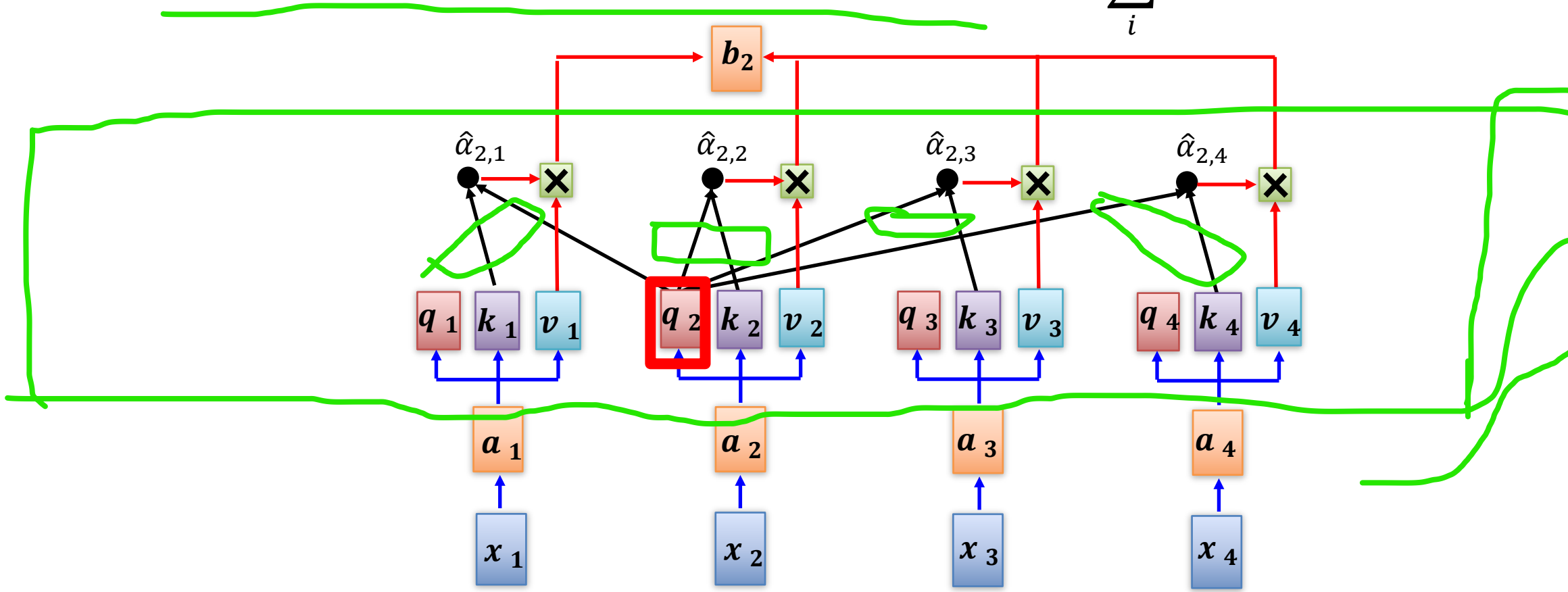
$$\boldsymbol{b_1} = \sum_i \hat{\alpha}_{1,i} \boldsymbol{v}_i$$



Self-attention

Take each query *q* to pay attention to each key *k*

$$b_2 = \sum_i \hat{\alpha}_{2,i} v_i$$

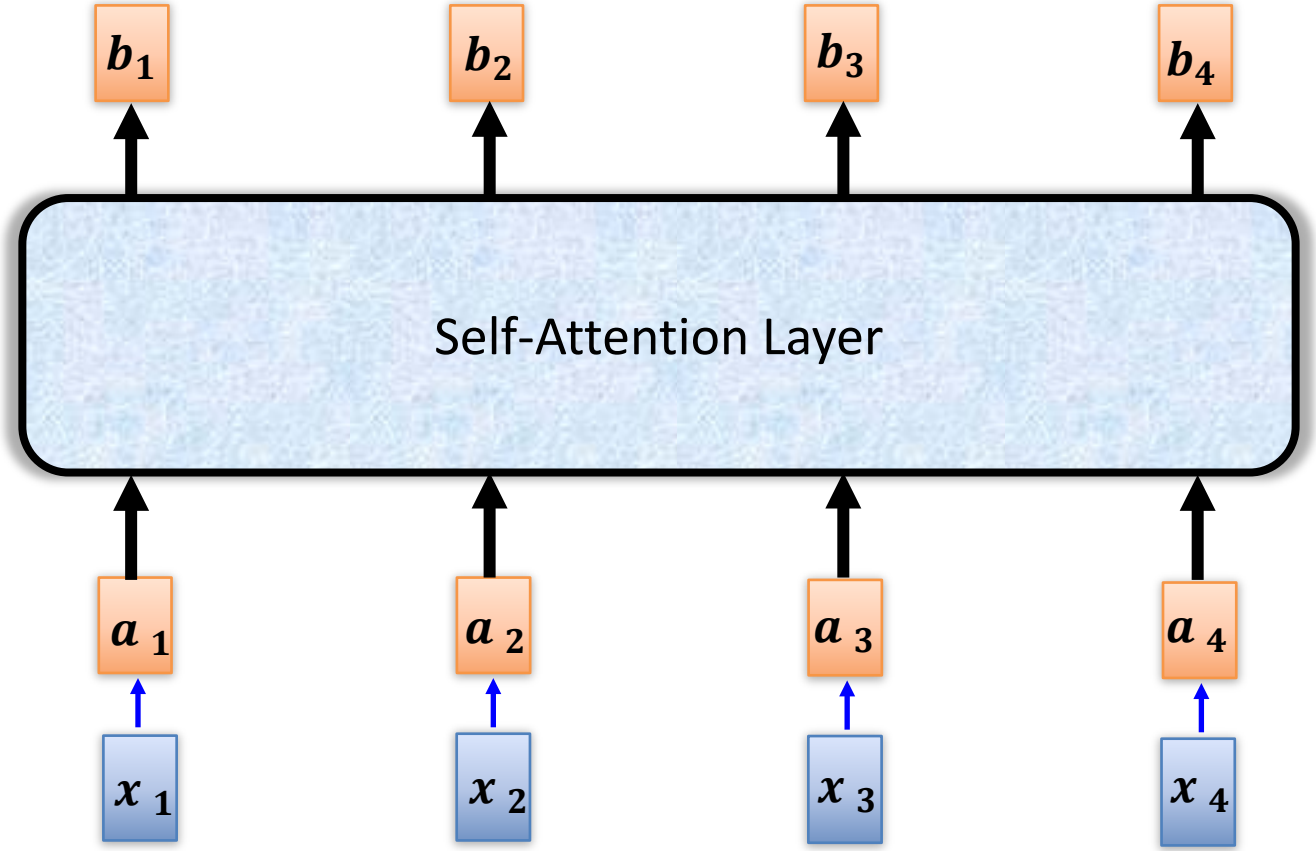


UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

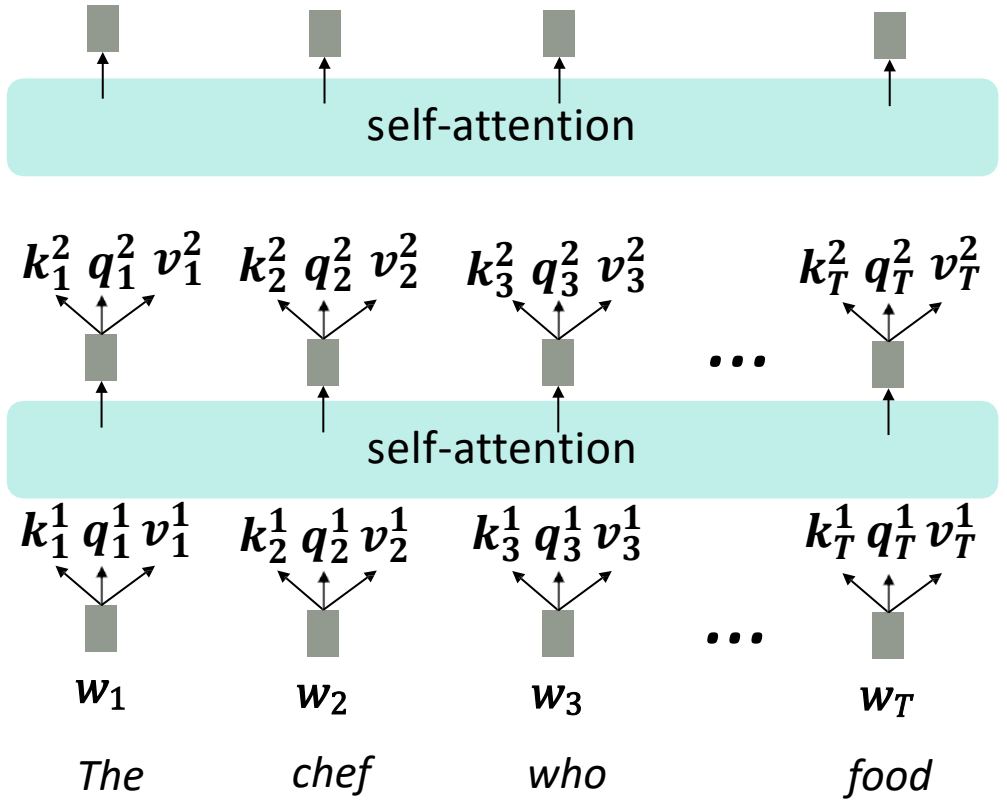
Self-attention

b_1 , b_2 , b_3 , b_4 can be parallelly computed.



Self-Attention

- In the diagram at the right, we have stacked self-attention blocks, like we might stack LSTM layers.
- Can self-attention be a drop-in replacement for recurrence?
- No. It has a few issues, which we'll go through.
- First, self-attention is an operation on **sets**. It has no inherent notion of order.



Self-attention doesn't know the order of its inputs.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Barriers and solutions for Self-Attention as a building block

Barriers

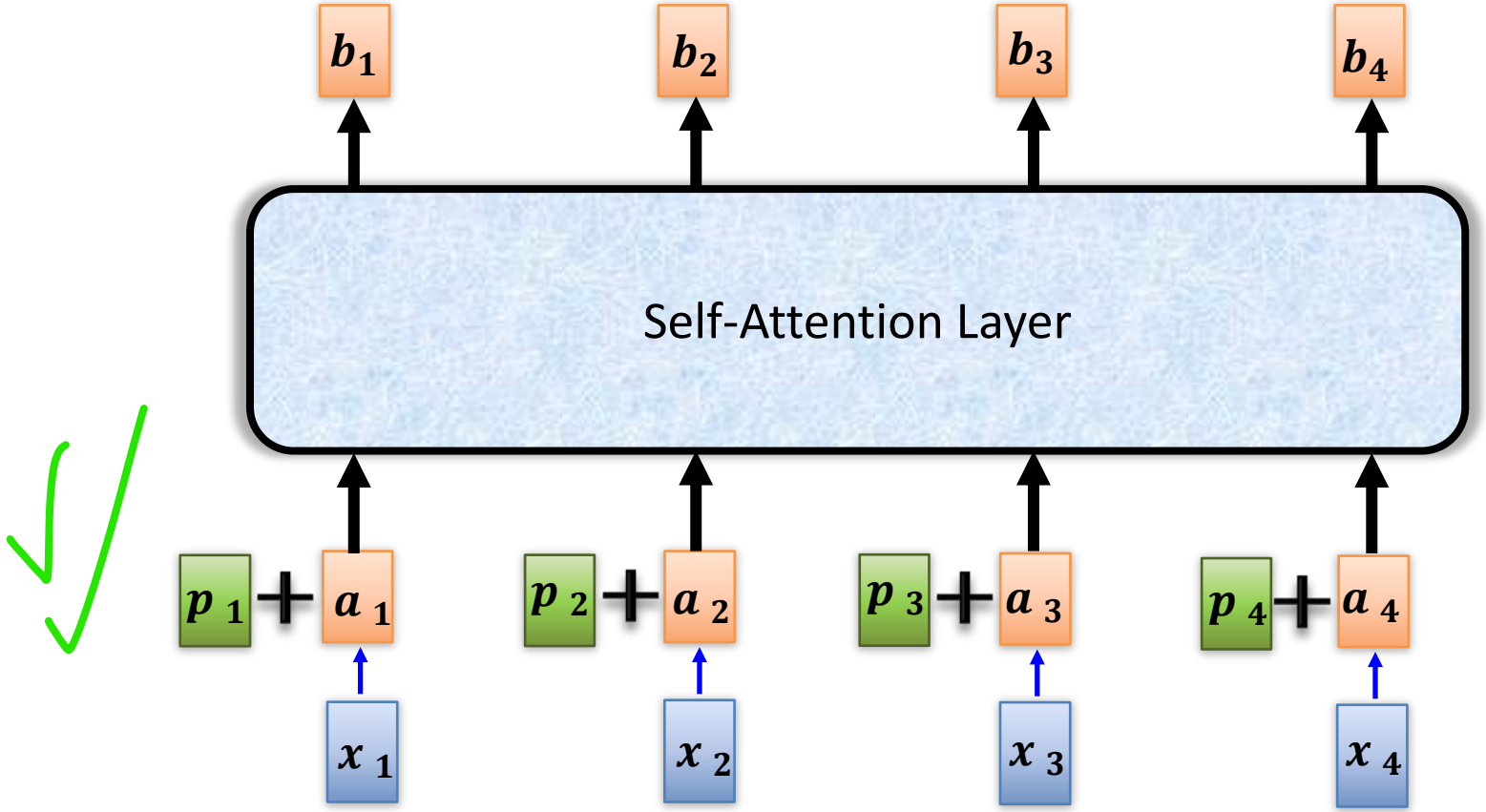
- Doesn't have an inherent notion of order!

Solutions

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Fixing the first self-attention problem:
Sequence order by adding position



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Barriers and solutions for Self-Attention as a building block

Barriers

- Doesn't have an inherent notion of order!
- No nonlinearities for deep learning! It's all just weighted averages

Solutions

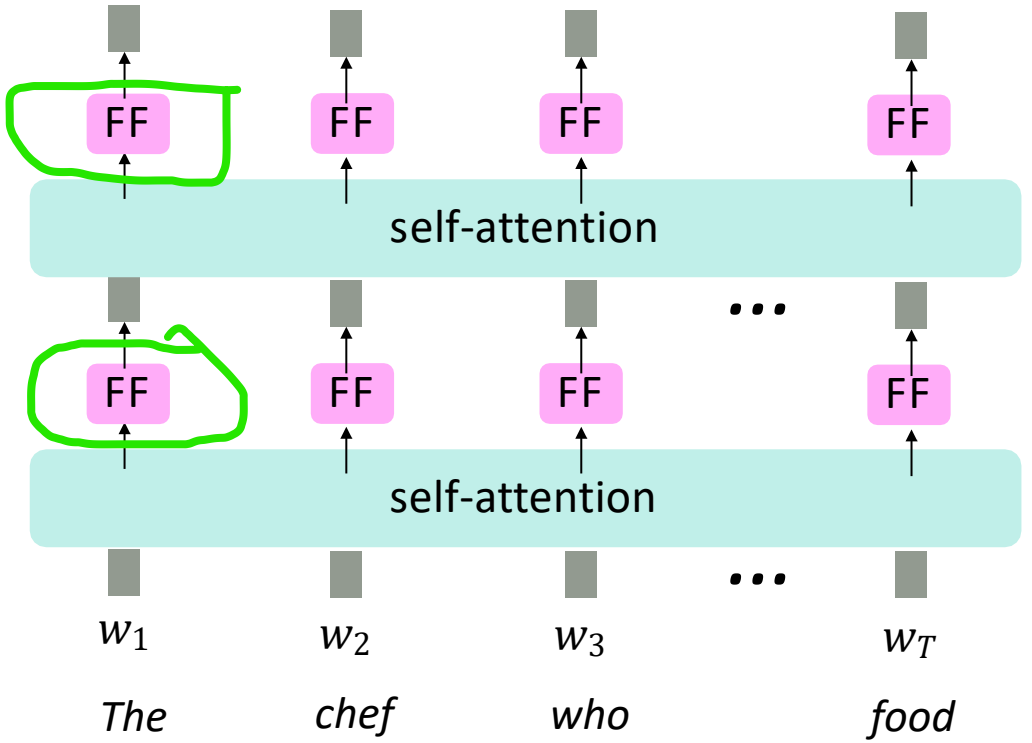
- Add position representations to the inputs

Adding nonlinearities in self-attention

- Note that there are no elementwise nonlinearities in self-attention; stacking more self-attention layers just re-averages **value** vectors

- Easy fix: add a **feed-forward network** to post-process each output vector.

$$m_i = MLP(\text{output}_i)$$
$$= W_2 * \text{ReLU}(W_1 \times \text{output}_i + b_1) + b_2$$

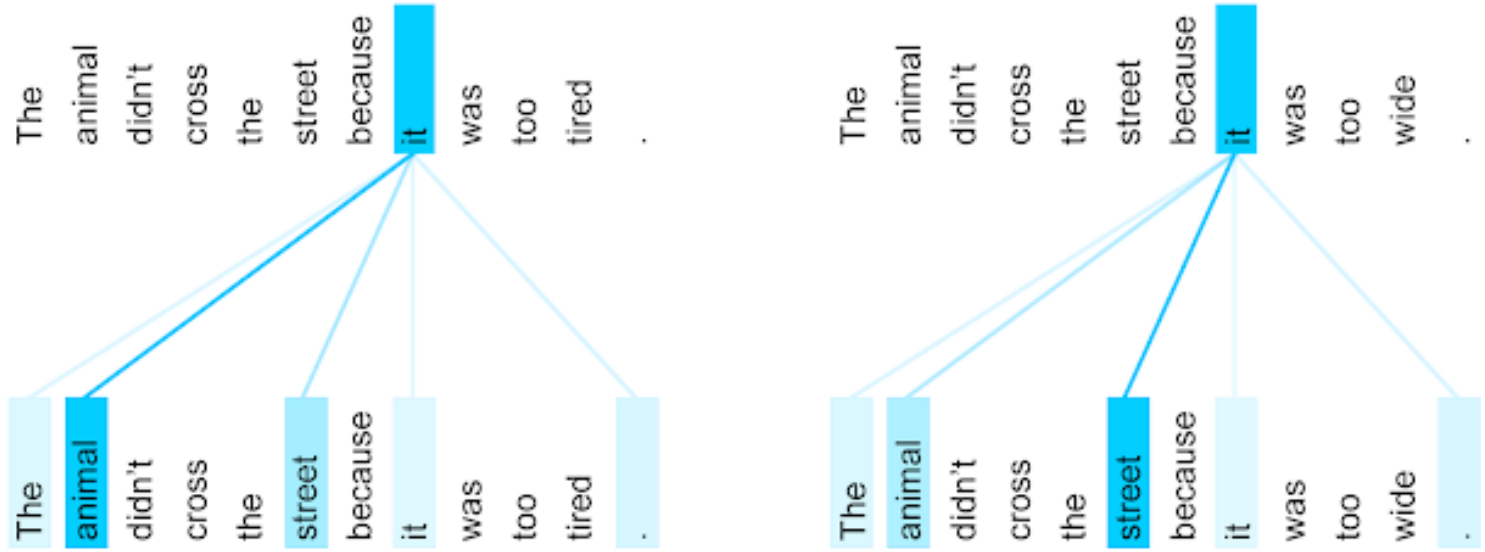


Intuition: the FF network processes the result of attention

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Attention Visualization



The encoder self-attention distribution for the word "it" from the 5th to the 6th layer of a Transformer trained on English to French translation (one of eight attention heads).

https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

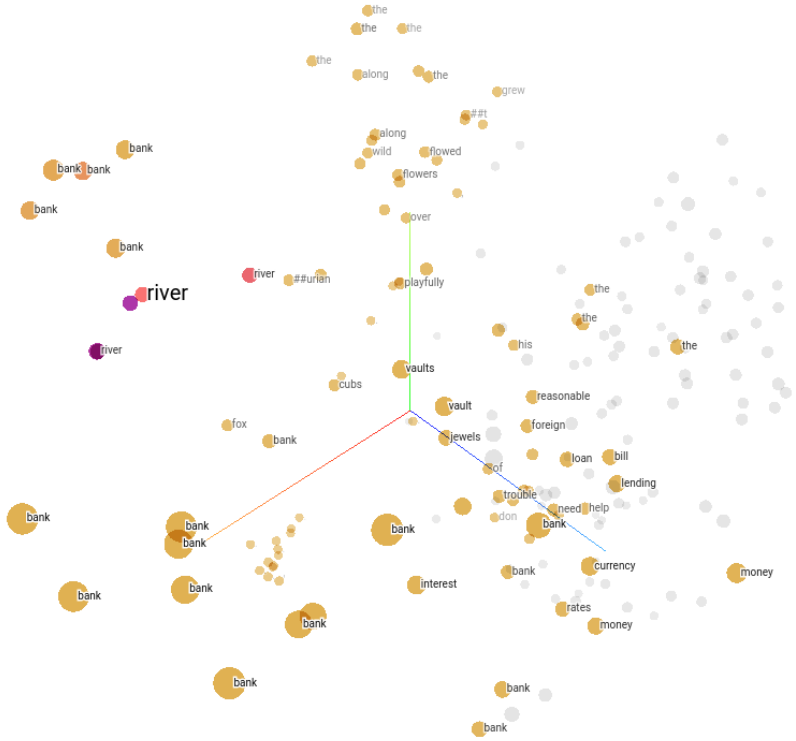
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Visualization of BERT embeddings

- A vector for the same word for each appearance:

```
sentences = ["bank",
        "he eventually sold the shares back to the bank at a
premium.",
        "the bank strongly resisted cutting interest rates.",
        "the bank will supply and buy back foreign currency.",
        "the bank is pressing us for repayment of the loan.",
        "the bank left its lending rates unchanged.",
        "the river flowed over the bank.",
        "tall, luxuriant plants grew along the river bank.",
        "his soldiers were arrayed along the river bank.",
        "wild flowers adorned the river bank.",
        "two fox cubs romped playfully on the river bank.",
        "the jewels were kept in a bank vault.",
        "you can stow your jewellery away in the bank.",
        "most of the money was in storage in bank vaults.",
        "the diamonds are shut away in a bank vault somewhere.",
        "thieves broke into the bank vault.",
        "can I bank on your support?",
        "you can bank on him to hand you a reasonable bill for your
services.",
        "don't bank on your friends to help you out of trouble.",
        "you can bank on me when you need money.",
        "i bank on your help."
        ]
```



https://projector.tensorflow.org/?config=https%3A%2F%2Fgist.githubusercontent.com%2Farminmirrezai%2F2c59f675a997fef20d59851291f8c268%2Fraw%2Fc542be898dc77aad2d593ebcd7b60ac302b71359%2Fgistfile1.txt

Credits: Prof. L. Sassatelli

Necessities for a self-attention building block:

- **Self-attention**:
  - the basis of the method.
- **Position representations**:
  - Specify the sequence order, since self-attention is an unordered function of its inputs.
- **Nonlinearities**:
  - At the output of the self-attention block
  - Frequently implemented as a simple feed-forward network.
- **Masking**:
  - In order to parallelize operations while not looking at the future.
  - Keeps information about the future from "leaking" to the past.

