

COMP5046

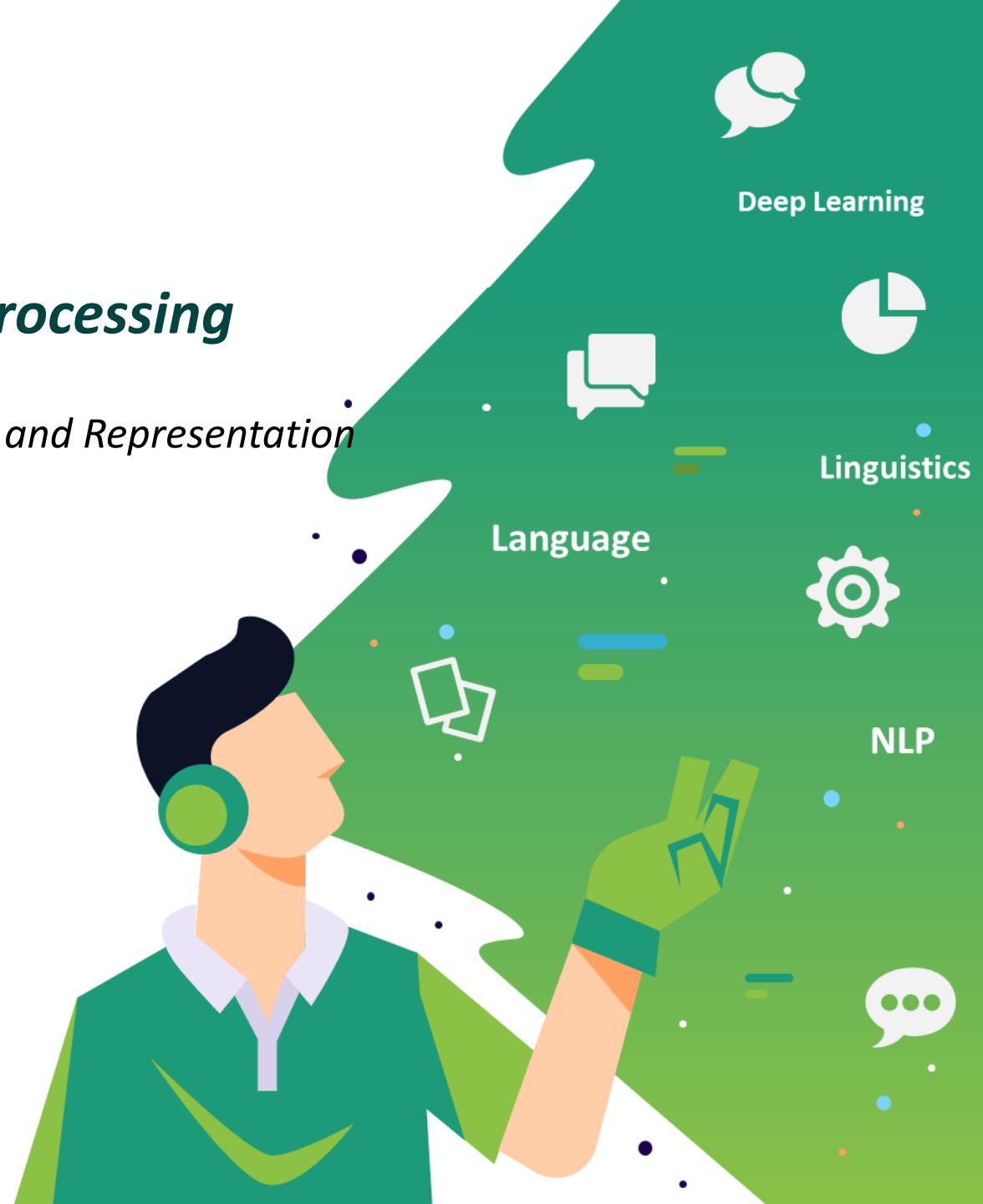
Natural Language Processing

Lecture 2: Word Embeddings and Representation

Dr. Caren Han

Semester 1, 2022

School of Computer Science,
University of Sydney



0 LECTURE PLAN

Lecture 2: Word Embeddings and Representation

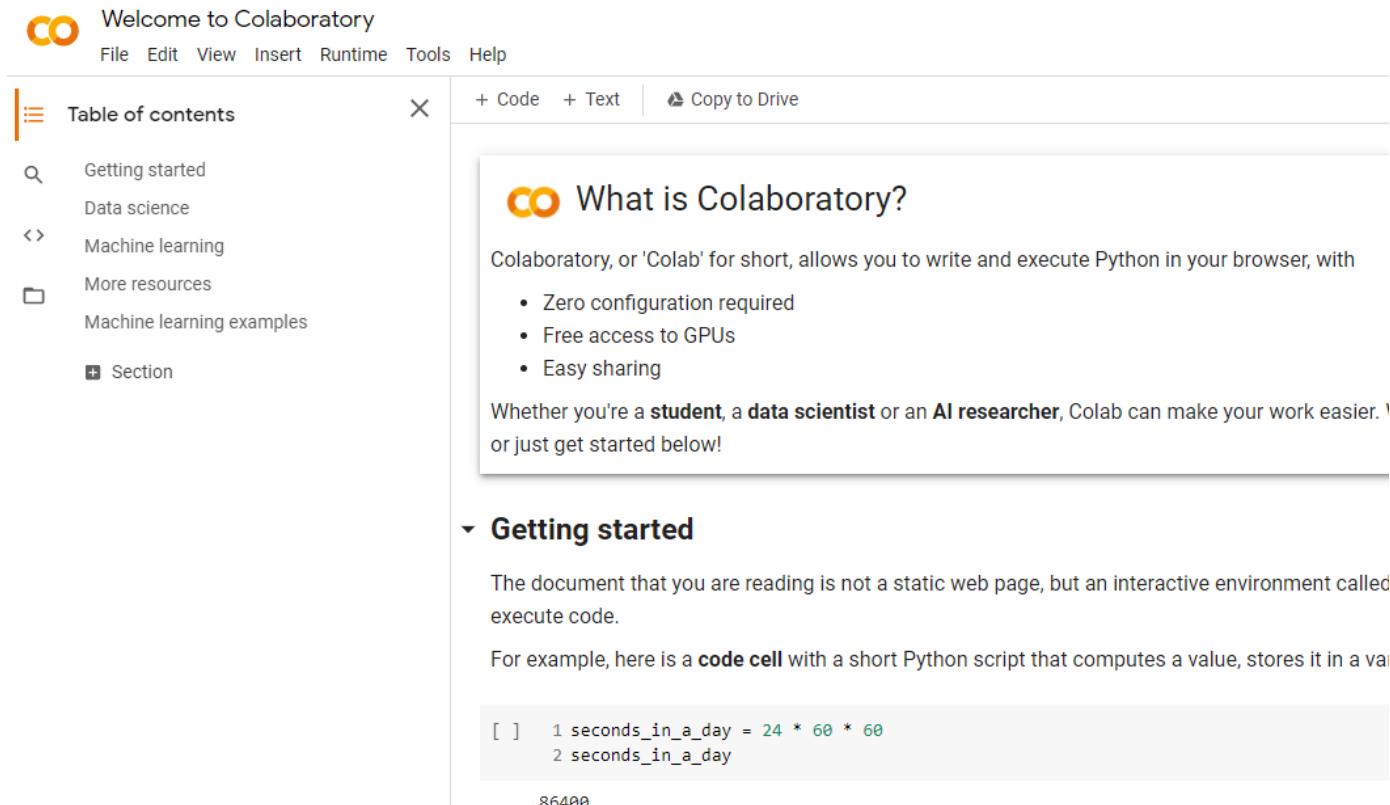
1. Lab Info
2. Previous Lecture Review
 1. Word Meaning and WordNet
 2. Count based Word Representation
3. Prediction based Word Representation
 1. Introduction to the concept 'Prediction'
 2. Word2Vec
 3. FastText
 4. GloVe
4. Next Week Preview

1 Info: Lab Exercise

What do we do during Labs?

In Labs, Students will use Google Colab

Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud. With Colaboratory you can write and execute code, save and share your analyses, and access powerful computing resources, all for free from your browser.