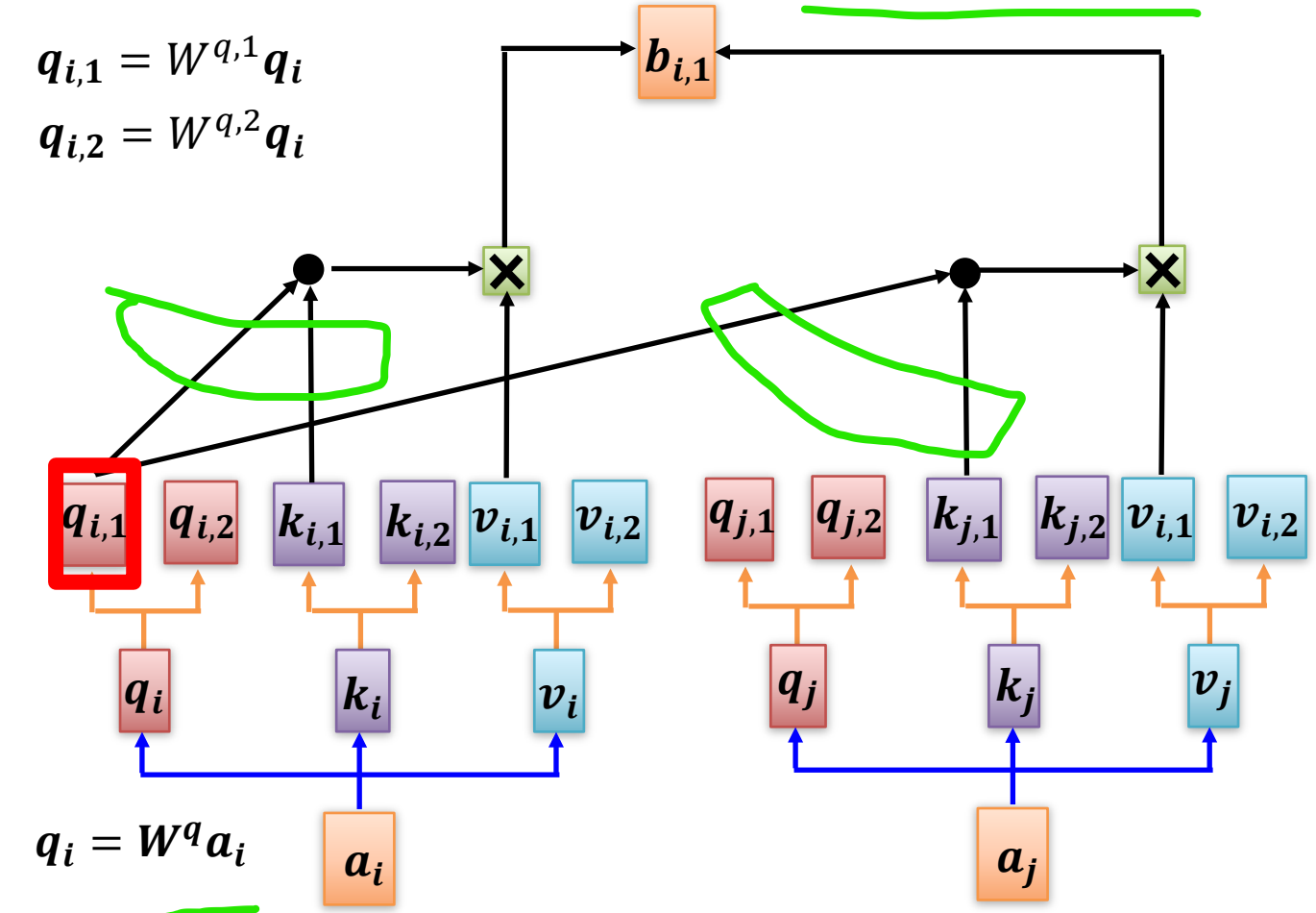
- These are the **main steps to build Transformers**…But this is not yet the **Transformer** model we've been hearing about!

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The Transformer Encoder:
Multi-headed attention

(2 heads as example)

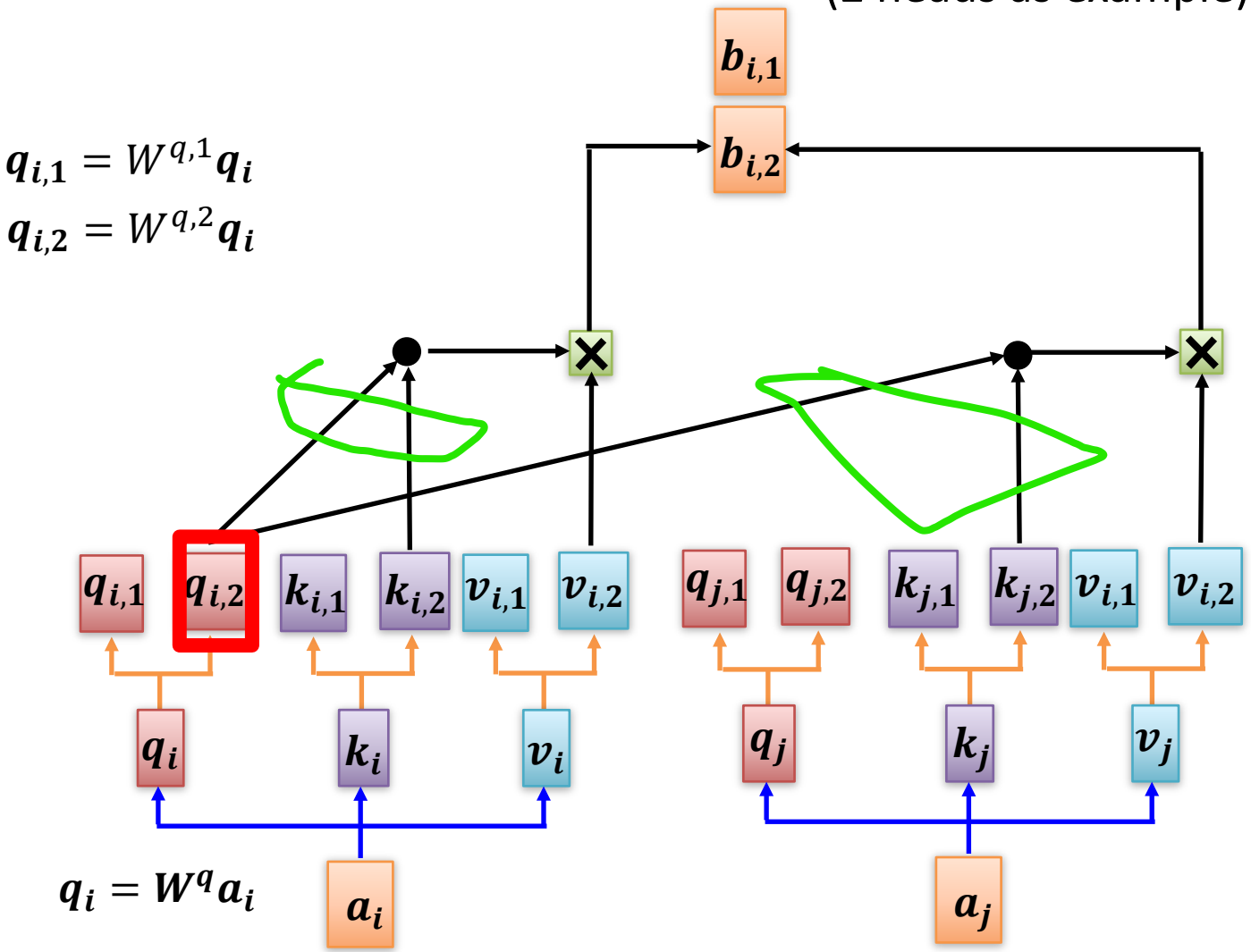


UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The Transformer Encoder: **Multi-headed attention**

(2 heads as example)

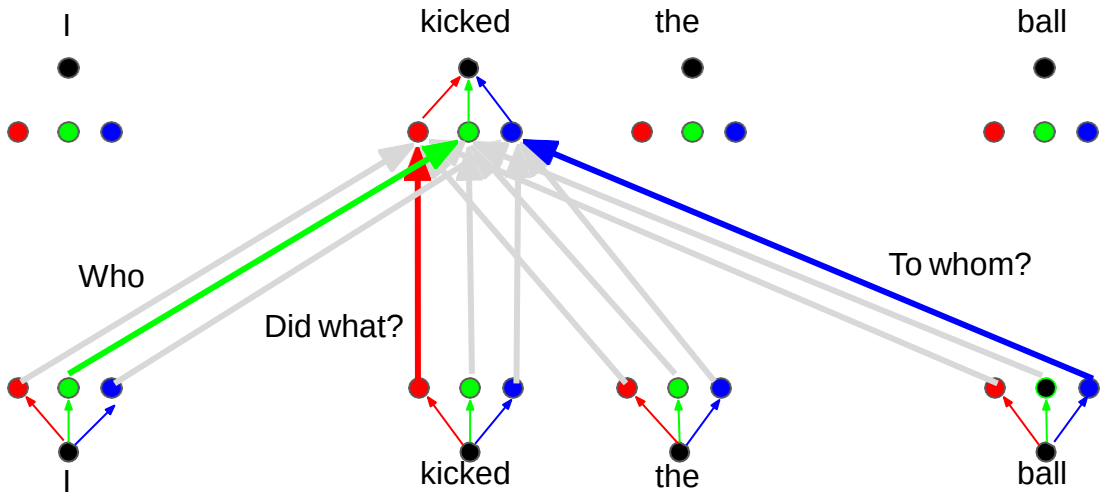


The principle of multi-head attention in transformers is similar to the one of multi-kernels in convolution layers of a CNN (i.e. multi-feature maps, yellow, red, blue, green…)

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Parallel attention heads

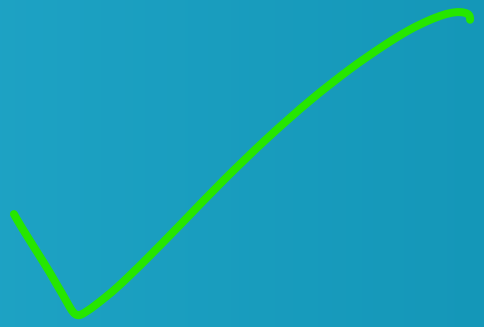


Ashish Vaswani

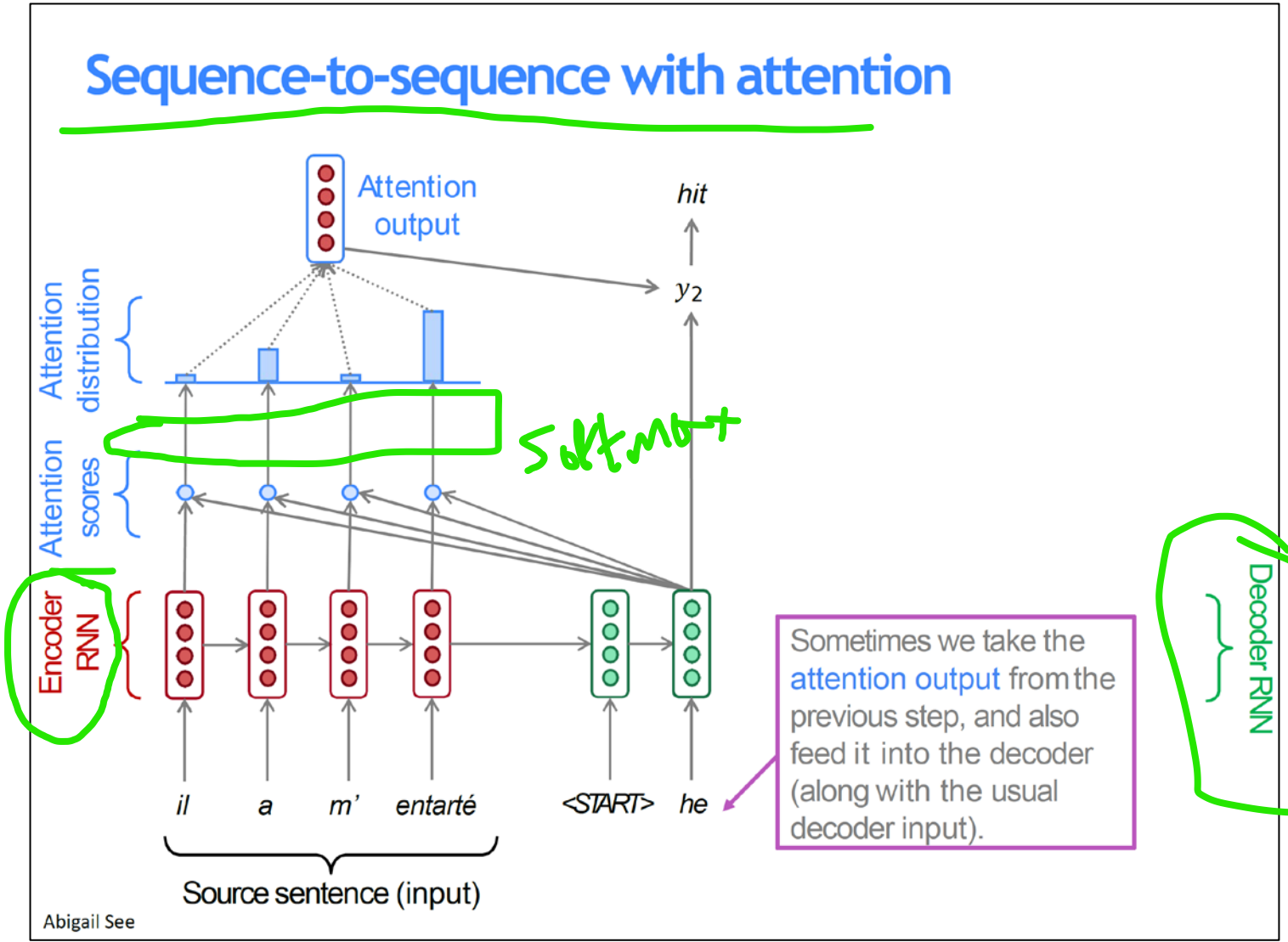
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

TRANSFORMERS



Before Transformers…



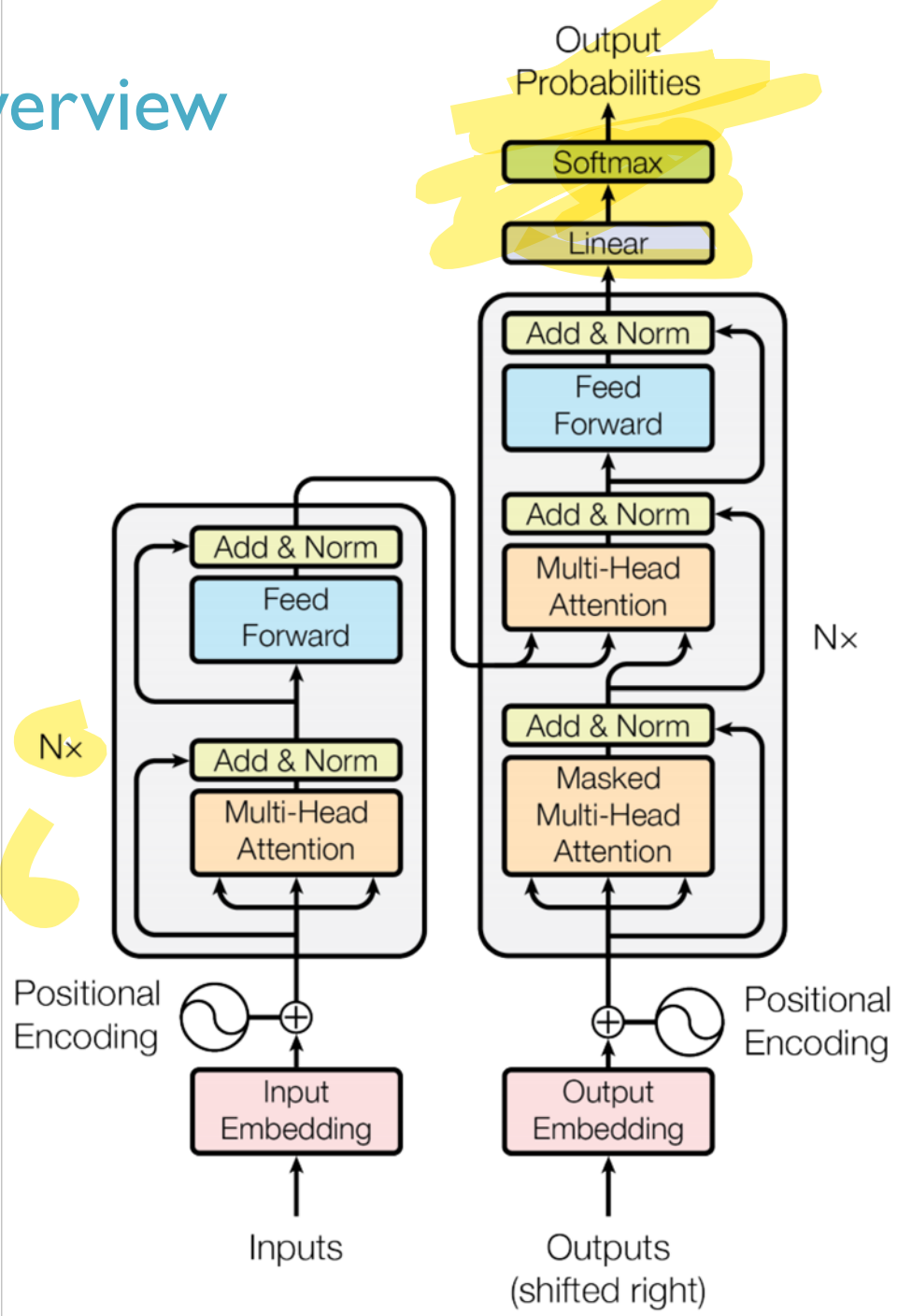
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Transformer Overview

Attention is all you need. 2017. Aswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, Polosukhin
https://arxiv.org/pdf/1706.03762.pdf

- **Non-recurrent sequence-to-sequence encoder-decoder model**
- **Task: machine translation with parallel corpus**
- Predict each translated word
- Final cost/error function is standard cross-entropy error on top of a softmax classifier



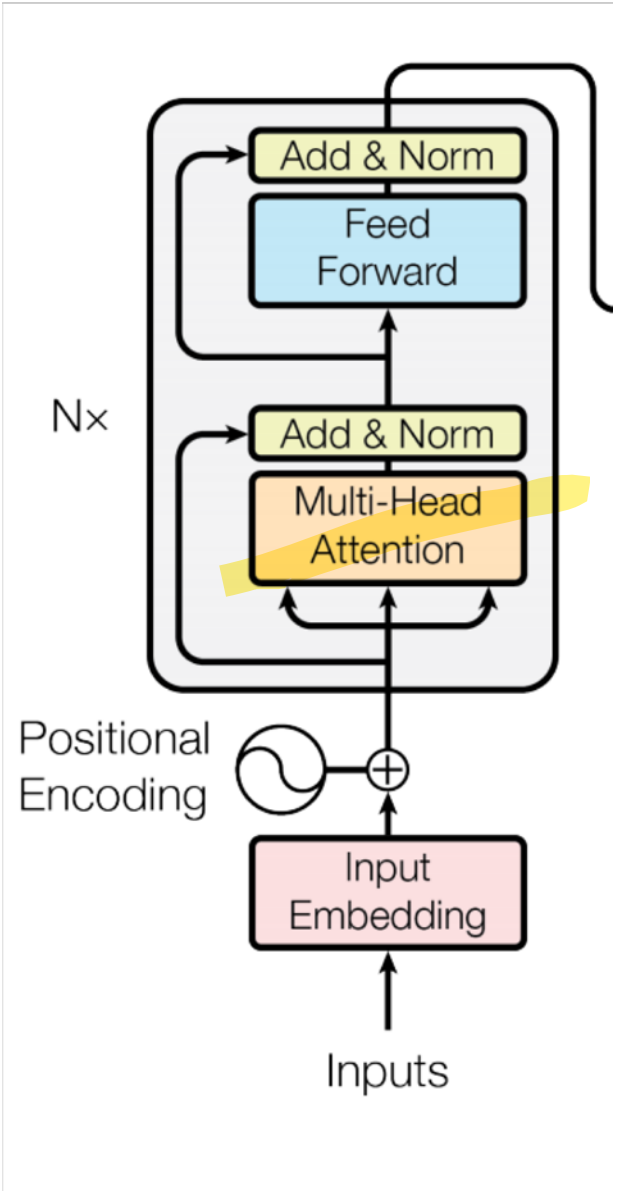
This and related figures from paper ⇑

Transformer Encoder

- For encoder, at each block, we use the same ***q***, ***k*** and ***v*** from the previous layer:

$$q_i^l = q_i^{l+1}$$
$$k_i^l = k_i^{l+1}$$
$$v_i^l = v_i^{l+1}$$

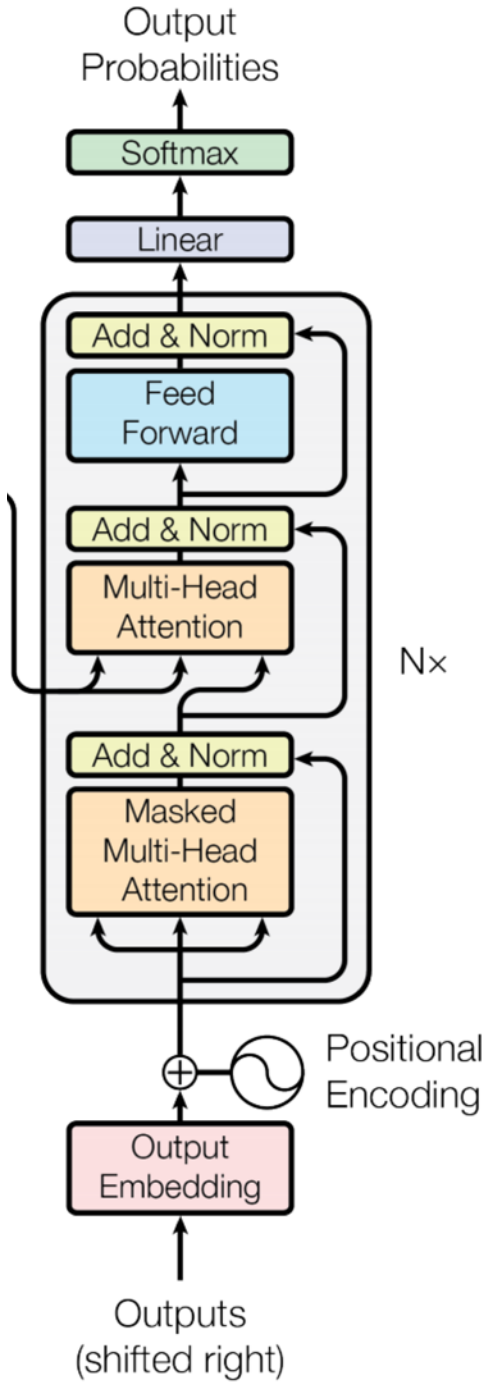
- Blocks are repeated 6 times (in vertical stack)



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Transformer Decoder



Christopher Manning

Barriers and solutions for Self-Attention as a building block

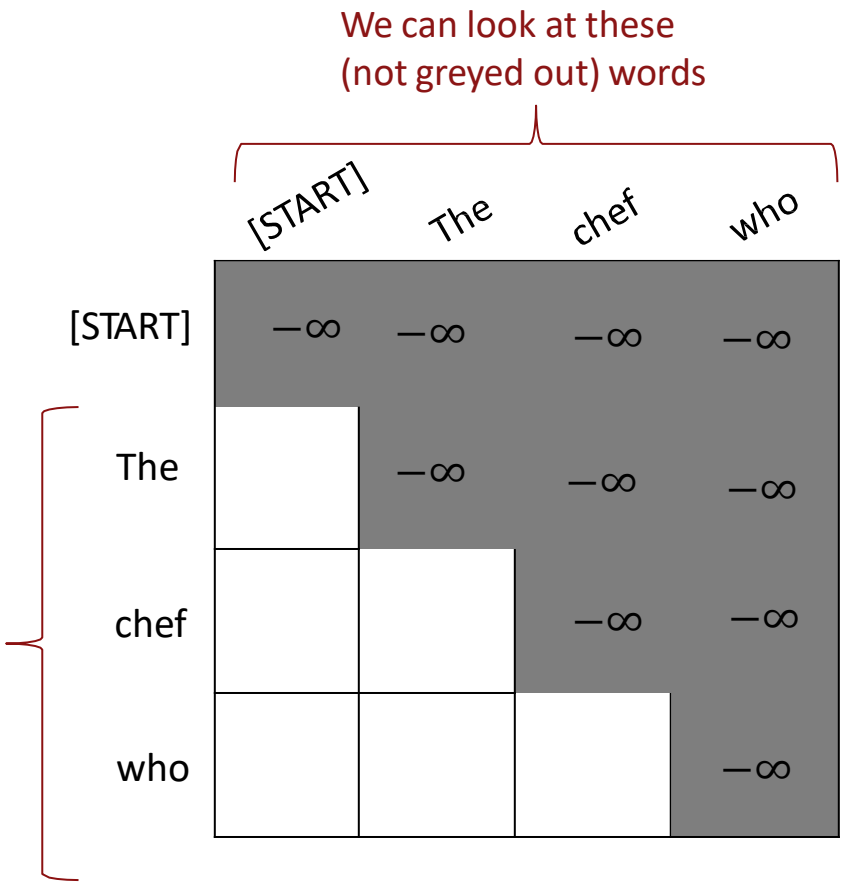
Barriers	Solutions
• Doesn't have an inherent notion of order!	• Add position representations to the inputs
• No nonlinearities for deep learning magic! It's all just weighted averages	• Easy fix: apply the same feedforward network to each self-attention output.

• Need to ensure we don't "look at the future" when predicting a sequence
  • Like in machine translation
  • Or language modeling	

Masking the future in self-attention

- To use self-attention in **decoders**, we need to ensure we can't peek at the future.
- At every timestep, we could change the set of **keys and queries** to include only past words. (Inefficient!)
- To enable parallelization, we **mask out attention** to future words by setting attention scores to $-\infty$.

$$e_{ij} = \begin{cases} q_i^\top k_j, j < i \\ -\infty, j \geq i \end{cases}$$



Barriers and solutions for Self-Attention as a building block

Barriers

- Doesn't have an inherent notion of order!
- No nonlinearities for deep learning magic! It's all just weighted averages
- Need to ensure we don't "look at the future" when predicting a sequence
  - Like in machine translation
  - Or language modeling

Solutions

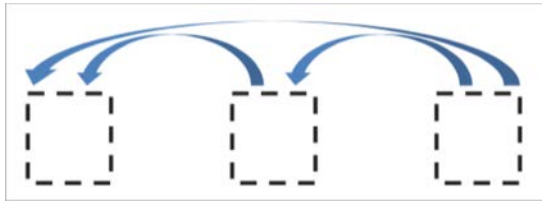
- Add position representations to the inputs
- Easy fix: apply the same feedforward network to each self- attention output.
- Mask out the future by artificially setting attention weights to 0!

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

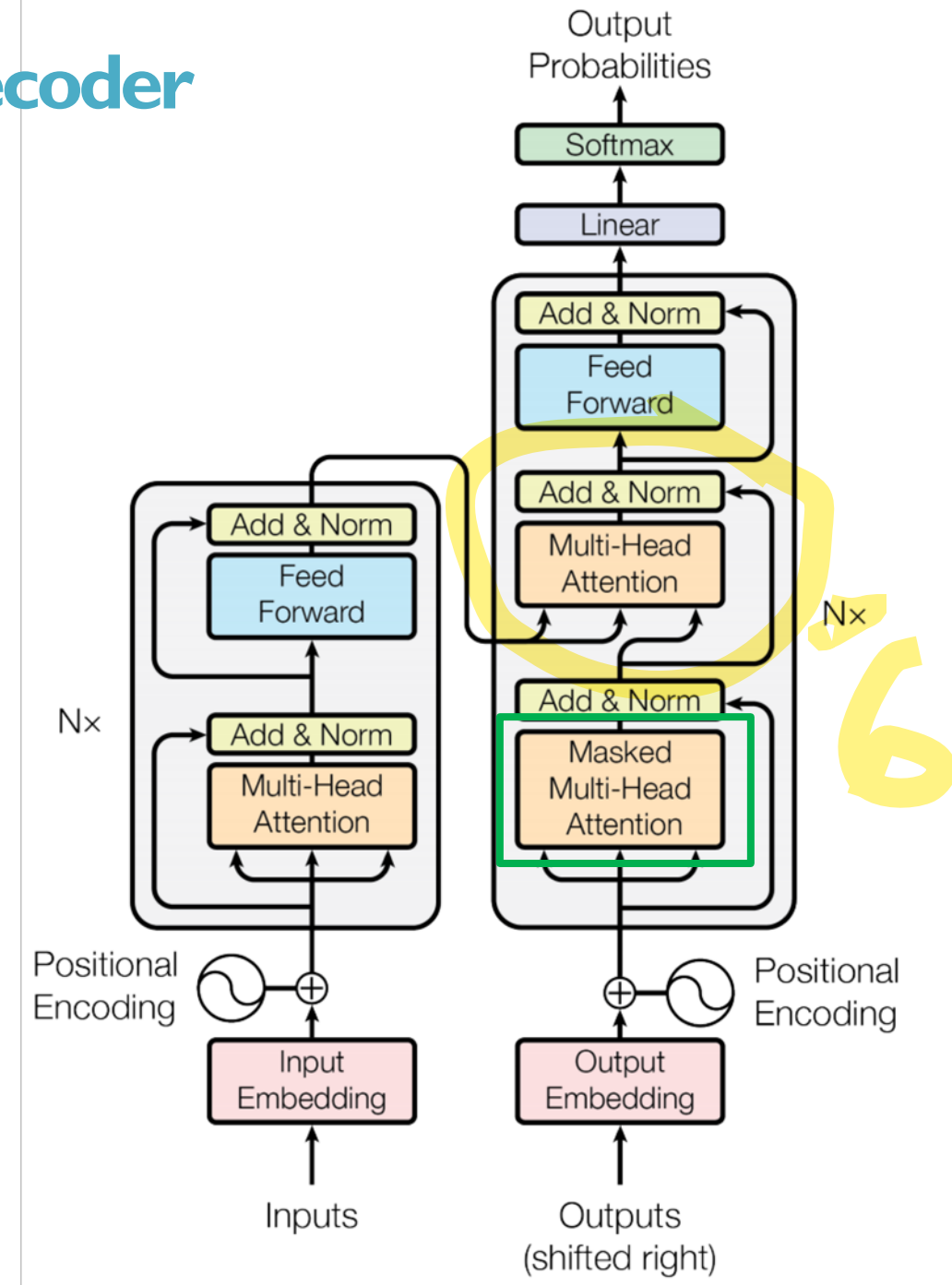
Transformer Decoder

- 2 sublayer changes in decoder
- Masked decoder self-attention on previously generated outputs:



- Encoder-Decoder Attention, where queries come from previous decoder layer and keys and values come from output of encoder, **Cross-Attention**

Blocks repeated 6 times also

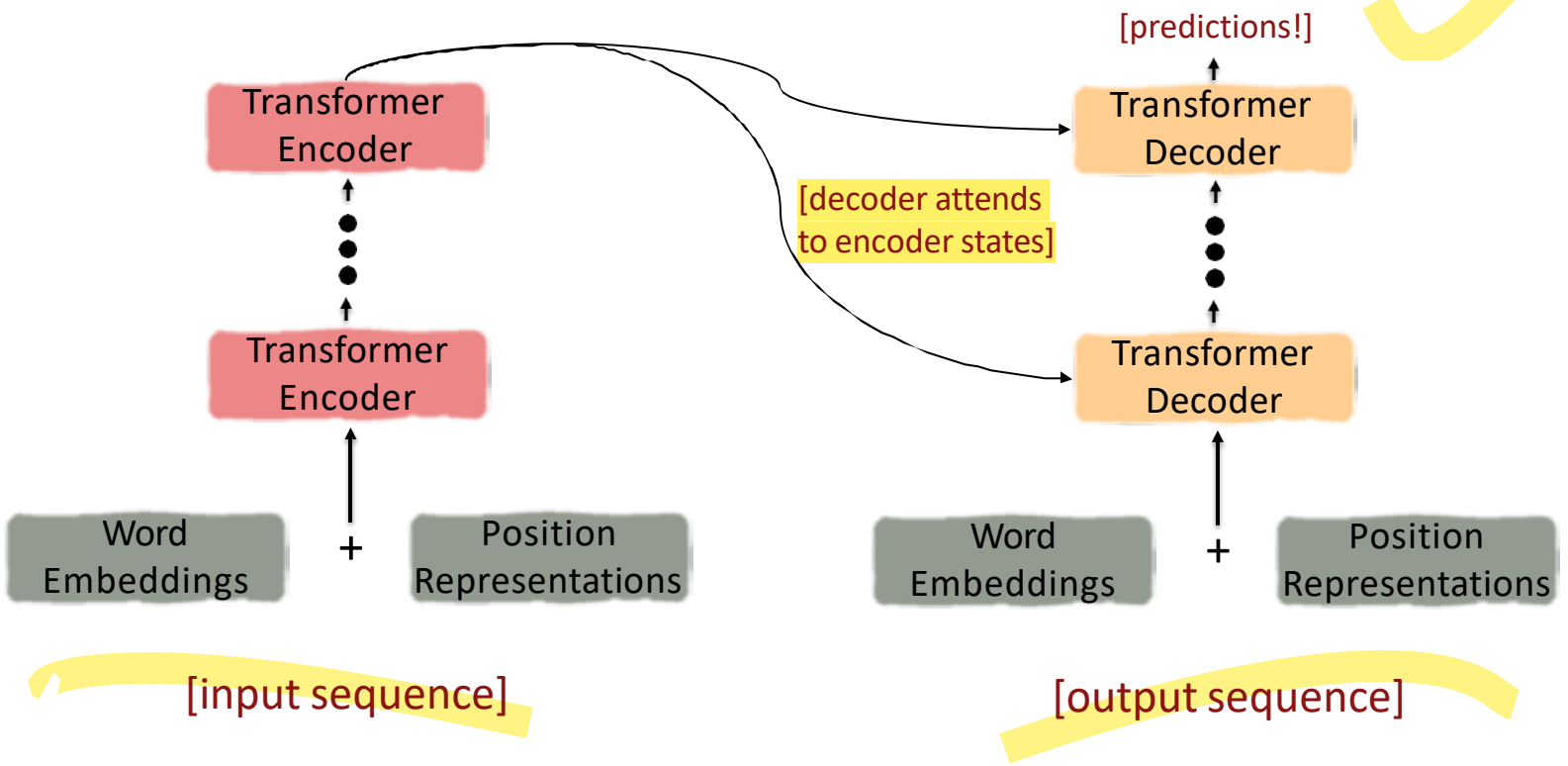


UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The Transformer Encoder-Decoder [Vaswani et al., 2017]

Another look at the Transformer Encoder and Decoder Blocks at a high level



UNIVERSITÉ
CÔTE D'AZUR

Transformers: tips and tricks

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The Transformer Encoder-Decoder [Vaswani et al., 2017]

Next, let's look at the Transformer Encoder and Decoder Blocks

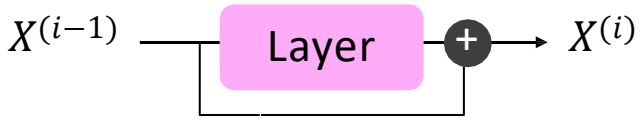
What's left in a Transformer Encoder Block that we haven't covered?

1. **Key-query-value attention:** How do we get the k, q, v vectors from a single word embedding?
2. **Multi-headed attention**: Attend to multiple places in a single layer!
3. **Tricks to help with training!**
    1. Residual connections
    2. Layer normalization
    3. Scaling the dot product
    4. These tricks **don't improve** what the model is able to do; **they help improve the training process**

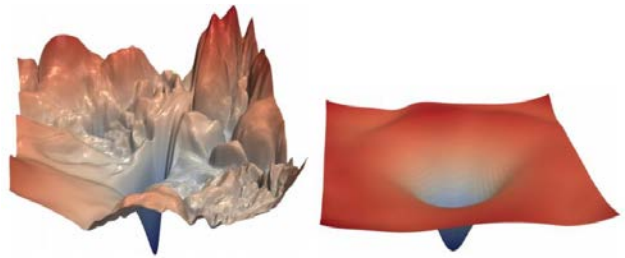
The Transformer Encoder:
Residual connections [He et al., 2016]

- **Residual connections** are a trick to help models train better.
  - Instead of $X^{(i)} = \text{Layer}(X^{(i-1)})$ (where i represents the layer)

  - We let $X^{(i)} = X^{(i-1)} + \text{Layer}(X^{(i-1)})$ (so we only have to learn "the residual" from the previous layer)



  - Residual connections are thought to make the loss landscape considerably smoother (thus easier training!)



[no residuals] [residuals]

[Loss landscape visualization, Li et al., 2018, on a ResNet]

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The Transformer Encoder:
Layer normalization [Ba et al., 2016]

- **Layer normalization** is a trick to help models train faster (has been applied to RNN first).
- It's a ''Batch normalization'' when the batch is size = 1.
- Idea: cut down on uninformative variation in hidden vector values by normalizing to unit mean and standard deviation **within each layer**.
  - LayerNorm's success may be due to its normalizing gradients [Xu et al., 2019]
- Let $x \in \mathbb{R}^d$ be an individual (word) vector in the model.
- Let $\mu = \Sigma_{j=1}^{d} x_j$; this is the mean; $\mu \in \mathbb{R}$.
  - Let $\sigma = \sqrt{\frac{1}{d}\sum_{j=1}^{d}(x - \mu)^2}$; this is the standard deviation; $\sigma \in \mathbb{R}$.
  - Let $\gamma \in \mathbb{R}^d$ and $\beta \in \mathbb{R}^d$ be learned "gain" and "bias" parameters. (Can omit!)
  - Then layer normalization computes:

$$\text{output} = \frac{x - \mu}{\sqrt{\sigma + \epsilon}} * \gamma + \beta$$

Normalize by scalar mean and variance

Modulate by learned elementwise gain and bias

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The Transformer Encoder:
Scaled Dot Product [Vaswani et al., 2017]

- **"Scaled Dot Product"** attention is a final variation to aid in Transformer training.
- When dimensionality d becomes large, dot products between vectors become large, inputs to the softmax function can be large, making gradients small.

- Instead of the self-attention function we've seen:

$$\text{output}_P = \text{softmax}\left(XQ_P K_P^\top X^\top\right) * XV_P$$

- We divide the attention scores by $\sqrt{d/h}$, to stop the scores from becoming large just as a function of d/h (The dimensionality divided by the number of heads.)

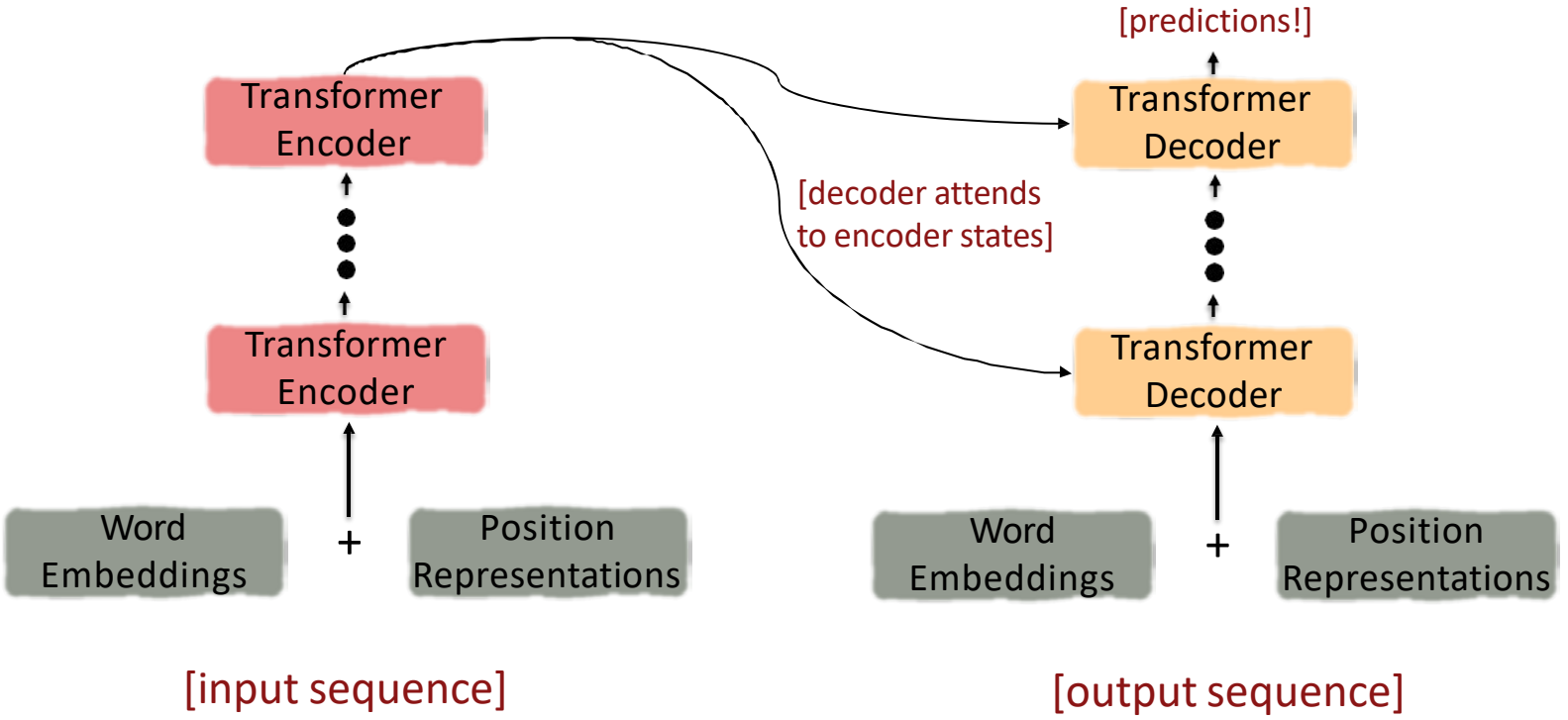
$$\text{output}_P = \text{softmax}\left(\frac{XQ_P K_P^\top X^\top}{\sqrt{d/h}}\right) * XV_P$$

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The Transformer Encoder-Decoder [Vaswani et al., 2017]

Looking back at the whole model, zooming in on an Encoder block:

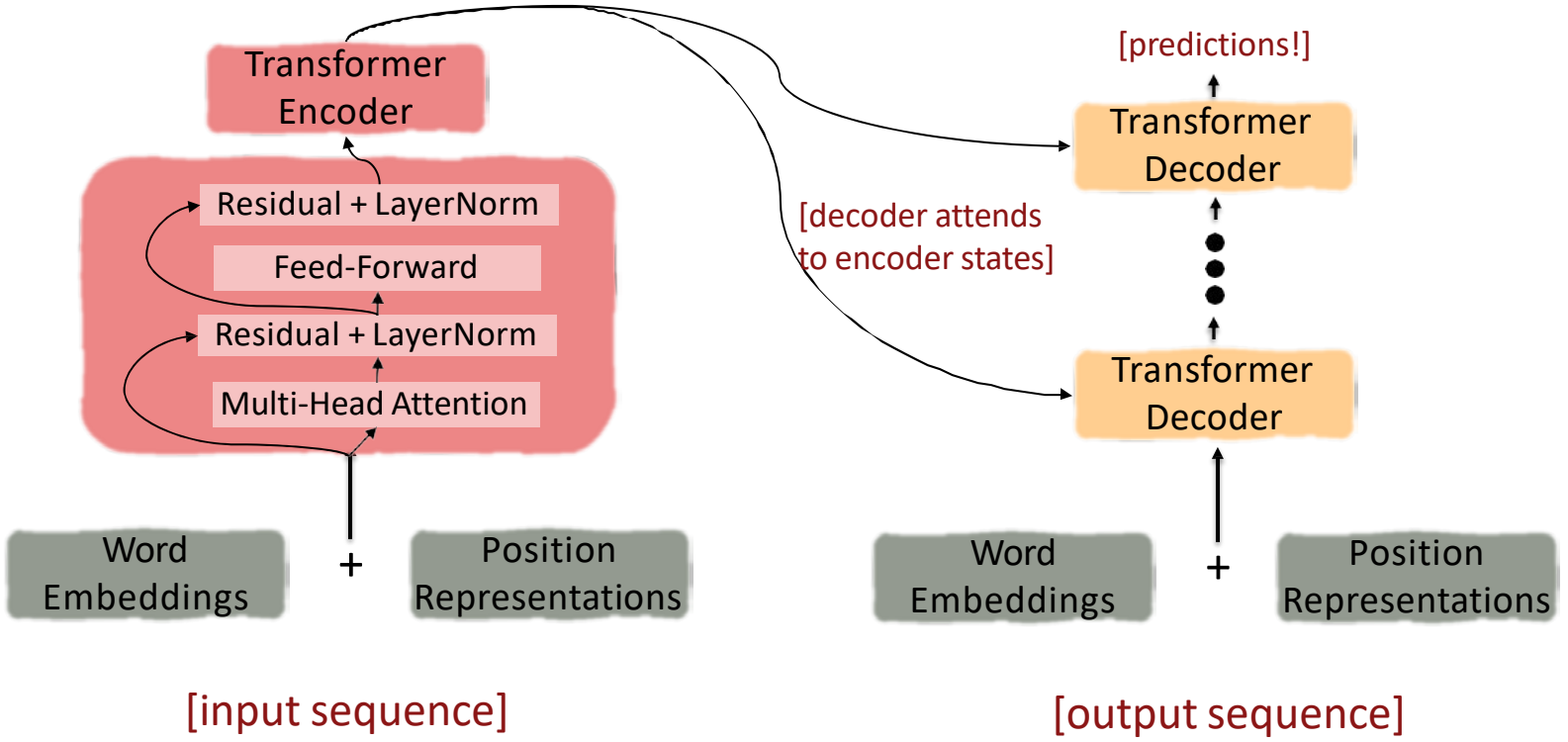


UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The Transformer Encoder-Decoder [Vaswani et al., 2017]

Looking back at the whole model, zooming in on an Encoder block:



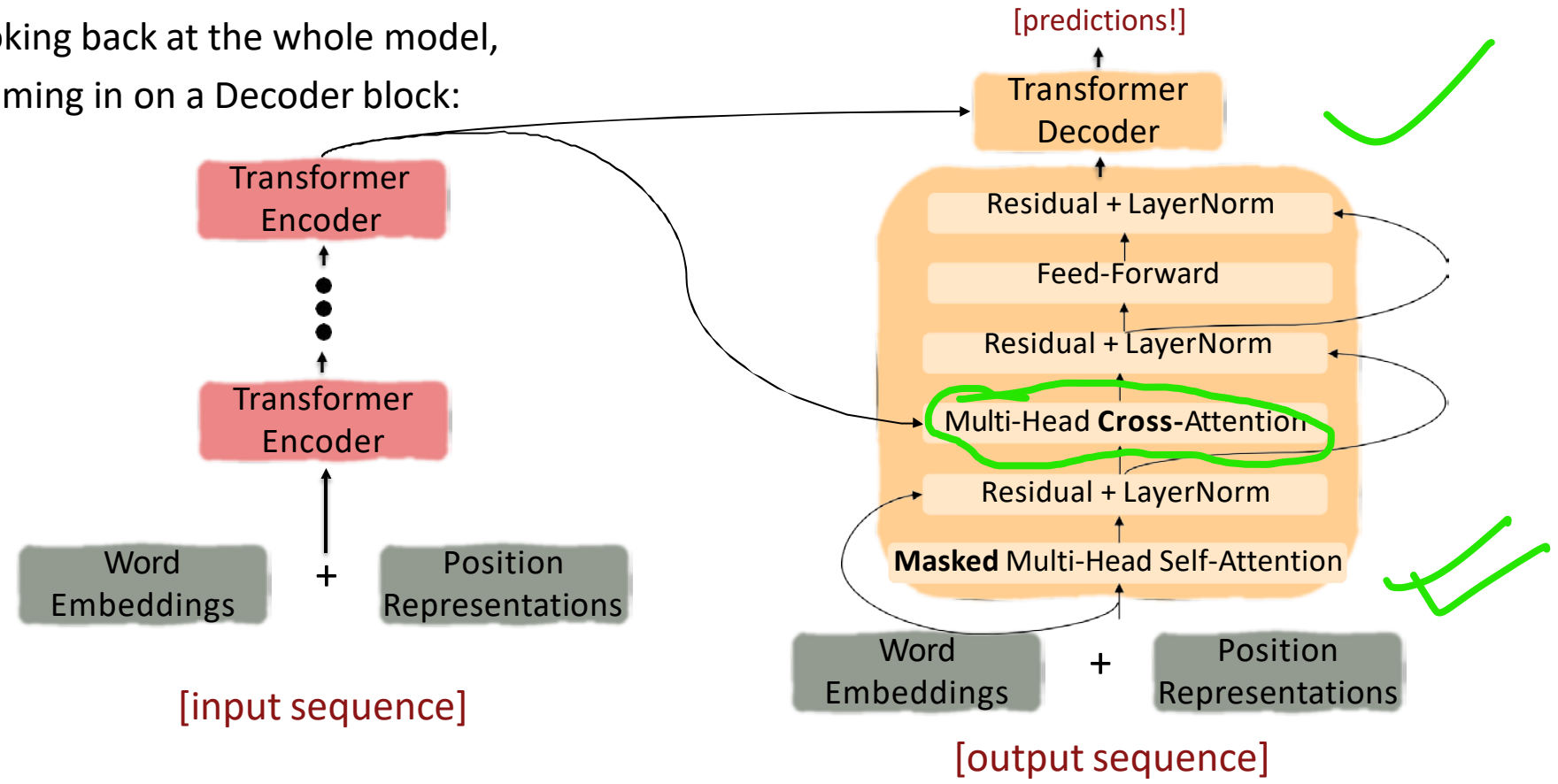
UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The Transformer Encoder-Decoder

[Vaswani et al., 2017]

Looking back at the whole model, zooming in on a Decoder block:



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Great Results with Transformers

Next, document generation!