Welcome to Colaboratory

File Edit View Insert Runtime Tools Help

Table of contents

- Getting started
- Data science
- Machine learning
- More resources
 - Machine learning examples
- Section

+ Code + Text Copy to Drive

What is Colaboratory?

Colaboratory, or 'Colab' for short, allows you to write and execute Python in your browser, with

- Zero configuration required
- Free access to GPUs
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier.¹ or just get started below!

Getting started

The document that you are reading is not a static web page, but an interactive environment called execute code.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result.

```
[ ] 1 seconds_in_a_day = 24 * 60 * 60
2 seconds_in_a_day
```

86400

1 Info: Lab Exercise

Submissions

How to Submit

Students should submit “**ipynb**” file (Download it from “File” > “Download .ipynb”) to Canvas.

When and Where to Submit

Students must submit the Lab 1(for Week2) by **Week 3 Monday 11:59PM**.



- Lab1: Introduction to the NLP (PyTorch)**

Check [How to Use the Colab file for your Lab](#)

 - Lab1: [Lab1 Colab](#) (Release date: 27/02/2022)
 - Submission: [Lab1_Submission](#) (Due: 07/03/2022 11:59PM)
 - Sample Solution: [Lab1 Sample Answer](#) (Release date: 14/03/2022 9PM)

The sample solution is not a model answer. There are several ways to solve the lab exercise!
 - Mark Release Date ([Release date: 21/03/2022](#))

0 LECTURE PLAN

Lecture 2: Word Embeddings and Representation

1. Lab Info
2. **Count-based Word Representation**
 1. Word Meaning
 2. Limitations
3. Prediction based Word Representation
 1. Introduction to the concept 'Prediction'
 2. Word2Vec
 3. FastText
 4. GloVe
4. Next Week Preview

How to represent the meaning of the word?

Definition: meaning (Collins dictionary).

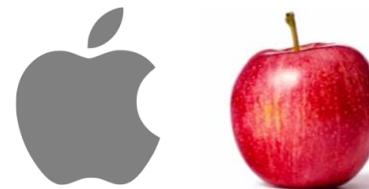
- the idea that it represents, and which can be explained using other words.
- the thoughts or ideas that are intended to be expressed by it.

signifier (symbol) \Leftrightarrow signified (idea or thing) = denotation

“Computer”



“Apple”



\x63\x6f\x6d\x70\x75\x74\x65\x72

\x61\x70\x70\x6c\x65

*Unicode (utf-8)

COUNT based WORD REPRESENTATION

Problem with one-hot vectors

Problem #1. No word similarity representation

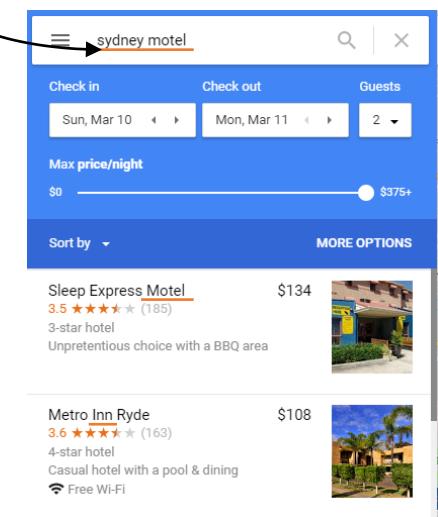
Example: in web search, if user searches for “Sydney motel”, we would like to match documents containing “Sydney Inn”

hotel *motel* *Inn*

motel = [0 0 0 0 0 0 0 0 1 0 0 0 0 0 ... 0]

hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 ... 0]

Inn = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ... 1]



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

Problem #2. Inefficiency

Vector dimension = number of words in vocabulary

Each representation has only a single ‘1’ with all remaining 0s.

Problem with BoW (Bag of Words)

- The intuition is that documents are similar if they have similar content. Further, that from the content alone we can learn something about the meaning of the document.
- **Discarding word order** ignores the context, and in turn meaning of words in the document (semantics). Context and meaning can offer a lot to the model, that if modeled could tell the difference between the same words differently arranged (“this is interesting” vs “is this interesting”).