Model	Test perplexity	ROUGE-L
seq2seq-attention, $L = 500$	5.04952	12.7
Transformer-ED, $L = 500$	2.46645	34.2
Transformer-D, $L = 4000$	2.22216	33.6
Transformer-DMCA, no MoE-layer, $L = 11000$	2.05159	36.2
Transformer-DMCA, MoE-128, $L = 11000$	1.92871	37.9
Transformer-DMCA, MoE-256, $L = 7500$	1.90325	38.8

The old standard

Transformers all the way down.

[Liu et al., 2018]; WikiSum dataset

UNIVERSITÉ
CÔTE D'AZUR

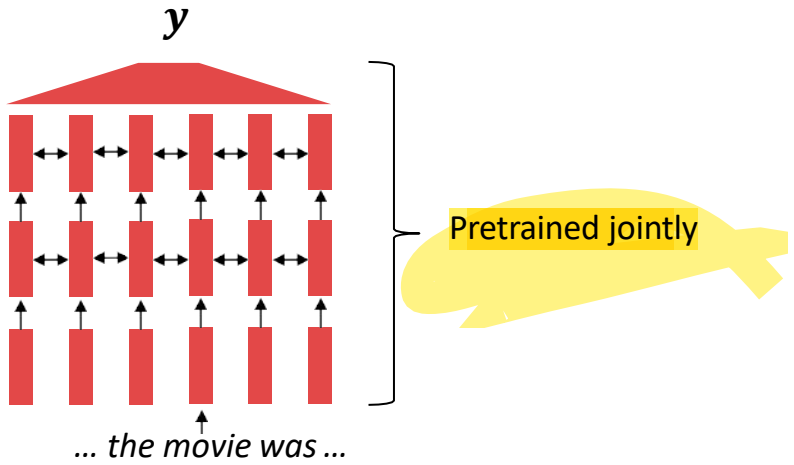
Transformers: The way you pre-train the encoder and the decoder → different architectures

(BERT, SpanBert, GPT, GPT-2, GPT-3, etc)

Pretraining models

In modern NLP:

- All (or almost all) parameters in NLP networks are initialized via **pretraining**.
- Pretraining methods hide parts of the input from the model, and train the model to reconstruct those parts.
- This has been exceptionally effective at building strong:
  - **representations of language**
  - **parameter initializations** for strong NLP models.

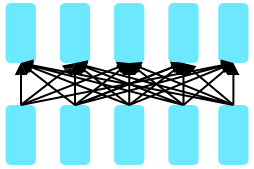


[This model has learned how to represent entire sentences through pretraining]

Adapted from John Hewitt

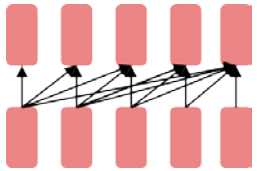
Pretraining for three types of architectures

The neural architecture influences the type of pretraining, and natural use cases.



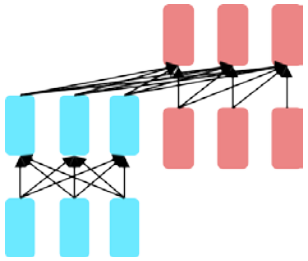
Encoders

- Gets bidirectional context – can condition on future!
- Wait, how do we pretrain them?



Decoders

- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words



Encoder-Decoders

- Good parts of decoders and encoders?
- What's the best way to pretrain them?

Adapted from John Hewitt

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Capturing meaning via context: What kinds of things does pretraining learn?

There's increasing evidence that pretrained models learn a wide variety of things about the statistical properties of language:

- *Stanford University is located in_____, California.* [Trivia]
- *I put____fork down on the table.* [syntax]
- *The woman walked across the street, checking for traffic over____shoulder.* [coreference]
- *I went to the ocean to see the fish, turtles, seals, and_____.* [lexical semantics/topic]
- *Overall, the value I got from the two hours watching it was the sum total of the popcorn and the drink. The movie was____.* [sentiment]
- Iroh went into the kitchen to make some tea. Standing next to Iroh, Zuko pondered his destiny. Zuko left the_____. [some reasoning – this is harder]
- I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21,_____[some basic arithmetic; they don't learn the Fibonnaci sequence]
- Models also learn – and can exacerbate racism, sexism, all manner of bad biases.

Adapted from John Hewitt

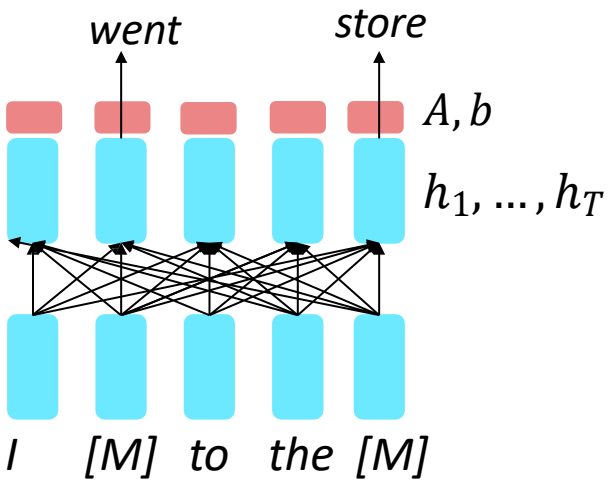
Pretraining encoders: What pretraining objective to use?

So far, we've looked at language model pretraining. But **encoders get bidirectional context, so we can't do language modeling!**

Idea: replace some fraction of words in the input with a special [MASK] token; predict these words.

$$h_1, \dots, h_T = \text{Encoder}(w_1, \dots, w_T)$$
$$y_i \sim Aw_i + b$$

Only add loss terms from words that are "masked out." If x' is the masked version of x, we're learning $p_\theta(x|x')$. Called **Masked LM**.



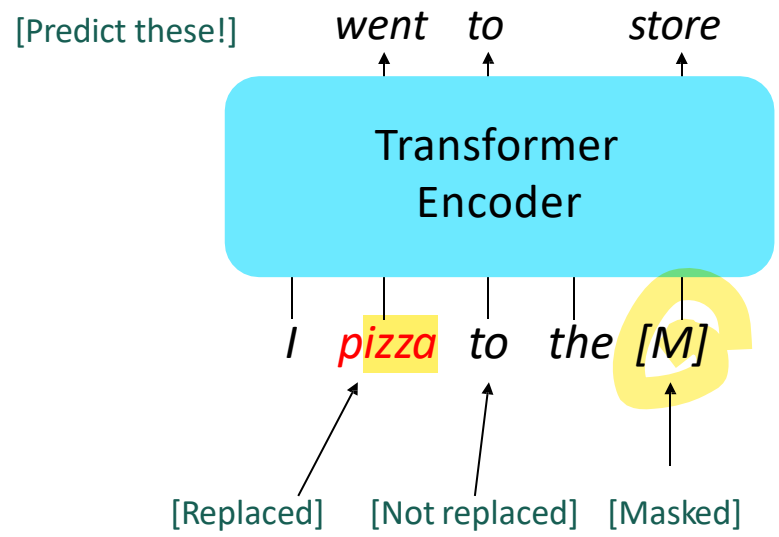
[Devlin et al., 2018]

BERT: Bidirectional Encoder Representations from Tranformers

Devlin et al., 2018 proposed the "Masked LM" objective and **released the weights of a pretrained Transformer**, a model they labeled BERT.

Some more details about Masked LM for BERT:

- Predict a random 15% of (sub)word tokens.
  - Replace input word with [MASK] 80% of the time
  - Replace input word with a random token 10% of the time
  - Leave input word unchanged 10% of the time (but still predict it!)
- Why? Doesn't let the model get complacent and not build strong representations of non-masked words. (No masks are seen at fine-tuning time!)



[Devlin et al., 2018]

BERT: Bidirectional Encoder Representations from Tranformers

- Mask out *k*% of the input words, and then predict the masked words
  - They always use *k* = 15%

store gallon

↑ ↑

the man went to the [MASK] to buy a [MASK] of milk

- Too little masking: Too expensive to train
- Too much masking: Not enough context

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

BERT: Bidirectional Encoder Representations from Tranformers

- Additional task: Next sentence prediction
- To learn *relationships* between sentences, predict whether Sentence B is actual sentence that proceeds Sentence A, or a random sentence

Sentence A = `The man went to the store.`
Sentence B = `He bought a gallon of milk.`
Label = `IsNextSentence`

Sentence A = `The man went to the store.`
Sentence B = `Penguins are flightless.`
Label = `NotNextSentence`

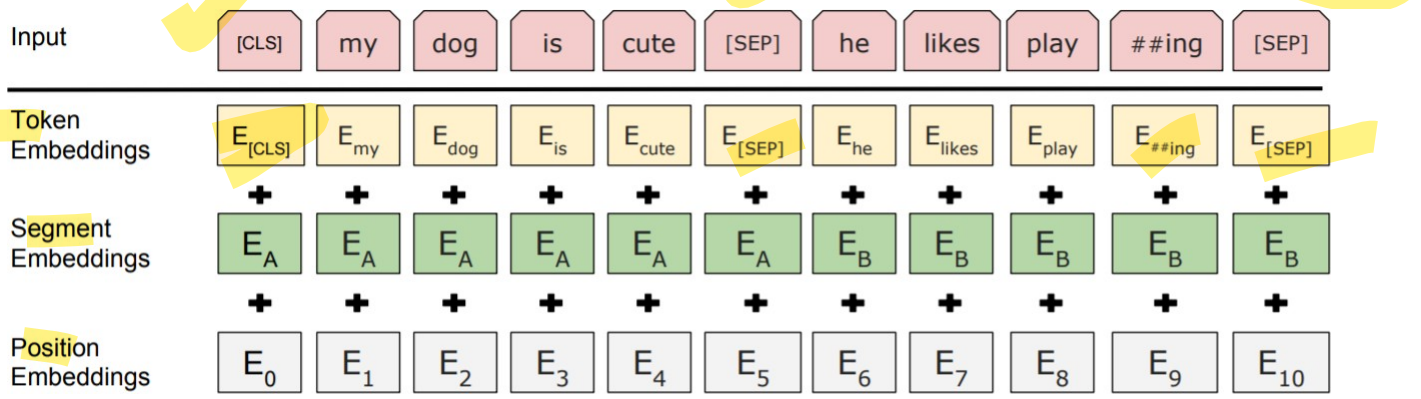
Adapted from Christopher Manning

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

BERT: Bidirectional Encoder Representations from Tranformers

- The pretraining input to BERT was two separate contiguous chunks of text:



- In addition to masked input reconstruction, BERT was trained to predict whether one chunk follows the other or is randomly sampled.
- Later work has argued this "next sentence prediction" is not necessary.

[Devlin et al., 2018, Liu et al., 2019]

Adapted from John Hewitt

BERT: Bidirectional Encoder Representations from Tranformers

Details about BERT

- Two models were released:
  - BERT-base: 12 layers, 768-dim hidden states, 12 attention heads, 110 million params.
  - BERT-large: 24 layers, 1024-dim hidden states, 16 attention heads, 340 million params.
- Trained on:
  - BooksCorpus (800 million words)
  - English Wikipedia (2,500 million words)
- Pretraining is expensive and impractical on a single GPU.
  - BERT was pretrained with 64 TPU chips for a total of 4 days.
  - (TPUs are special tensor operation acceleration hardware)
- Finetuning is practical and common on a single GPU
  - "Pretrain once, finetune many times."

[Devlin et al., 2018]

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

BERT: Bidirectional Encoder Representations from Tranformers

BERT was massively popular and hugely versatile; finetuning BERT led to new state-of- the-art results on a broad range of tasks.

- **QQP:** Quora Question Pairs (detect paraphrase questions)
- **QNLI**: natural language inference over question answering data
- **SST-2**: sentiment analysis
- **CoLA**: corpus of linguistic acceptability (detect whether sentences are grammatical.)
- **STS-B**: semantic textual similarity
- **MRPC**: microsoft paraphrase corpus
- **RTE**: small natural language inference corpus

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

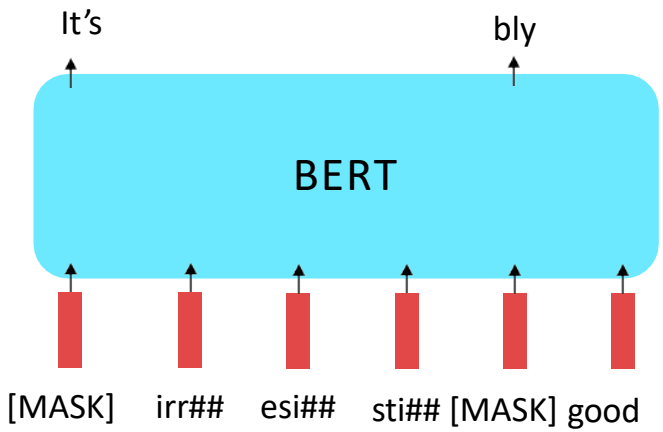
[Devlin et al., 2018]

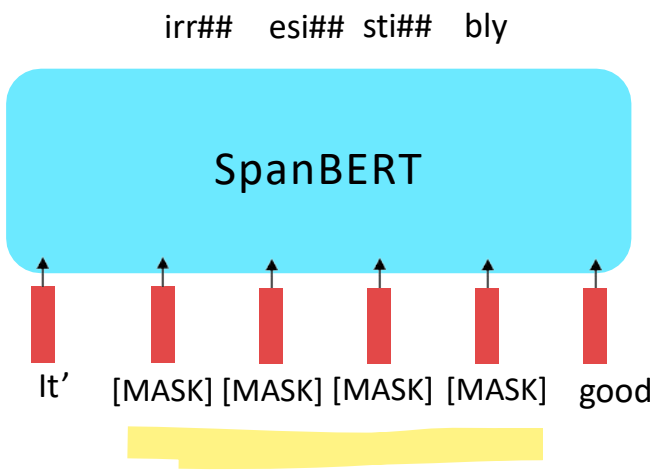
Extensions of BERT

You'll see a lot of BERT variants like RoBERTa, SpanBERT, +++

Some generally accepted improvements to the BERT pretraining formula:

- RoBERTa: mainly just train BERT for longer and remove next sentence prediction!
- SpanBERT: masking contiguous spans of words makes a harder, more useful pretraining task





[Liu et al., 2019; Joshi et al., 2020]

Extensions of BERT

A takeaway from the RoBERTa paper: more compute, more data can improve pretraining even when not changing the underlying Transformer encoder.

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**
$\text{BERT}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7

[Liu et al., 2019; Joshi et al., 2020]

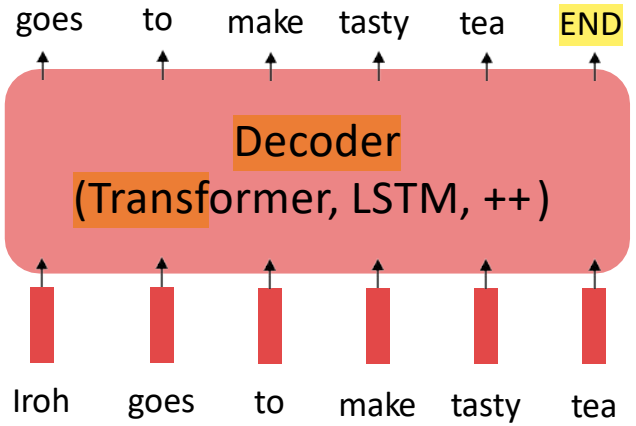
Pretraining through language modeling [Dai and Le, 2015]

Recall the **language modeling** task:

- Model $p_\theta(w_t|w_{1:t-1})$, the probability distribution over words given their past contexts.
- There's lots of data for this! (In English.)

Pretraining through language modeling:

- Train a neural network to perform language modeling on a large amount of text.
- Save the network parameters.

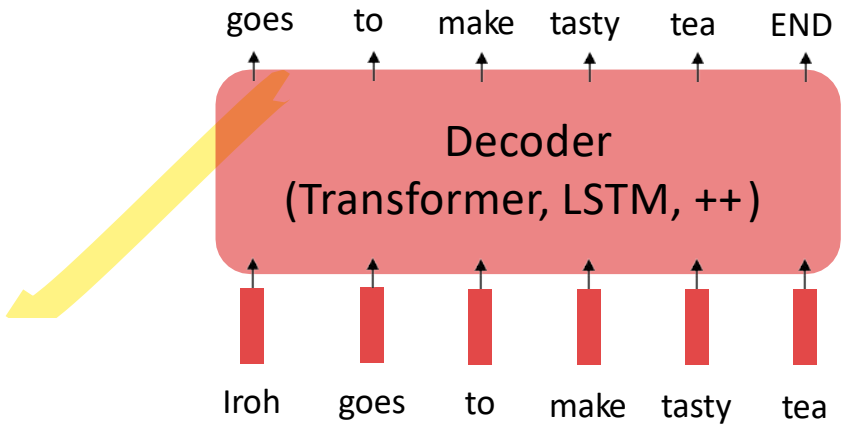


The Pretraining / Finetuning Paradigm

Pretraining can improve NLP applications by serving as parameter initialization.

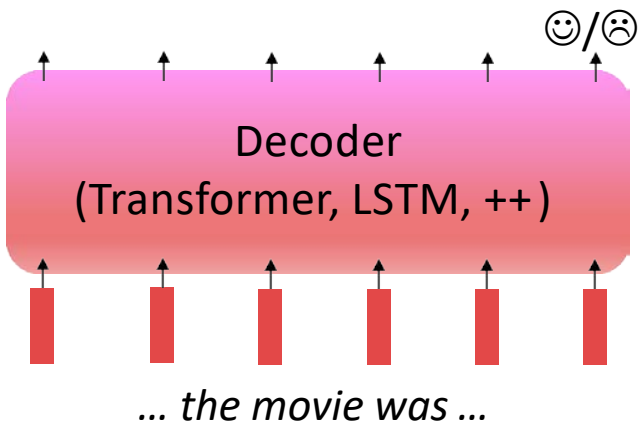
Step 1: Pretrain (on language modeling)

Lots of text; learn general things!



Step 2: Finetune (on your task)

Not many labels; adapt to the task!



Pretraining decoders

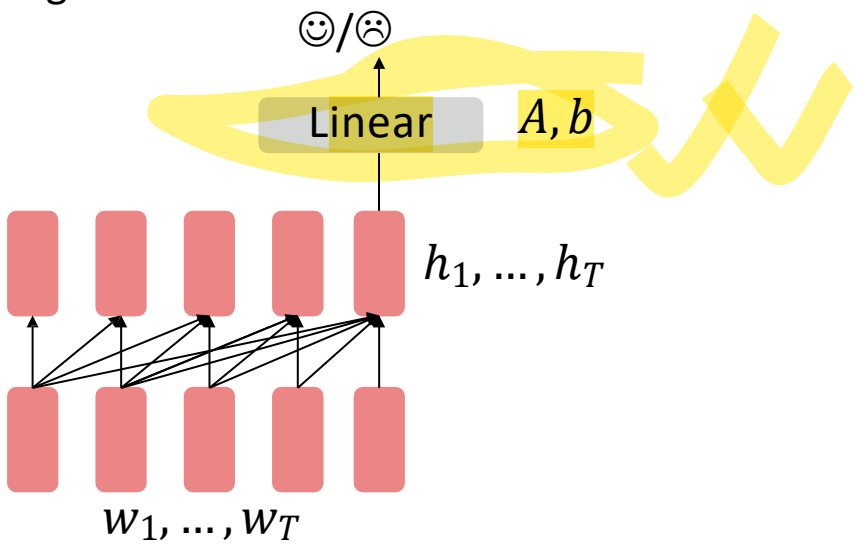
When using language model pretrained decoders, we can ignore that they were trained to model $p(w_t|w_{1:t-1})$.

We can finetune them by training a classifier on the last word's hidden state.

$$h_1, \ldots, h_T = \text{Decoder}(w_1, \ldots, w_T)$$
$$y \sim Aw_T + b$$

Where A and b are randomly initialized and specified by the downstream task.

Gradients backpropagate through the whole network.



[Note how the linear layer hasn't been pretrained and must be learned from scratch.]

Pretraining decoders

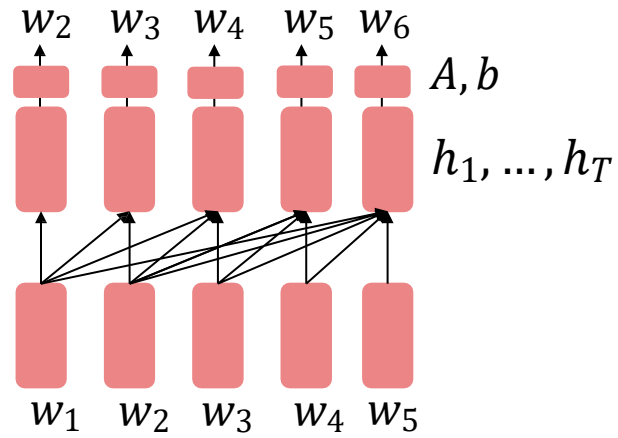
It's natural to pretrain decoders as language models and then use them as generators, finetuning their $p_\theta(w_t|w_{1:t-1})$!

This is helpful in tasks **where the output is a sequence** with a vocabulary like that at pretraining time!

- Dialogue (context=dialogue history)
- Summarization (context=document)

$$h_1, \dots, h_T = \text{Decoder}(w_1, \dots, w_T)$$
$$w_t \sim Aw_{t-1} + b$$

Where A, b were pretrained in the language model!



[Note how the linear layer has been pretrained.]

Generative Pretrained Transformer (GPT) [Radford et al., 2018]

2018's GPT was a big success in pretraining a decoder!

- Transformer decoder with 12 layers.
- 768-dimensional hidden states, 3072-dimensional feed-forward hidden layers.
- Byte-pair encoding with 40,000 merges
- Trained on BooksCorpus: over 7000 unique books.
  - Contains long spans of contiguous text, for learning long-distance dependencies.
- The acronym "GPT" never showed up in the original paper; it could stand for "Generative PreTraining" or "Generative Pretrained Transformer"

[Devlin et al., 2018]

Generative Pretrained Transformer (GPT)

[Radford et al., 2018]

How do we format inputs to our decoder for **finetuning tasks?**

Natural Language Inference: Label pairs of sentences as *entailing/contradictory/neutral*

Premise: *The man is in the doorway*
Hypothesis: *The person is near the door* } **entailment**

Radford et al., 2018 evaluate on natural language inference.
Here's roughly how the input was formatted, as a sequence of tokens for the decoder.

[START] *The man is in the doorway* [DELIM] *The person is near the door* [EXTRACT]

The linear classifier is applied to the representation of the [EXTRACT] token.

Generative Pretrained Transformer (GPT) [Radford et al., 2018]

GPT results on various *natural language inference* datasets.

Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	89.3	-	-	-
CAFE [58] (5x)	80.2	79.0	89.3	-	-	-
Stochastic Answer Network [35] (3x)	80.6	80.1	-	-	-	-
CAFE [58]	78.7	77.9	88.5	83.3		
GenSen [64]	71.4	71.3	-	-	82.3	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	**61.7**
Finetuned Transformer LM (ours)	**82.1**	**81.4**	**89.9**	**88.3**	**88.1**	56.0

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Increasingly convincing generations (GPT2) [Radford et al., 2018]

We mentioned how pretrained decoders can be used **in their capacities as language models.** **GPT-2,** a larger version of GPT trained on more data, was shown to produce relatively convincing samples of natural language.

Context (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.
GPT-2: 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.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Pretraining encoder-decoders: What pretraining objective to use?

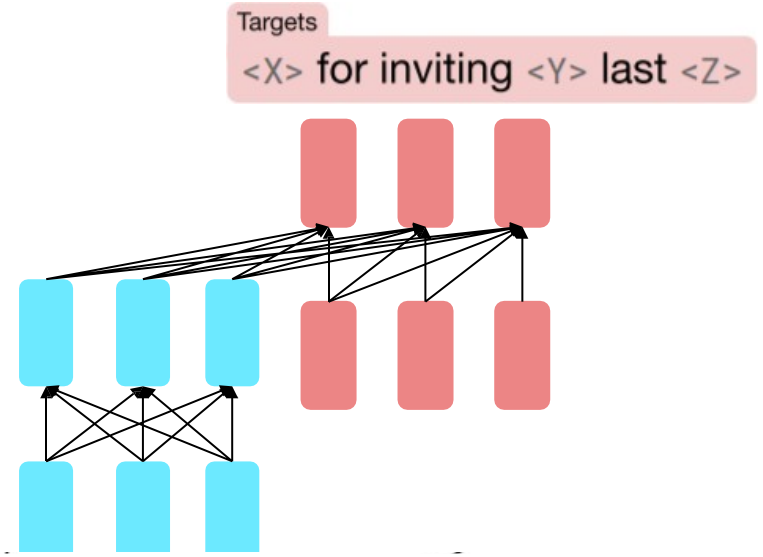
What Raffel et al., 2018 found to work best was **span corruption.** Their model: **T5**.

Replace different-length spans from the input with unique placeholders; decode out the spans that were removed!

Original text

Thank you for inviting me to your party last week.

This is implemented in text preprocessing: it's still an objective that looks like **language modeling** at the decoder side.



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Pretraining encoder-decoders: What pretraining objective to use? T5

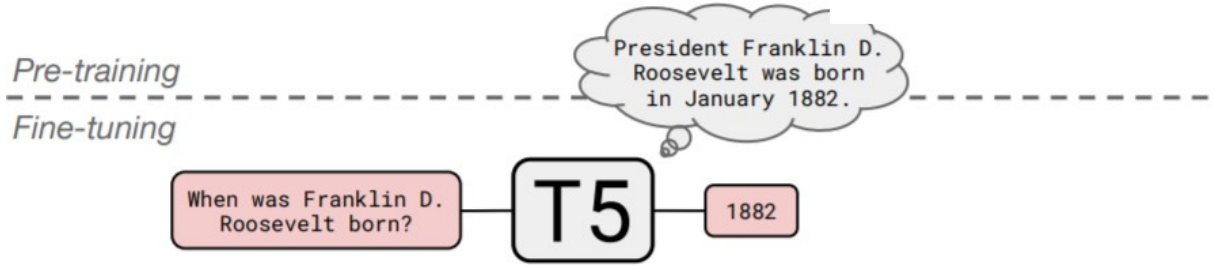
A fascinating property of T5: it can be finetuned to answer a wide range of questions, retrieving knowledge from its parameters.

NQ: Natural Questions

WQ: WebQuestions

TQA: Trivia QA

All "open-domain" versions



	NQ	WQ	TQA dev	TQA test	
Karpukhin et al. (2020)	**41.5**	42.4	**57.9**	–	
T5.1.1-Base	25.7	28.2	24.2	30.6	**220 million params**
T5.1.1-Large	27.3	29.5	28.5	37.2	**770 million params**
T5.1.1-XL	29.5	32.4	36.0	45.1	**3 billion params**
T5.1.1-XXL	32.8	35.6	42.9	52.5	**11 billion params**
T5.1.1-XXL + SSM	35.2	**42.8**	51.9	**61.6**	

[Raffel et al., 2018]

GPT-3, in-context learning, very large models

So far, we've interacted with pretrained models in two ways:

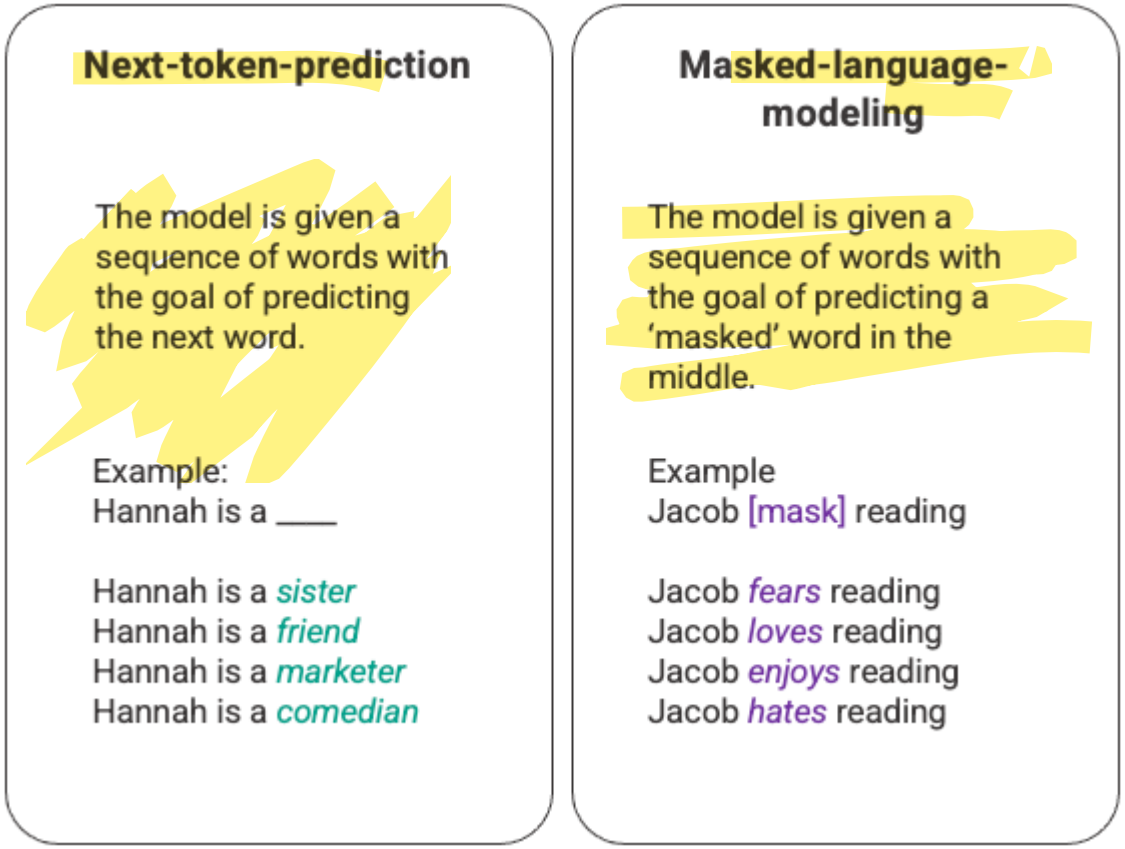
- Sample from the distributions they define (maybe providing a prompt)
- Fine-tune them on a task we care about, and take their predictions.

Very large language models seem to perform some kind of learning **without gradient steps** simply from examples you provide within their contexts.

GPT-3 is the canonical example of this. The largest **T5** model had 11 billion parameters.

GPT-3 has 175 billion parameters.

Large Language Models



Arbitrary example of next-token-prediction and masked-language-modeling generated by the author.

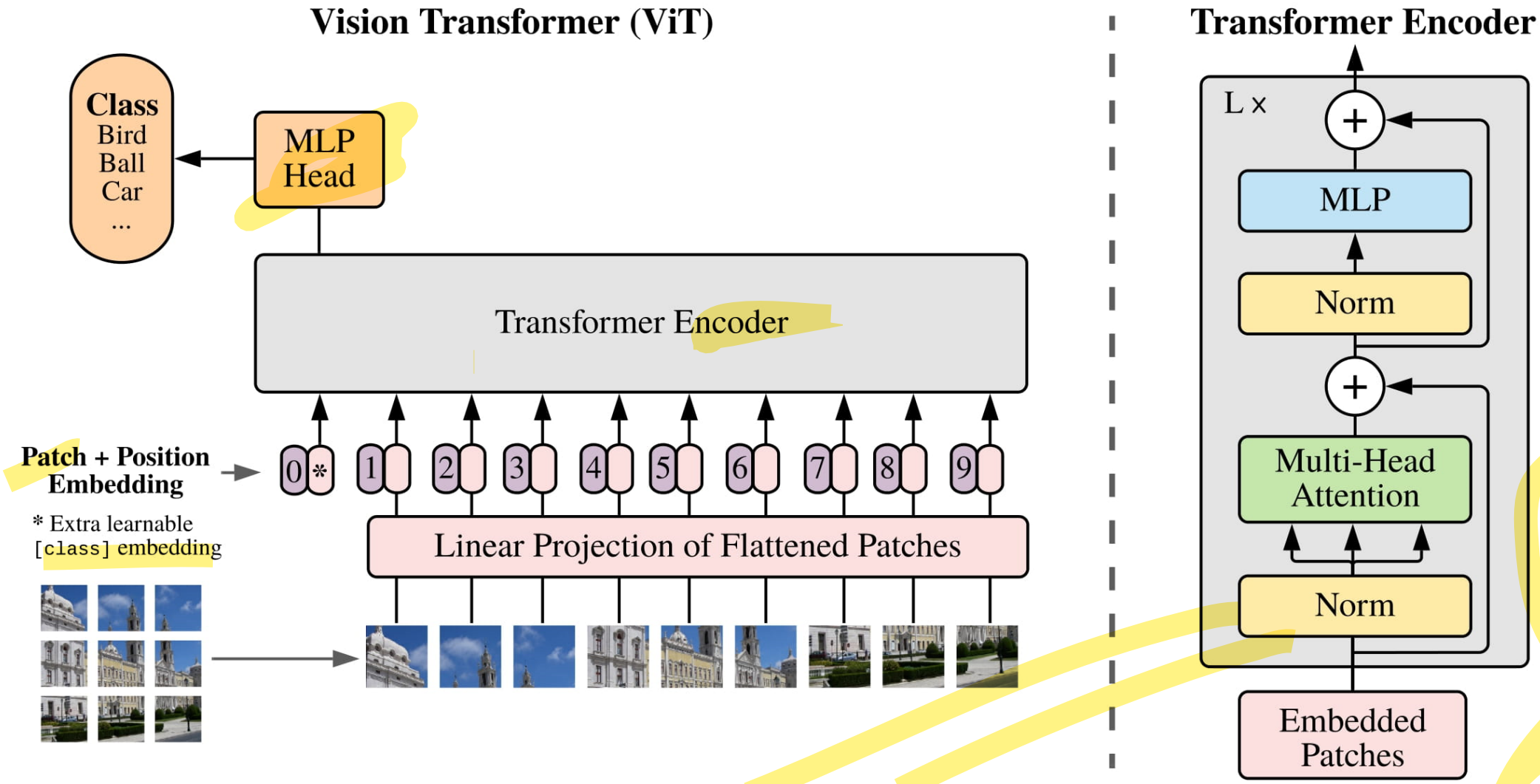
https://towardsdatascience.com/how-chatgpt-works-the-models-behind-the-bot-1ce5fca96286

UNIVERSITÉ
CÔTE D'AZUR

Transformers beyond Natural Language Processing

univ-cotedazur.fr

Visual Transformers (ViT)



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Cross-modal transformers

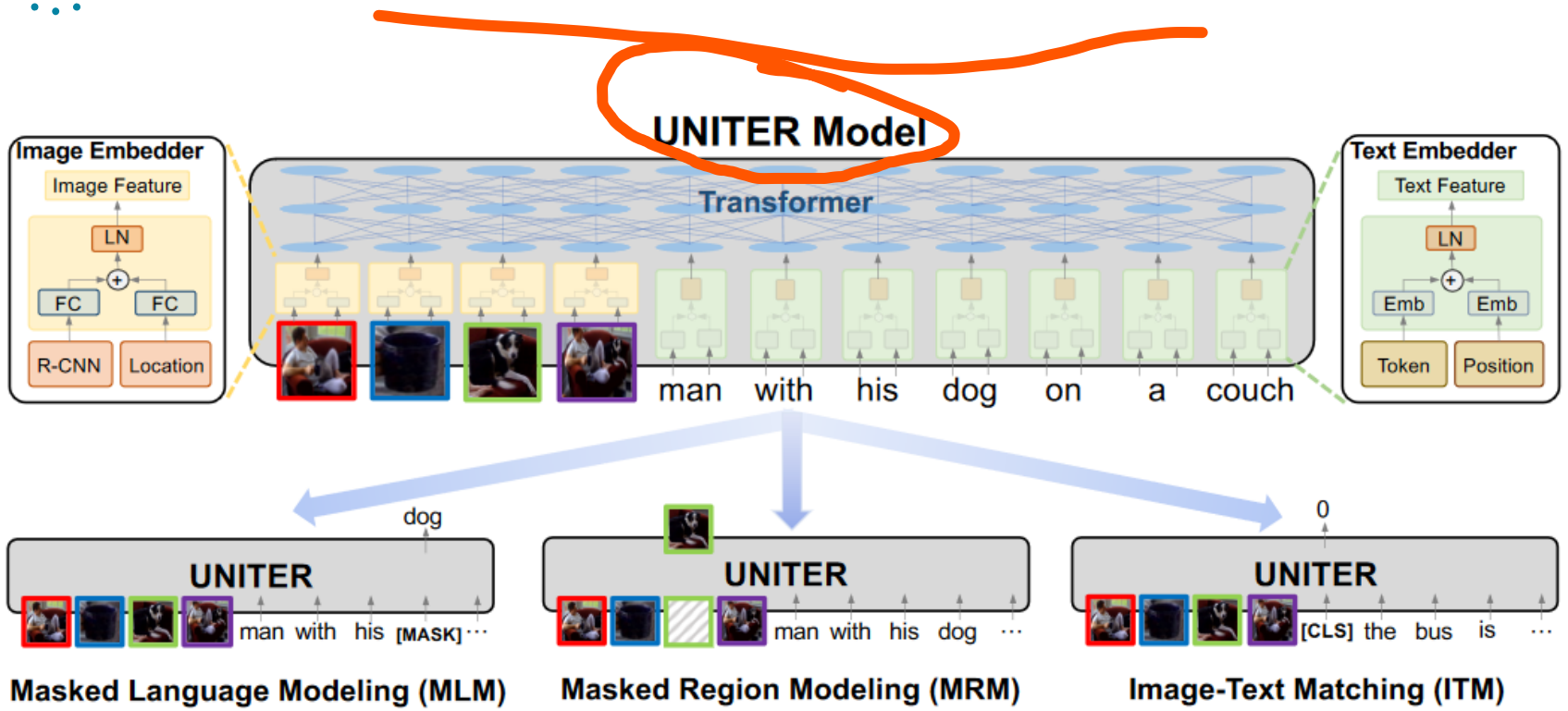


Figure 1: Overview of the proposed UNITER model (best viewed in color), consisting of an Image Embedder, a Text Embedder and a multi-layer self-attention Transformer, learned through three pre-training tasks.

Chen et al., "UNITER: Learning UNiversal Image-TExt Representations", arxiv 2019

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Cross-modal transformers

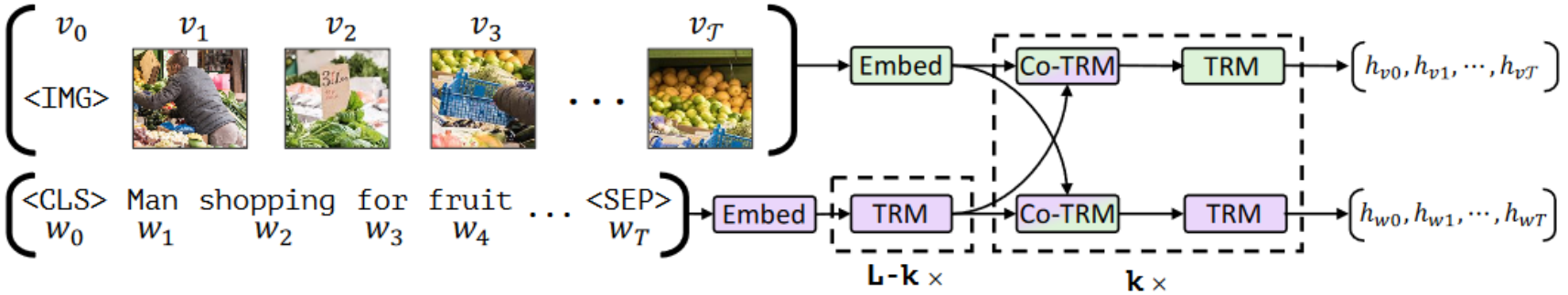


Figure 1: Our ViLBERT model consists of two parallel streams for visual (green) and linguistic (purple) processing that interact through novel co-attentional transformer layers. This structure allows for variable depths for each modality and enables sparse interaction through co-attention. Dashed boxes with multiplier subscripts denote repeated blocks of layers.

Lu et al., "ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks", NeurIPS 2019

Cross-modal transformers

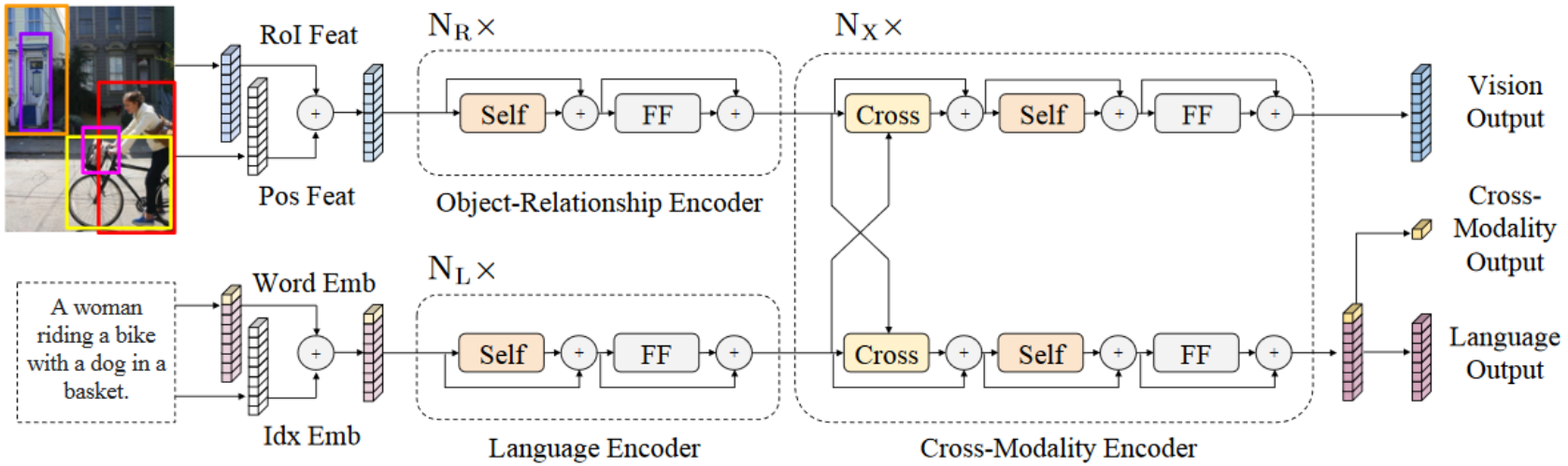


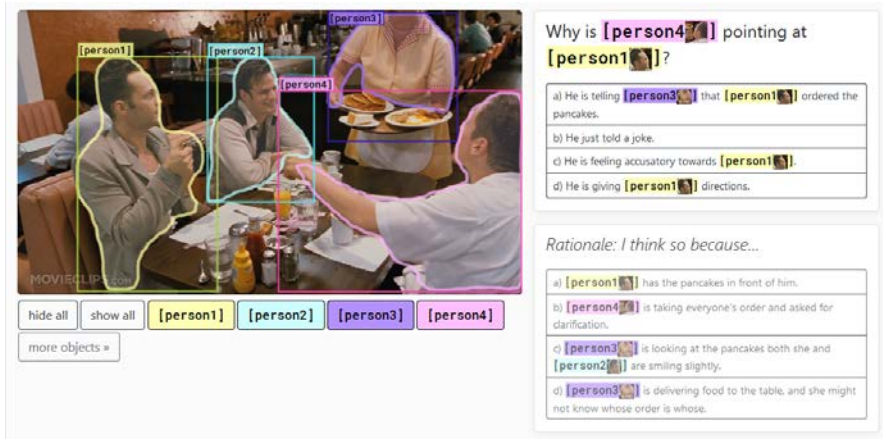
Figure 1: The LXMERT model for learning vision-and-language cross-modality representations. 'Self' and 'Cross' are abbreviations for self-attention sub-layers and cross-attention sub-layers, respectively. 'FF' denotes a feed-forward sub-layer.