S1= I love you but you hate me

S2= I hate you but you love me



Limitation of Term Frequency Inverse Document Frequency

$$w_{i,j} = tf_{i,j} \times \log \left(\frac{N}{df_i} \right)$$

$w_{i,j}$ = weight of term i in document j

$tf_{i,j}$ = number of occurrences of term i in document j

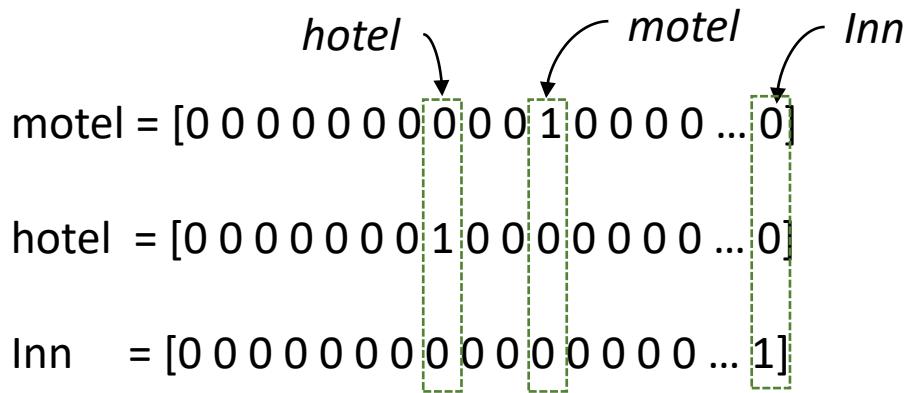
N = total number of documents

df_i = number of documents containing term i

- It computes document similarity directly in the word-count space, which may be slow for large vocabularies.
- It assumes that the counts of different words provide independent evidence of similarity.
- It makes no use of ***semantic similarities between words.***

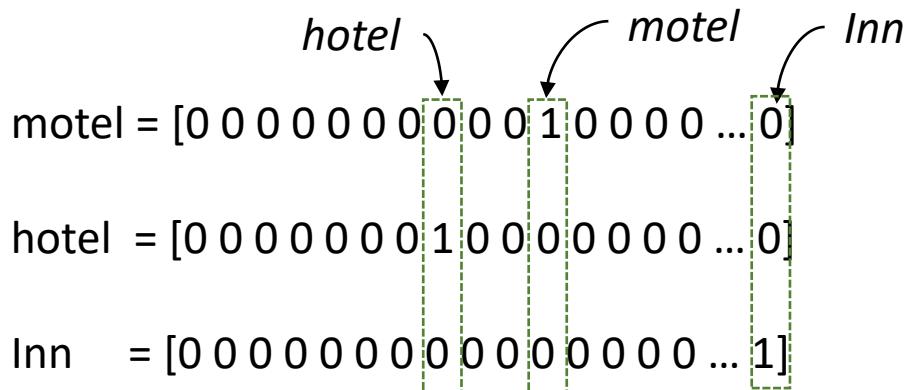
Sparse Representation

With **COUNT based word representation** (especially, one-hot vector), linguistic information was represented with **sparse representations** (high-dimensional features)



Sparse Representation

With **COUNT based word representation** (especially, one-hot vector), linguistic information was represented with **sparse representations** (high-dimensional features)



A Significant Improvement Required!

1. How to get the low-dimensional vector representation
2. How to represent the word similarity

maybe a low-dimensional vector?

Can we use a list of fixed numbers (properties) to represent the word?

0 LECTURE PLAN

Lecture 2: Word Embeddings and Representation

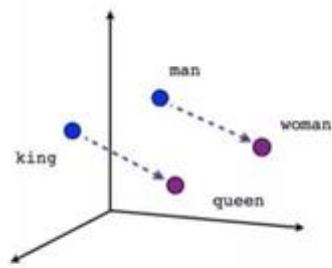
1. Lab Info
2. Previous Lecture Review
 1. Word Meaning and WordNet
 2. Count based Word Representation
3. **Prediction based Word Representation**
 1. Word Embedding
 2. Word2Vec
 3. FastText
 4. Glove
4. Next Week Preview

3

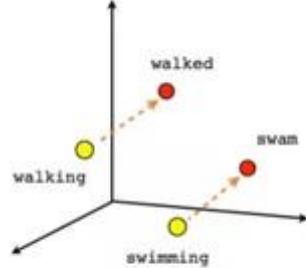
Prediction based Word representation

How to Represent the Word Similarity!