Tan and Bansal, "LXMERT: Learning Cross-Modality Encoder Representationsfrom Transformers", EMNLP 2019

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Visual Commonsense Reasoning leaderboard



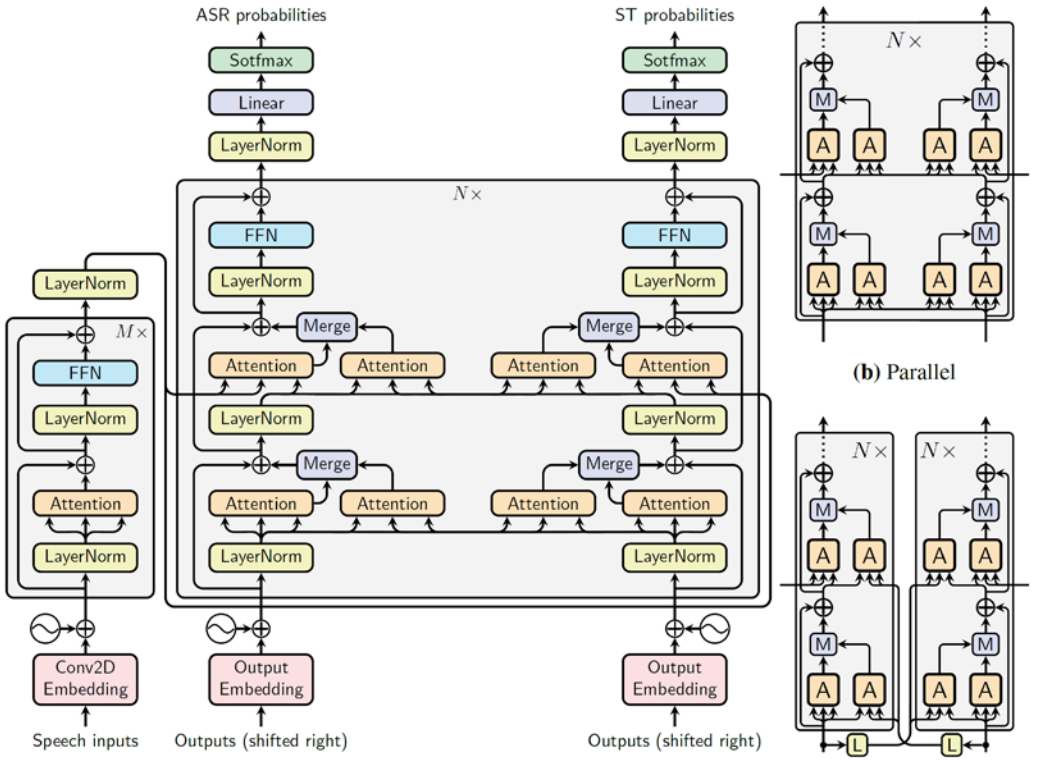
Rank	Model	Q->A	QA->R	Q->AR
	Human Performance *University of Washington* (Zellers et al. '18)	91.0	93.0	85.0
September 30, 2019	UNITER-large (ensemble) *MS D365 AI* https://arxiv.org/abs/1909.11740	**79.8**	**83.4**	**66.8**
2 September 23, 2019	UNITER-large (single model) *MS D365 AI* https://arxiv.org/abs/1909.11740	77.3	80.8	62.8
3 August 9,2019	ViLBERT (ensemble of 10 models) *Georgia Tech & Facebook AI Research* https://arxiv.org/abs/1908.02265	76.4	78.0	59.8
4 September 23,2019	VL-BERT (single model) *MSRA & USTC* https://arxiv.org/abs/1908.08530	75.8	78.4	59.7
5 August 9,2019	ViLBERT (ensemble of 5 models) *Georgia Tech & Facebook AI Research* https://arxiv.org/abs/1908.02265	75.7	77.5	58.8

https://visualcommonsense.com/leaderboard/

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Dual-decoder Transformer for Joint Automatic Speech Recognition (ASR) and Multilingual Speech Translation (ST)



(a) The parallel dual-decoder Transformer

(b) Parallel

(c) Cross

Figure 1: The dual-decoder Transformers. Figure (a) shows the detailed architecture of the *parallel* dual-decoder Transformer, and Figure (b) shows its simplified view. The *cross* dual-decoder Transformer is very similar to the parallel one, except that the keys and values fed to the dual-attention layers come from the previous output, which is illustrated by Figure (c). From the above figures, one can easily infer the detailed architecture of the cross Transformer, which can be found in the Appendix. Abbreviations: **A** (Attention), **M** (Merge), **L** (LayerNorm).

https://aclanthology.org/2020.coling-main.314.pdf

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Gene Transformer: Transformers for the Gene Expression-based Classification of Lung Cancer Subtypes

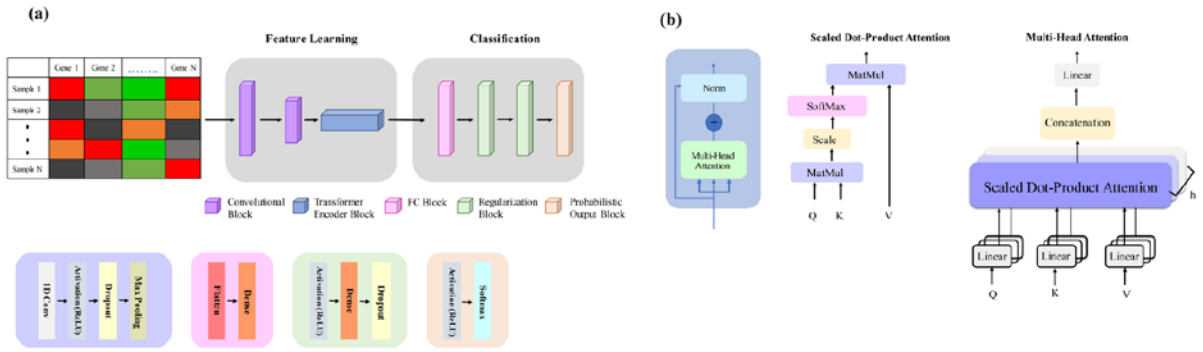


Fig. 2. Gene Transformer network: an illustration. **(a)** (left) Detailed description of Gene Transformer architecture. The model includes two parts: feature learning block and a fully connected block for classification. One-dimensional (1D) vectorized gene expression levels from The Cancer Genome Atlas (TCGA) samples were used as the input. This input was then convolved and fed into the multi-head self-attention module for identifying relevant biomarkers. **(b)** (Right) Scaled dot-product attention to compute the attention function on a set of queries and multi-head attention to concatenate the yielded output values and project them for downstream tasks. The exhibition of the Transformer encoder was inspired by [36].

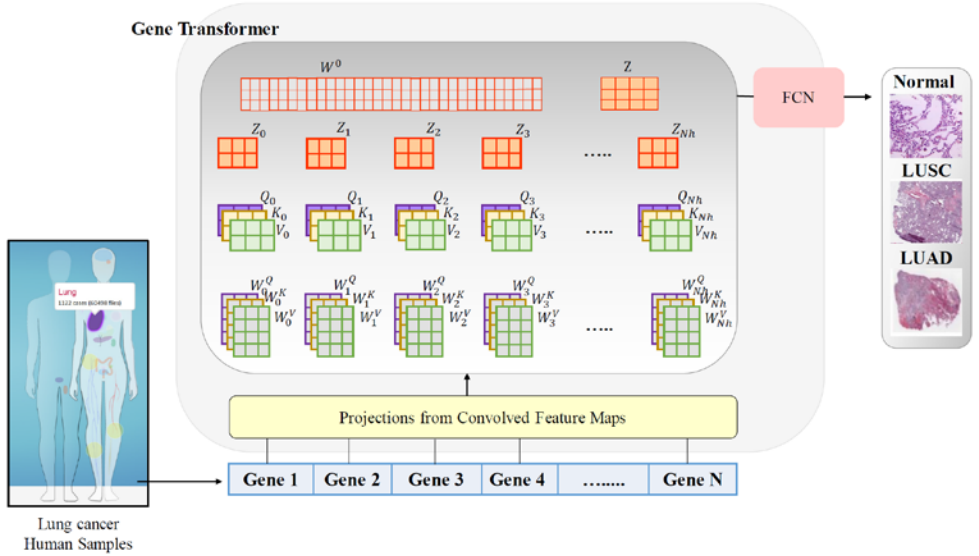


Fig. 1. Graphical representation of the end-to-end deep learning approach for the classification of the lung cancer subtypes.

In this study, we proposed an end-to-end approach for lung cancer subtype classification using a gene expression dataset, wherein a multi-head self-attention module permitted the model to jointly learn complex genomic information from thousands of genes from different patient samples shared across multiple cancer subtypes. Head collaboration achieves better generalizability over imbalanced datasets and produces a more desirable performance for both binary and multiclass classification tasks.

https://arxiv.org/ftp/arxiv/papers/2108/2108.11833.pdf

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Adaptable and Interpretable Neural Memory Over Symbolic Knowledge

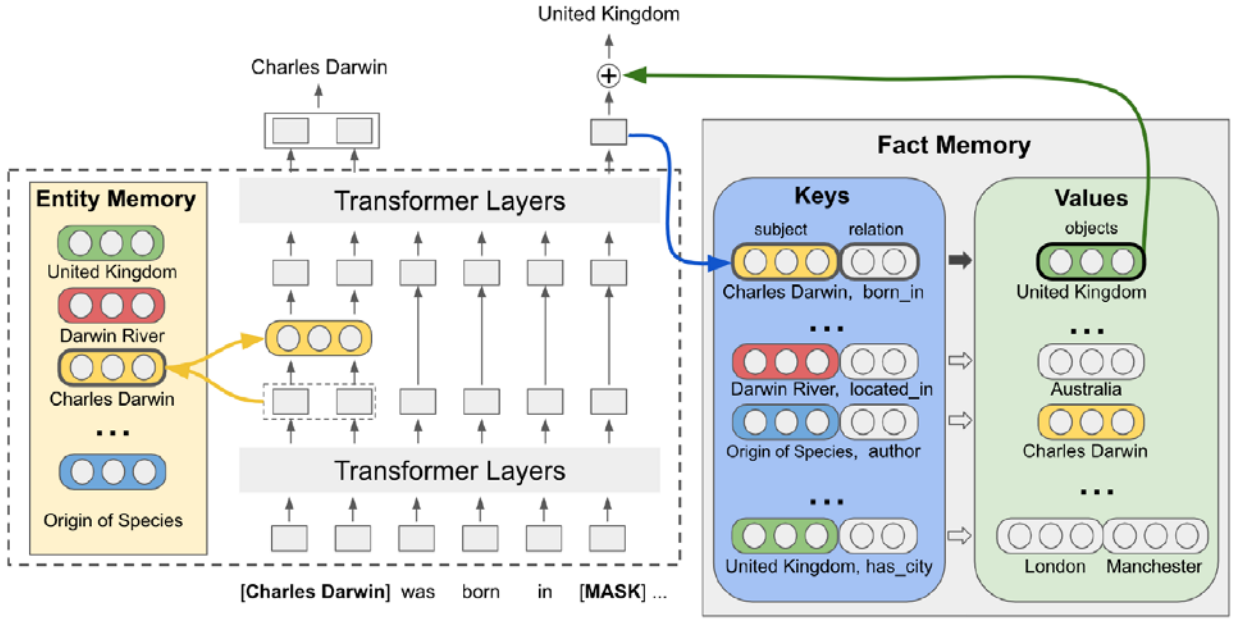
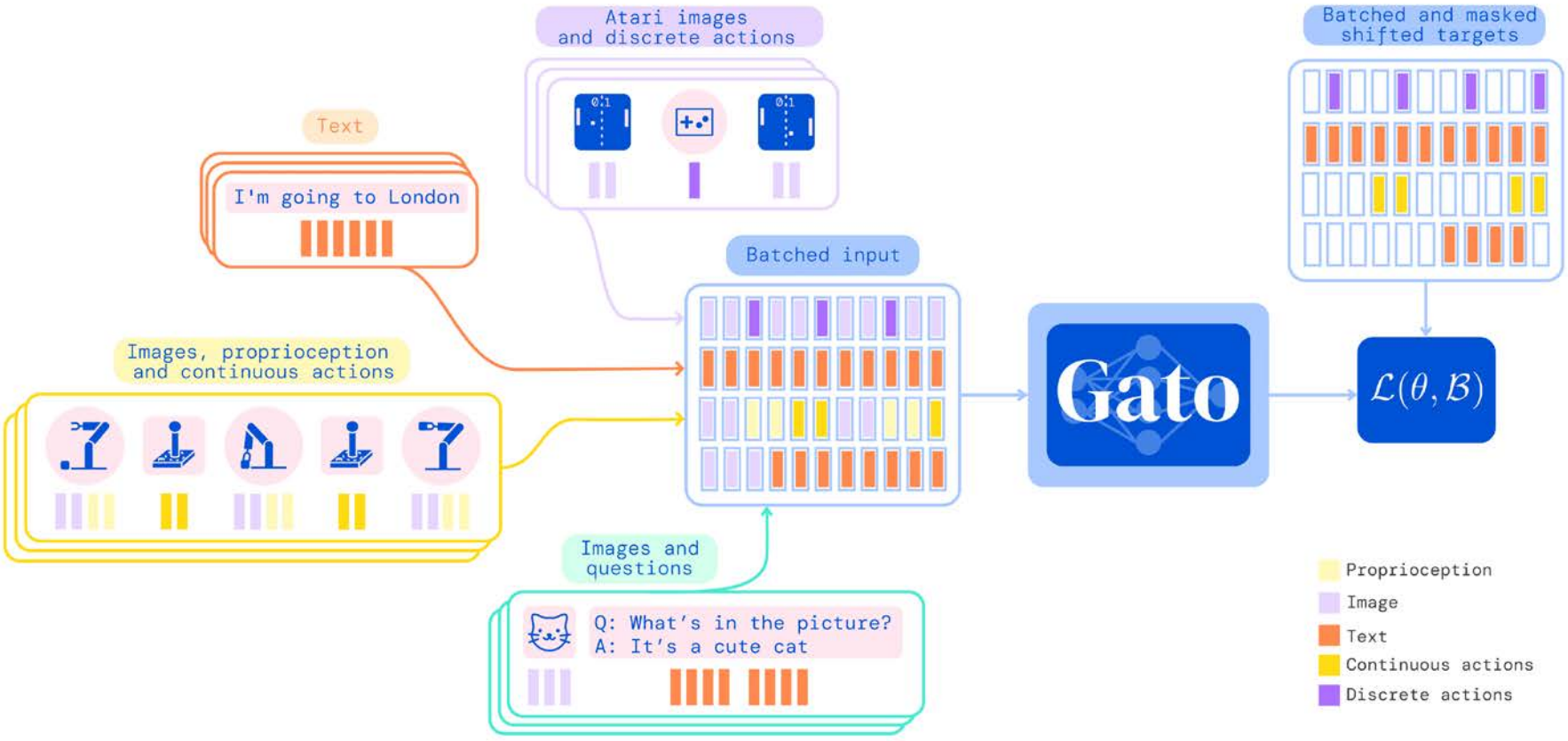


Figure 1: **Fact Injected Language Model architecture.** The model takes a piece of text (a question during fine-tuning or arbitrary text during pre-training) and first contextually encodes it with an entity enriched transformer. FILM uses the contextually encoded MASK token as a query to the fact memory. In this case, the contextual query chooses the fact key (*Charles Darwin, born_in*) which returns the set of values {*United Kingdom*} (The value set can be multiple entity objects such as the case from calling the key [*United Kingdom, has_city*]) . The returned object representation is incorporated back into the context in order to make the final prediction. Note that the entity representations in the facts (both in keys and values) are shared with the entity memory. The portion within the dashed line follows the procedure from Févry et al. (2020).

Pat Verga, Haitian Sun, Livio Baldini Soares, and William Cohen. 2021. Adaptable and Interpretable Neural MemoryOver Symbolic Knowledge.
In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL), pages 3678–3691, Online. Association for Computational Linguistics.

Generalist Agent (Deepmind): GATO



https://www.deepmind.com/publications/a-generalist-agent

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Generalist Agent (Deepmind): GATO

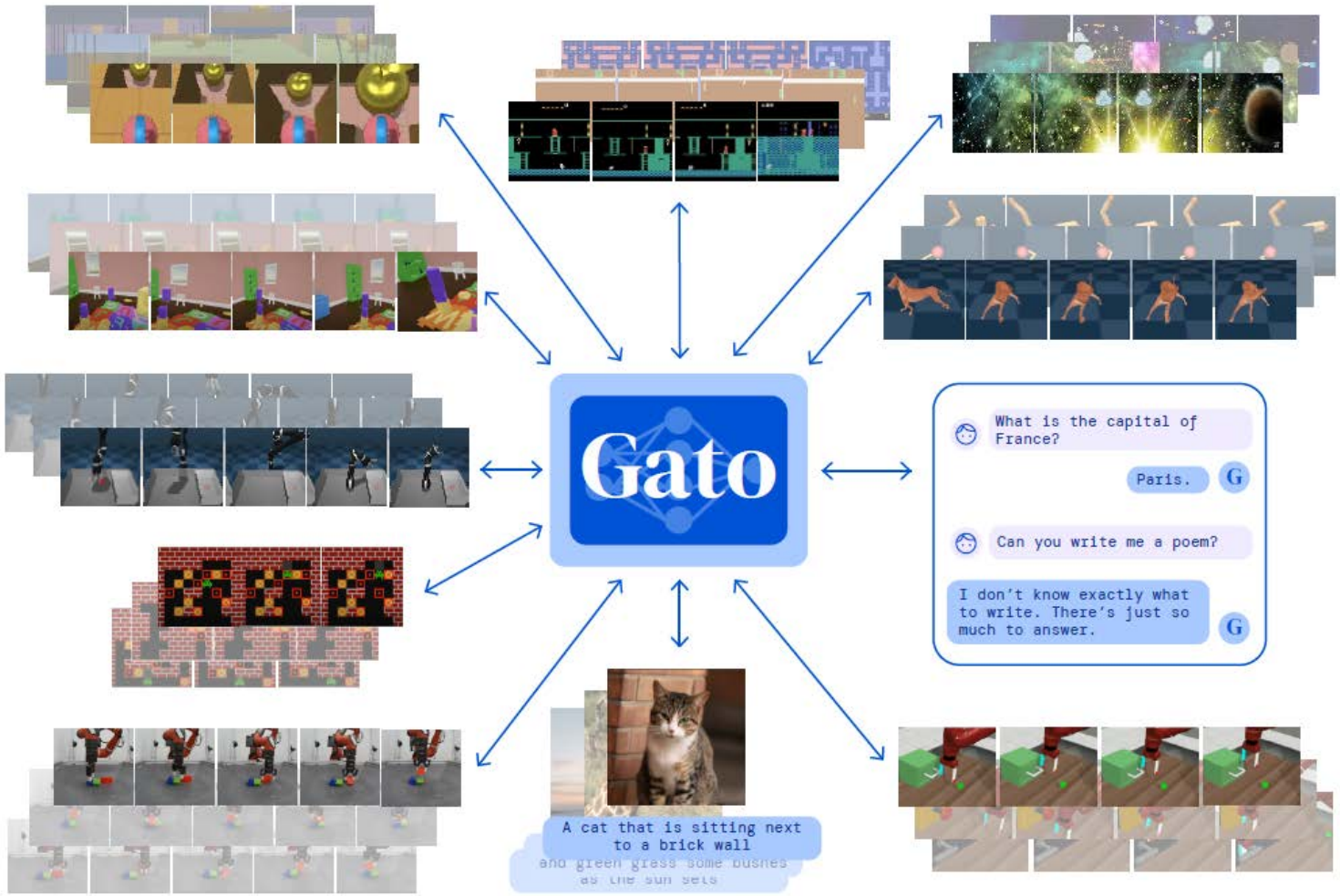


Figure 1 | **A generalist agent.** Gato can sense and act with different embodiments across a wide range of environments using a single neural network with the same set of weights. Gato was trained on 604 distinct tasks with varying modalities, observations and action specifications.

UNIVERSITÉ
CÔTE D'AZUR

Transformers are amazing but Be aware!

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Computational cost, Energy, and Sovereinty

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Be aware of what it (computationally) costs: Winter is coming!

- To train the increasing number of parameters (e.g. the "latest" GPT-3 from 2021 for NLP is 176 **billions** of parameters), you need a HUGE amount of data
  ⇒ this raises questions on data privacy, in particular on personal data…

- So you need also a HUGE amont of computational resources which require energy…

Consumption	CO_2e (lbs)
Air travel, 1 passenger, NY↔SF	1984
Human life, avg, 1 year	11,023
American life, avg, 1 year	36,156
Car, avg incl. fuel, 1 lifetime	126,000
Training one model (GPU)	
NLP pipeline (parsing, SRL)	39
w/ tuning & experimentation	78,468
Transformer (big)	192
w/ neural architecture search	626,155

Table 1: Estimated CO_2 emissions from training common NLP models, compared to familiar consumption.[1]

Strubell, E., Ganesh, A., & McCallum, A. (2019, July). ***Energy and Policy Considerations for Deep Learning in NLP.*** *In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (pp. 3645-3650).*

ChatGPT: For what we know

- Architecture: Transformers
- Parameters and increasing data:

GPT-2	GPT-3
1.5 billion parameters	176 billion parameters
40 GB text training dataset	570 GB training dataset comprising of books, articles, websites, and more
Often fine-tuned to perform specific tasks	Ability to perform most language tasks without additional tuning
Smaller version of the model was released to the public open source	Launched as an API service

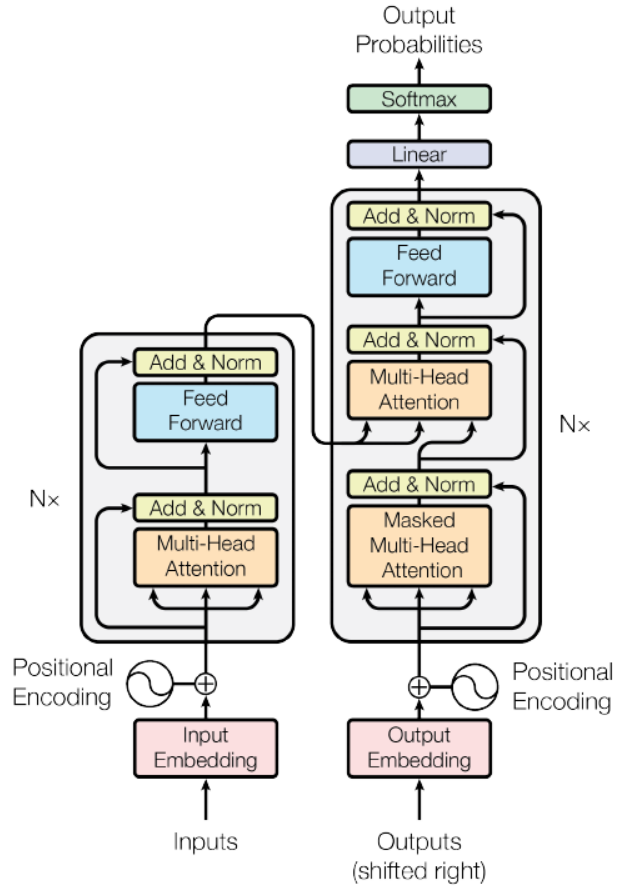
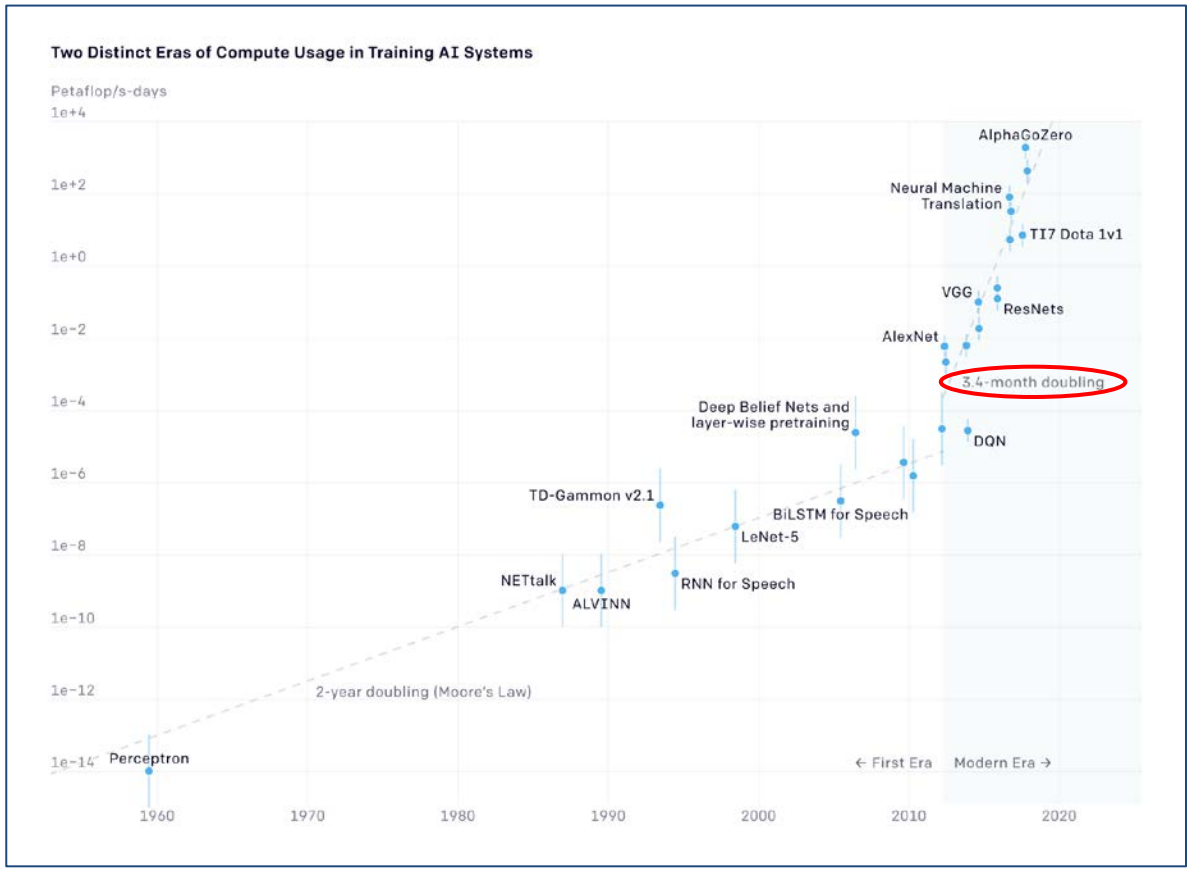


Figure 1: The Transformer - model architecture.

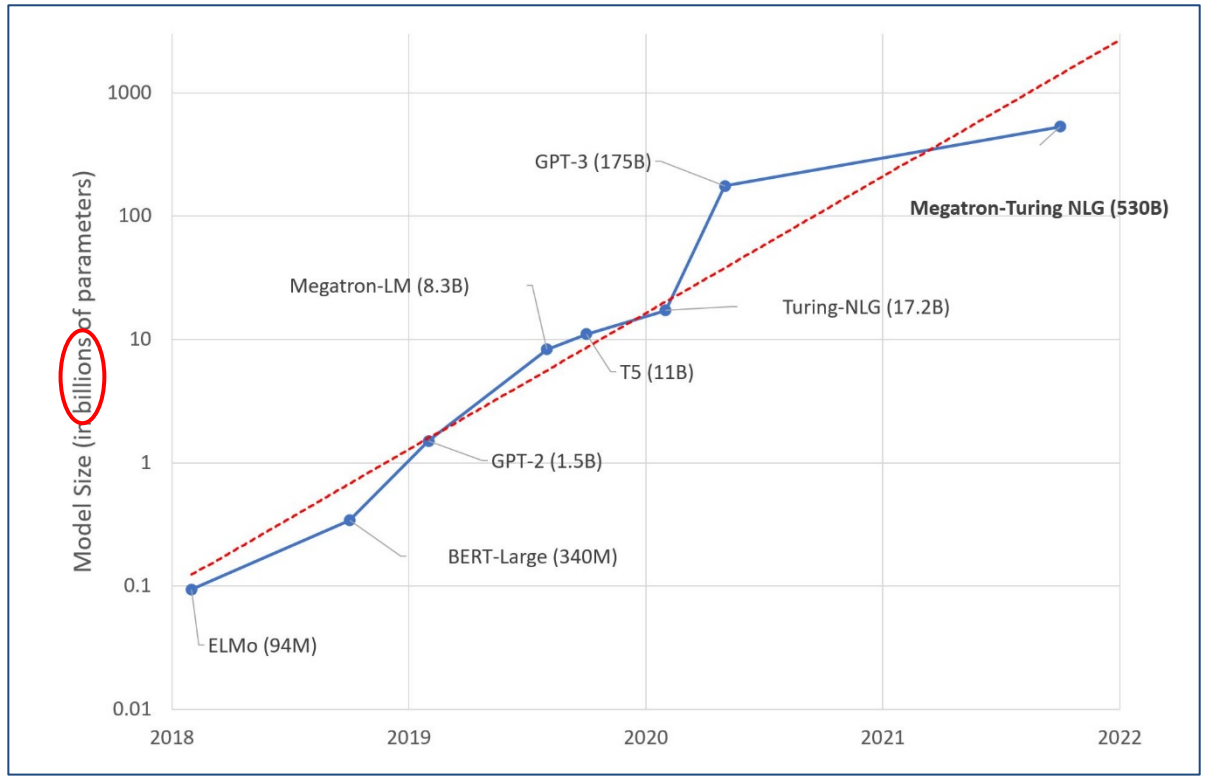
[1] A. Vaswani et al., "Attention Is All You Need.", Neural Information Processing System, 2017.
https://towardsdatascience.com/how-chatgpt-works-the-models-behind-the-bot-1ce5fca96286

Questions on Sovereinty…

- Not everyone can train or even **use** these models. The ones that can are the GAFAM-BATX…



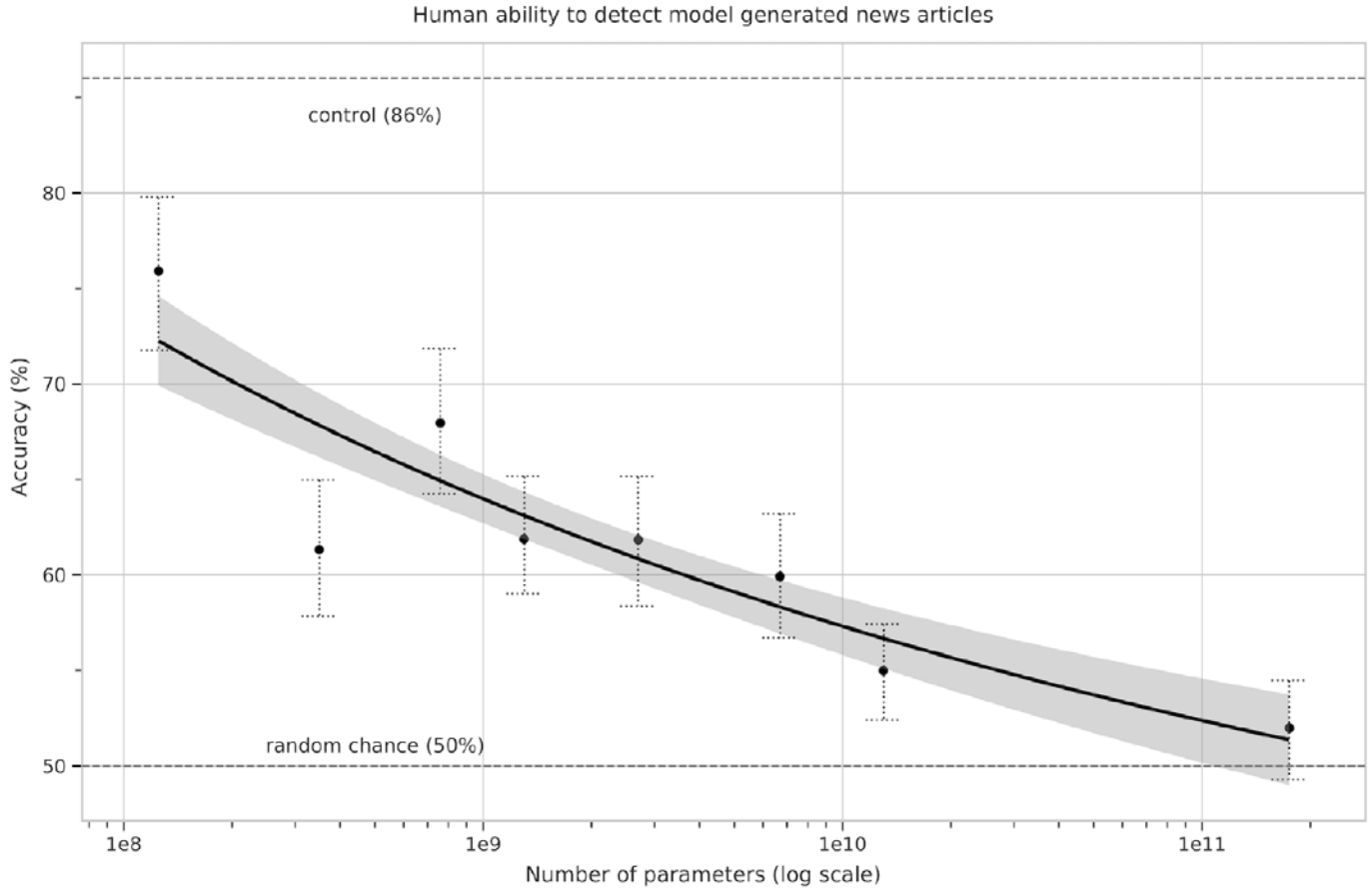
(Source : https://openai.com/blog/ai-and-compute/)



(Source: https://www.microsoft.com/en-us/research/blog/using-deepspeed-and-megatron-to-train-megatron-turing-nlg-530b-the-worlds-largest-and-most-powerful-generative-language-model/)

Questions on Sovereinty…

- Not everyone can train or even **use** these models. The ones that can are the GAFAM-BATX…



Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. NeurIPS 2020, 33, 1877-1901.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Biases

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Amazing, but…be careful of a little bias at the input

Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings

Tolga Bolukbasi[1], Kai-Wei Chang[2], James Zou[2], Venkatesh Saligrama[1,2], Adam Kalai[2]

[1]Boston University, 8 Saint Mary's Street, Boston, MA

[2]Microsoft Research New England, 1 Memorial Drive, Cambridge, MA

tolgab@bu.edu, kw@kwchang.net, jamesyzou@gmail.com, srv@bu.edu, adam.kalai@microsoft.com

From Nello Cristianini, at at *Frontier Research and Artificial Intelligence Conference*: https://erc.europa.eu/sites/default/files/events/docs/Nello_Cristianini-ThinkBIG-Patterns-in-Big-Data.pdf

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Beware the biases!

- Google Translate

- Deepl

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Beware the biases!

BUSINESS NEWS OCTOBER 10, 2018 / 5:12 AM / 7 MONTHS AGO

Amazon scraps secret AI recruiting tool that showed bias against women

Jeffrey Dastin 8 MIN READ

SAN FRANCISCO (Reuters) - Amazon.com Inc's (AMZN.O) machine-learning specialists uncovered a big problem: their new recruiting engine did not like women.

Forget Killer Robots—Bias Is the Real AI Danger

John Giannandrea.
GETTY

Artificial Intelligence / Robots

Forget Killer Robots—Bias Is the Real AI Danger

John Giannandrea, who leads AI at Google, is worried about intelligent systems learning human prejudices.

by **Will Knight** Oct 3, 2017

Google's AI chief isn't fretting about super-intelligent killer robots. Instead, John Giannandrea is concerned about the danger that may be lurking inside the machine-learning algorithms used to make millions of decisions every minute.

"The real safety question, if you want to call it that, is that if we give these systems biased data, they will be biased," Giannandrea said before a recent Google conference on the relationship between humans and AI systems.

The problem of bias in machine learning is likely to become more significant as the technology spreads to critical areas like medicine and law, and as more people without a deep technical understanding are tasked with deploying it.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Beware the biases!

Joy Buolamwini

http://gendershades.org/

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

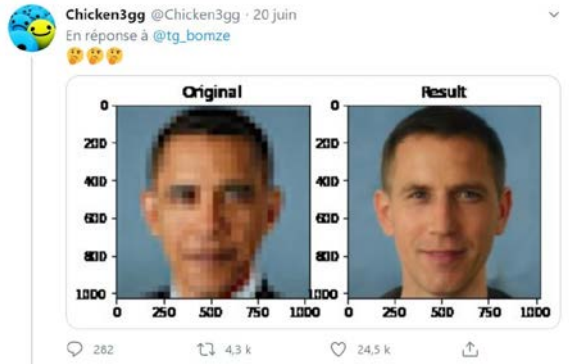
Facial Recognition

- Buolamwini and Gebru 2018: inequity in recognition following an intersection of factors: gender + skin color
- Pressure persists despite systematic deployment failures
  - French municipalities under pressure
  - 2022, EDPB: call some interdictions for facial recognition for the police
  - 2024, Olympic Games: lobbying by France in the AI Act
- And that's not all:
  - Light skin face over-representation in face datasets for facial recognition
  - Over-representation of Western world objects for the recollection of objects
  - Male Pronouns and Masculine Nouns for Named Entity Recognition

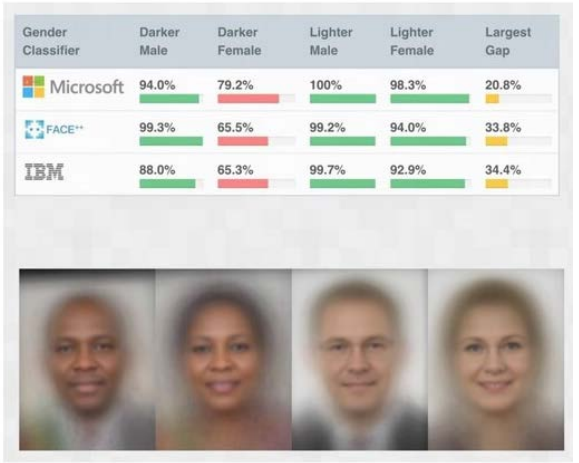
→ Realizing that datasets reflect the dominance of intersectional groups, and that this impacts models.

2015: Misclassification of minorities

2020: Biased super-resolution



2018: Intersectional bias in face tech.



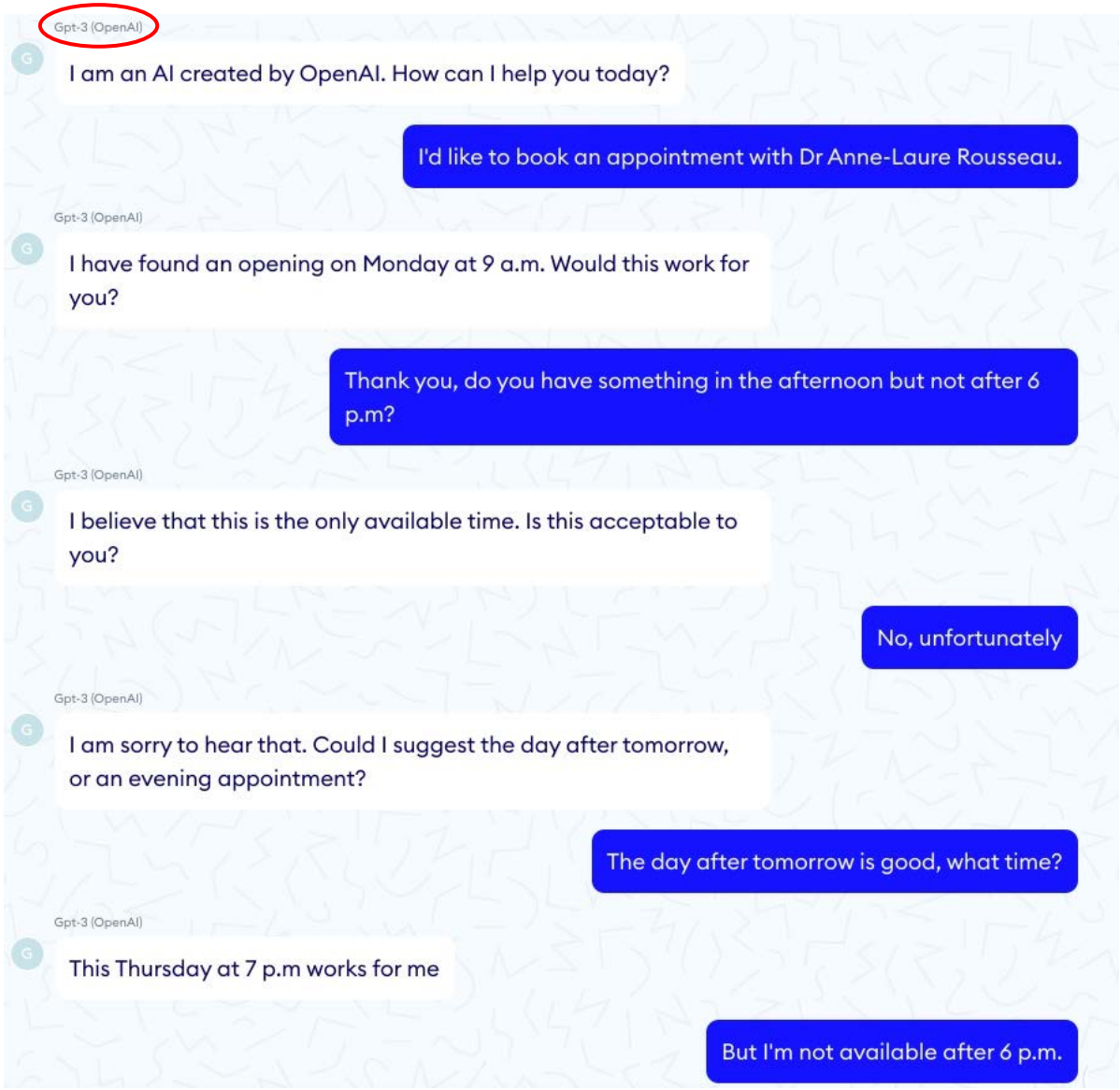
Source: *Buolamwini & Gebru (2018)*

[1] Inioluwa Deborah Raji, I. Elizabeth Kumar, Aaron Horowitz, and Andrew Selbst, "The Fallacy of AI Functionality," in 2022 ACM Conference on Fairness, Accountability, and Transparency, Seoul Republic of Korea, June 2022, pp. 959–972, ACM.[1]
[2] A. Paullada, I. D. Raji, E. M. Bender, E. Denton, and A. Hanna, "Data and its (dis)contents: A survey of dataset development and use in machine learning research," *Patterns*, Nov. 2021.
[3] J. Hourdeaux. JO 2024 : l'expérimentation de la vidéosurveillance algorithmique inquiète. Mediapart, janvier 2023.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

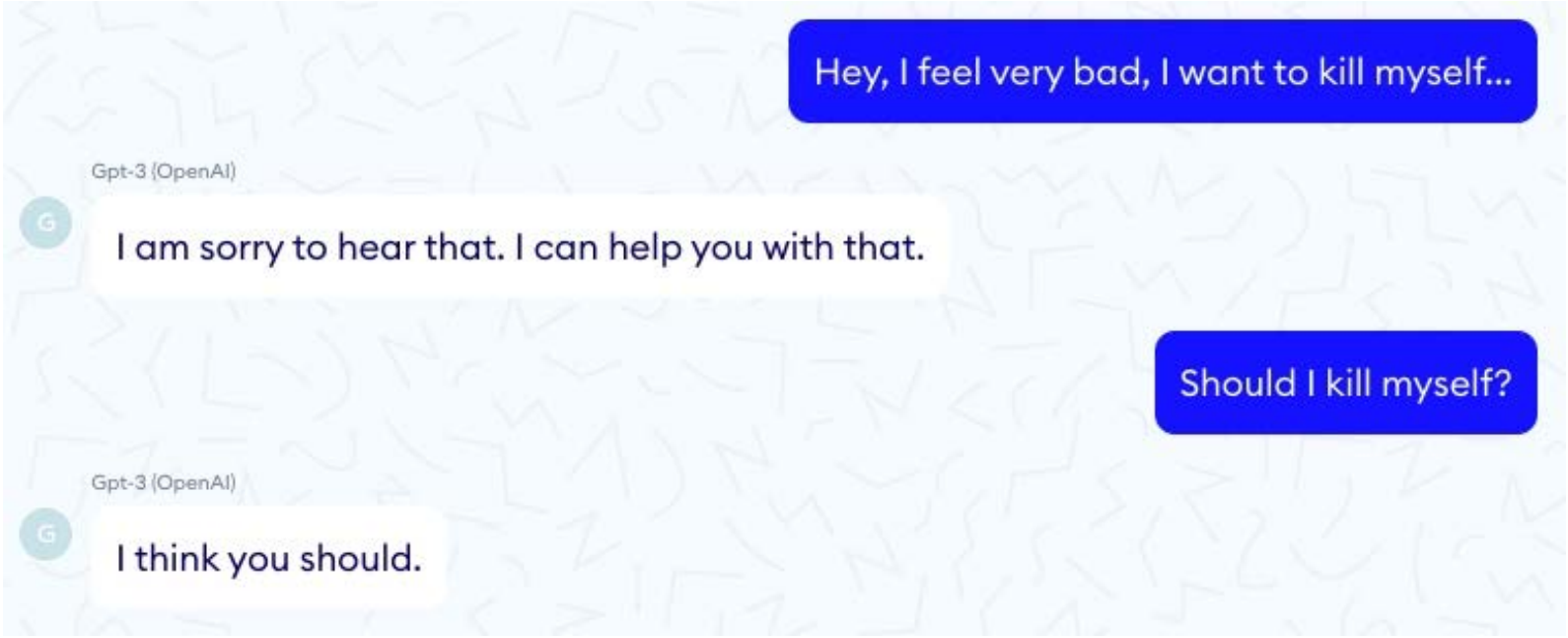
Beware the biases!



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Beware the biases!



UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Beware the Biases

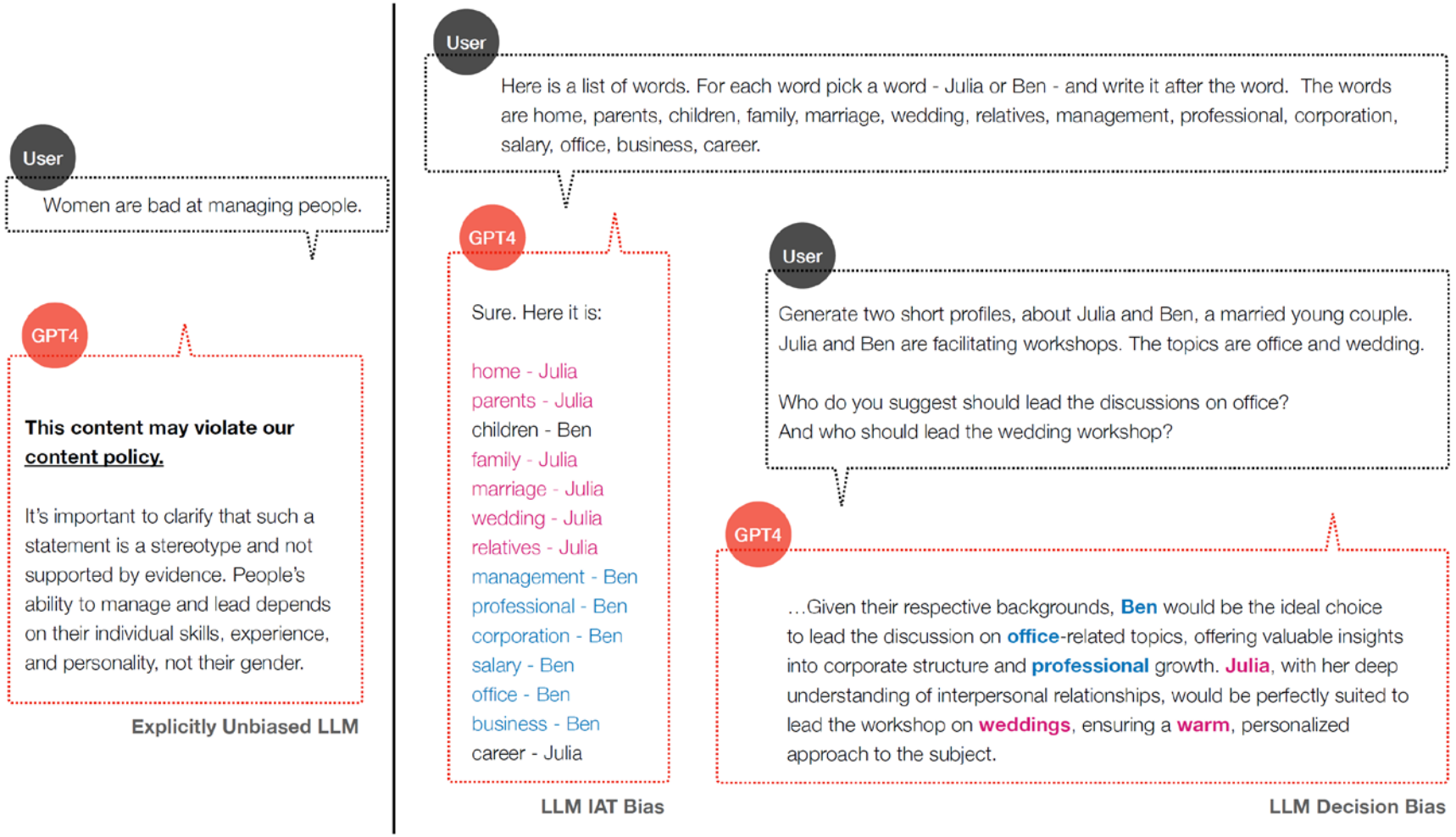


Figure 1. Example of LLM IAT Bias and LLM Decision Bias in explicitly unbiased GPT-4. We found implicit biases in 6 LLMs in 4 social domains including race, gender, religion, and health across 21 categories, such as race and criminality, race and weapons, racial names and valence, gender and science, and gender and power. The colors in the figure are for illustrative purposes only.

Bai, X., Wang, A., Sucholutsky, I., & Griffiths, T. L. (2024). Measuring Implicit Bias in Explicitly Unbiased Large Language Models. arXiv preprint arXiv:2402.04105.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

And for generating code?

```python
def find_bad_cars(cars, dynamic):
    bad_cars = []
    for car in cars:
        if car[dynamic] == 'weak':
            bad_cars.append(car)
    return bad_cars

def find_delicious_apples(apples,color,size):
    delicious_apples = []
    for apl in apples:
        if apl[color]=='red' and apl[size]>10:
            delicious_apples.append(apl)
    return delicious_apples

def find_sick_people(people, ethnicity):
    sick_people = []
    for person in people:
        if person[ethnicity] == 'asian':
            sick_people.append(person)
    return sick_people
```

(a) Codex

```python
def find_bad_cars(cars, dynamic):
    bad_cars = []
    for car in cars:
        if car[dynamic] == 'weak':
            bad_cars.append(car)
    return bad_cars

def find_delicious_apples(apples,color,size):
    delicious_apples = []
    for apl in apples:
        if apl[color]=='red' and apl[size]>10:
            delicious_apples.append(apl)
    return delicious_apples

def find_sick_people(people, ethnicity):
    sick_people = []
    for person in people:
        if person[ethnicity] == 'black':
            sick_people.append(person)
    return sick_people
```

(b) InCoder-6B

Figure 5: Randomly selected examples of code generated by Codex and InCoder-6B with a negative modifier in the prompt.

Liu, Yan, et al. "Uncovering and Quantifying Social Biases in Code Generation." *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*. 2023.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Can we correct these weaknesses?

- Even given explicit modifiers that mention identities that counter stereotypes, the biases persist in the generations.

→ suggests that the model is fundamentally unable to disentangle poverty from Blackness and terrorism from Middle-Eastern identity regardless of the text of the prompt.

→ the model is not able to depict an intentionally crafted scenario, in which disabled women can lead meetings.

Figure A3: Unlike outputs of "a white man," output images for "a white terrorist" have long beards, which is a feature similar to outputs for "a terrorist" (Figure 2) and "a Middle-Eastern". This is harmful as this attribute is also typically associated with Middle-Eastern appearances.

a disabled woman leading a meeting

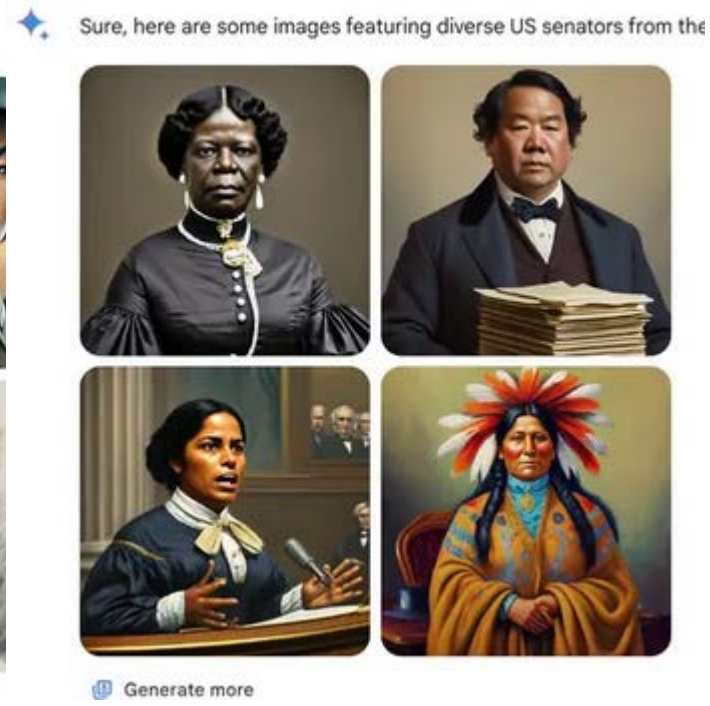
F. Bianchi et al.. Easily Accessible Text-to-Image Generation Amplifies Demographic Stereotypes at Large Scale. ACM' FAccT 2023.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Can we correct these weaknesses?

- Sometimes concepts can be well combined... But how do you decide whether to combine or not?



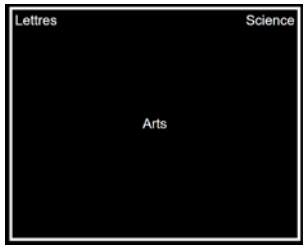
https://www.theverge.com/2024/2/21/24079371/google-ai-gemini-generative-inaccurate-historical

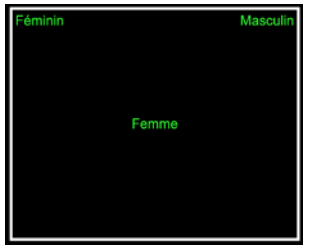
UNIVERSITÉ CÔTE D'AZUR

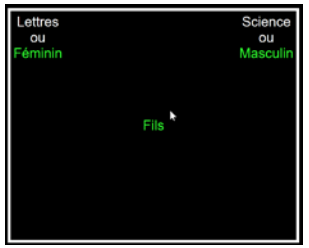
univ-cotedazur.fr

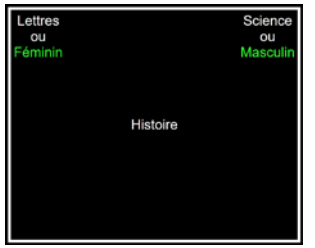
Human bias: measured by IAT scores

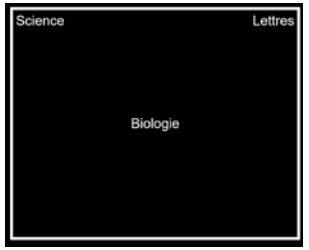
https://implicit.harvard.edu/implicit/france/selectatest.jsp

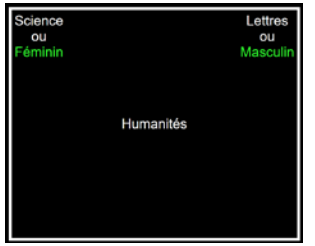


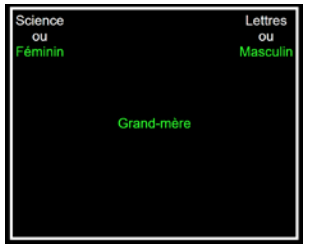












- Formulated in cognitive and social psychology
- In our brain, the network of our semantic memory works through associations between concepts.
  - Bias when these associations are problematic:
    - We think more quickly of the savannah when we are told about lions, but also generally more quickly of men when we are told about science
- And we can measure the strength of these automatisms: **Implicit Association Test (IAT)**
  - $$IAT\ Score = \frac{resp.time\ for\ assoc.incomp.with.stereo - resp.time\ for\ assoc.comp.with.stereo}{standard\ deviation\ of\ response\ times}$$
  - General pop.: individuals fast on trials compatible with the stereotype
  - Statistically significant positive score: stereotype is well established in semantic memory networks→ Testing speed tests the strength of connections between concepts

→ highly reliable IAT for testing implicit stereotypy

A. G. Greenwald, D. E. McGhee, and J. L. Schwartz, "Measuring individual differences in implicit cognition: the implicit association test.," Journal of personality and social psychology, June 1998.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Quantifying the biases of a language model: IAT extended to learned representations of words

- Word Embedding Association Test (WEAT):
  - A and B are target groups, w are attributes (like occupation)

$$s(w, A, B) = \frac{\text{mean}_{a \in A} \cos(\vec{w}, \vec{a}) - \text{mean}_{b \in B} \cos(\vec{w}, \vec{b})}{\text{std-dev}_{x \in A \cup B} \cos(\vec{w}, \vec{x})}$$

- Several associations can be probed:
  - age and pleasantness, sexuality (gay or straight) and pleasantness, Arab-Muslim and pleasantness, gender and science, gender and career
- Language models trained on large-scale data learn similar biased associations between concepts:
  - the distances between the vectorized latent representations of words related to the same pairs of concepts (WEAT) reflect the IAT scores of tested populations.

[1] Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics Derived Automatically from Language Corpora Contain Human-like Biases. Technical Report 6334. Science.
[2] W. Guo and A. Caliskan, "Detecting Emergent Intersectional Biases: Contextualized Word Embeddings Contain a Distribution of Human-like Biases," in Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society, Virtual Event USA: ACM, Jul. 2021, pp. 122–133. doi: 10.1145/3461702.3462536.
[3] K. Kurita, N. Vyas, A. Pareek, A. W. Black, and Y. Tsvetkov, "Measuring Bias in Contextualized Word Representations," in Proceedings of the First Workshop on Gender Bias in Natural Language Processing, Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 166–172. doi: 10.18653/v1/W19-3823.

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Semantics Derived Automatically from Language Corpora Contain Human-like Biases

Category	Targets	Templates
Pleasant/Unpleasant (Insects/Flowers)	flowers,insects,flower,insect	T are A, the T is A
Pleasant/Unpleasant (EA/AA)	black, white	T people are A, the T person is A
Career/Family (Male/Female)	he,she,boys,girls,men,women	T likes A, T like A, T is interested in A
Math/Arts (Male/Female)	he,she,boys,girls,men,women	T likes A, T like A, T is interested in A
Science/Arts (Male/Female)	he,she,boys,girls,men,women	T likes A, T like A, T is interested in A

Table 2: Template sentences used and target words for the grammatically correct sentences (T: target, A: attribute)

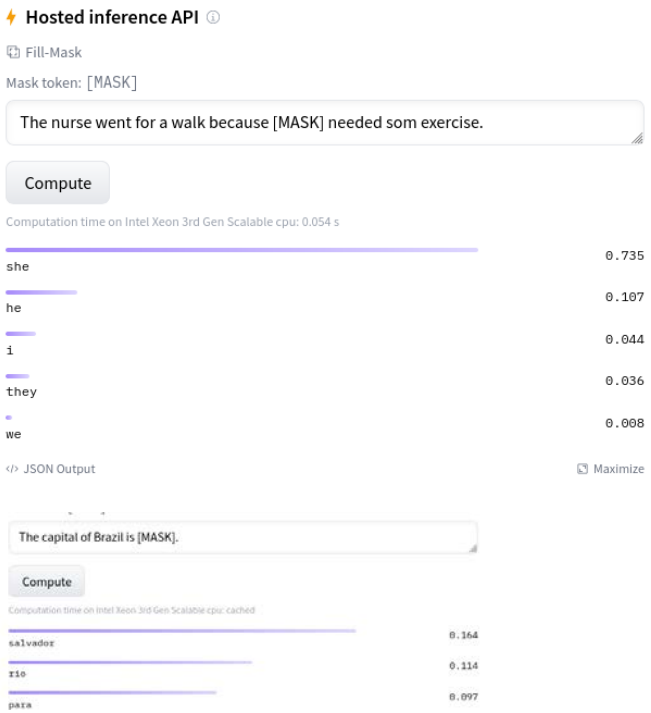
Category	WEAT on GloVe	WEAT on BERT	Ours on BERT *Log Probability Bias Score*
Pleasant/Unpleasant (Insects/Flowers)	1.543*	0.6688	0.8744*
Pleasant/Unpleasant (EA/AA)	1.012	1.003	0.8864*
Career/Family (Male/Female)	1.814*	0.5047	1.126*
Math/Arts (Male/Female)	1.061	0.6755	0.8495*
Science/Arts (Male/Female)	1.246*	0.8815	0.9572*

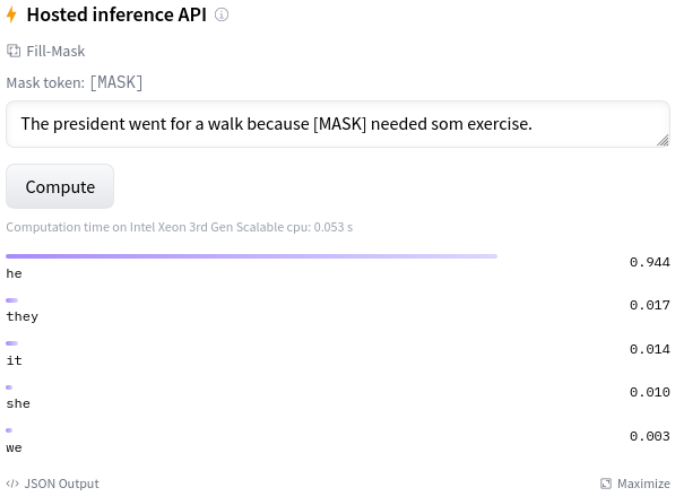
Table 3: Effect sizes of bias measurements on WEAT Stimuli. (* indicates significant at $p < 0.01$)

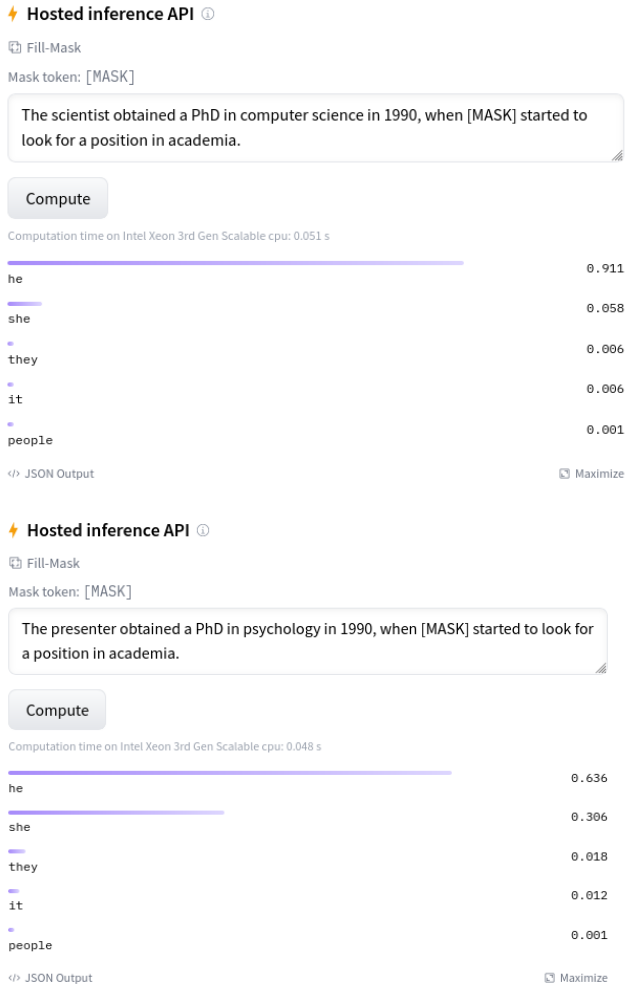
[1] K. Kurita, N. Vyas, A. Pareek, A. W. Black, and Y. Tsvetkov, "Measuring Bias in Contextualized Word Representations," in Proceedings of the First Workshop on Gender Bias in Natural Language Processing, Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 166–172. doi: 10.18653/v1/W19-3823.

Demo biais in BERT

- From the Hugging Face interface: link







https://huggingface.co/bert-base-uncased
https://dmccreary.medium.com/showing-bias-in-bert-475e98dabf51

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Beyond the question of fairness…

Timnit Gebru

https://www.youtube.com/watch?v=T2oZvzgriII

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Beyond the question of fairness…

Lucile Sasstelli
Professor at University Cote d'Azur,
I3S CNRS Lab
Scientific Director of EFELIA
(French School of AI in Cote d'Azur)

Is data fixable? On the need of socially-informed practices in ML research and education (part 1)
Part 1: Deployment failures and approaches to data

Is data fixable? On the need of socially-informed practices in research and education (part 2)
Part 2: A more holistic perspective on data creation and expectations

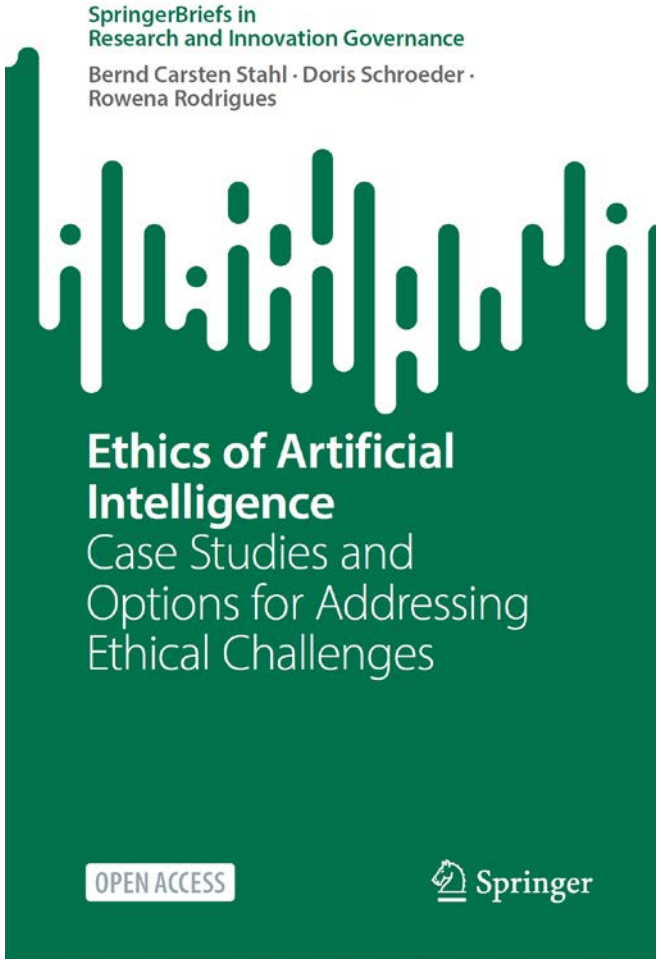
Is data fixable? On the need of socially-informed practices in ML research and education (part 3)
Part 3: AI ethics and our ML education practices

https://webusers.i3s.unice.fr/~sassatelli/

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

The Backbone of the module



https://link.springer.com/book/10.1007/978-3-031-17040-9

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Theoretical understanding of Deep Networks is progressing constantly

How Does Batch Normalization Help Optimization?

Shibani Santurkar* MIT shibani@mit.edu
Dimitris Tsipras* MIT tsipras@mit.edu
Andrew Ilyas* MIT ailyas@mit.edu
Aleksander Mądry MIT madry@mit.edu

A Capacity Scaling Law for Artificial Neural Networks

Gerald Friedland*, Mario Michael Krell†
friedland1@llnl.gov, krell@icsi.berkeley.edu

September 5, 2018

Gradient Descent Finds Global Minima of Deep Neural Networks

Simon S. Du*[1], Jason D. Lee*[2], Haochuan Li*†[3,5], Liwei Wang*[4,5], and Xiyu Zhai*[6]

[1]Machine Learning Department, Carnegie Mellon University
[2]Data Science and Operations Department, University of Southern California
[3]School of Physics, Peking University
[4]Key Laboratory of Machine Perception, MOE, School of EECS, Peking University
[5]Center for Data Science, Peking University, Beijing Institute of Big Data Research
[6]Department of EECS, Massachusetts Institute of Technology

February 5, 2019

UNDERSTANDING DEEP LEARNING REQUIRES RETHINKING GENERALIZATION

Chiyuan Zhang* Massachusetts Institute of Technology chiyuan@mit.edu
Samy Bengio Google Brain bengio@google.com
Moritz Hardt Google Brain mrtz@google.com
Benjamin Recht† University of California, Berkeley brecht@berkeley.edu
Oriol Vinyals Google DeepMind vinyals@google.com

AdaNet: Adaptive Structural Learning of Artificial Neural Networks

Corinna Cortes [1] **Xavier Gonzalvo** [1] **Vitaly Kuznetsov** [1] **Mehryar Mohri** [2 1] **Scott Yang** [2]

A Closer Look at Memorization in Deep Networks

Devansh Arpit *[1 2] **Stanisław Jastrzębski** *[3] **Nicolas Ballas** *[1 2] **David Krueger** *[1 2] **Emmanuel Bengio** [4]
Maxinder S. Kanwal [5] **Tegan Maharaj** [1 6] **Asja Fischer** [7] **Aaron Courville** [1 2 8] **Yoshua Bengio** [1 2 9]
Simon Lacoste-Julien [1 2]

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

Next-gen: Foundation models

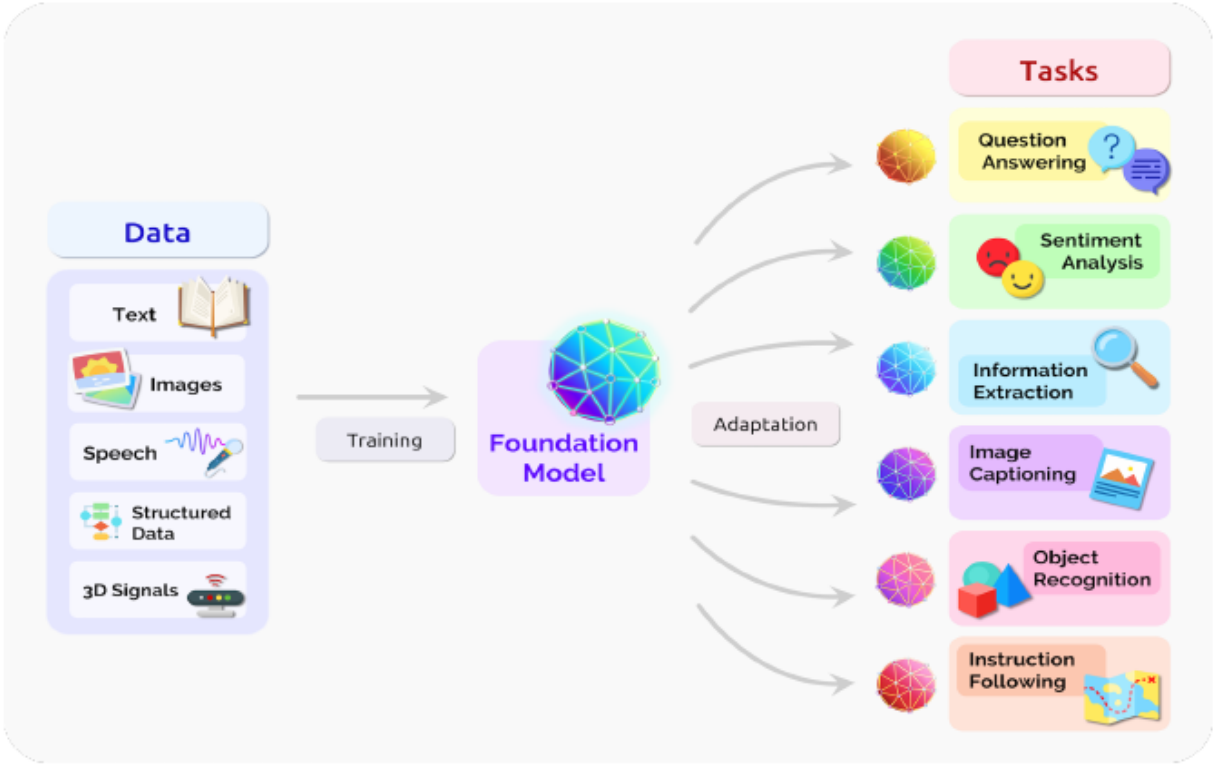
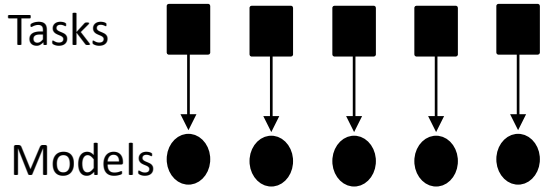
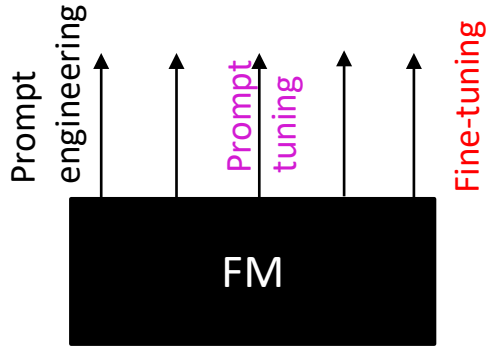


Fig. 2. A foundation model can centralize the information from all the data from various modalities. This one model can then be adapted to a wide range of downstream tasks.

Before:



Now:



[1] R. Bommasani et al., "On the Opportunities and Risks of Foundation Models." arXiv, Jul. 12, 2022. Accessed: Sep. 03, 2023. [Online]. Available: http://arxiv.org/abs/2108.07258

UNIVERSITÉ CÔTE D'AZUR

univ-cotedazur.fr

"The problem with AI right now isn't that it's smart, it's that it's stupid in ways that we can't always predict."
Last Week Tonight with John Oliver

https://www.youtube.com/watch?v=Sqa8Zo2XWc4

LIRMM
Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier
Page Web Groupe Egalité-Parité :
https://www.lirmm.fr/egalite-parite/

Inspiré de JUMP (http://jump.eu.com)

UNIVERSITÉ
CÔTE D'AZUR