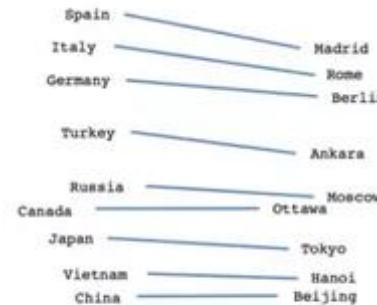
- How to represent the word similarity with dense vector



Male-Female



Verb tense



Country-Capital

- Try this with word2vec

Word Algebra

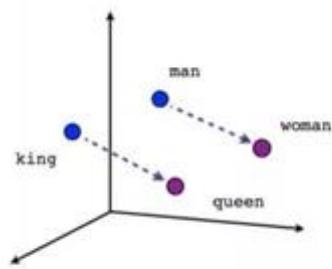
Enter all three words, the first two, or the last two and see the words that result.

<input type="text" value="shanghai"/>	+	<input type="text" value="australia"/>	-	<input type="text" value="sydney"/>	=	<input type="button" value="Get result"/>
<u>china</u>	0.7477672216910414					

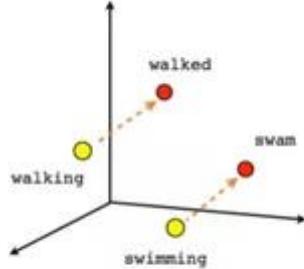
3

Prediction based Word representation

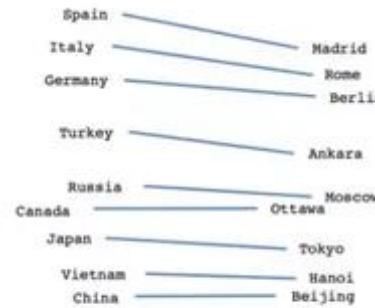
Let's make the word representation



Male-Female



Verb tense



Country-Capital

We need to...

1. Have the fixed low-dimensional vector representation
2. Represent the word similarity

maybe a low-dimensional vector?

What if we use a list of fixed numbers (properties) to represent the word?

3

Prediction based Word representation

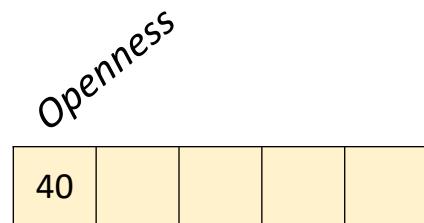
Let's get familiar with using vectors to represent things

Assume that you are taking a personality test (the Big Five Personality Traits test)

1)Openness, 2)Agreeableness, 3)Conscientiousness, 4)Negative emotionality, 5)Extraversion



Jane



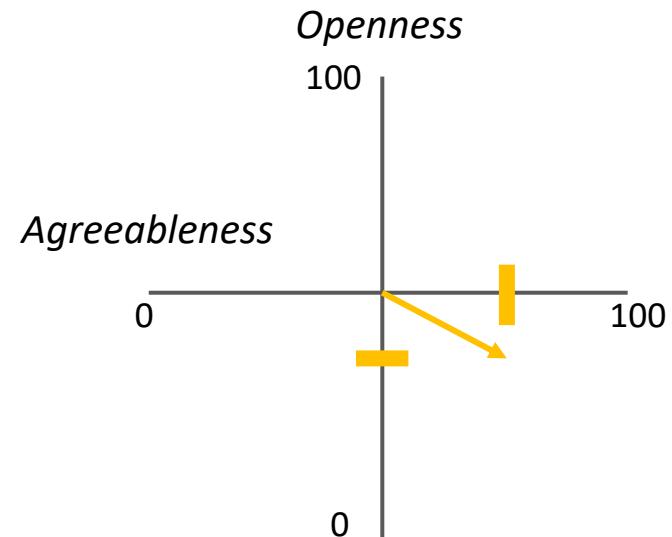
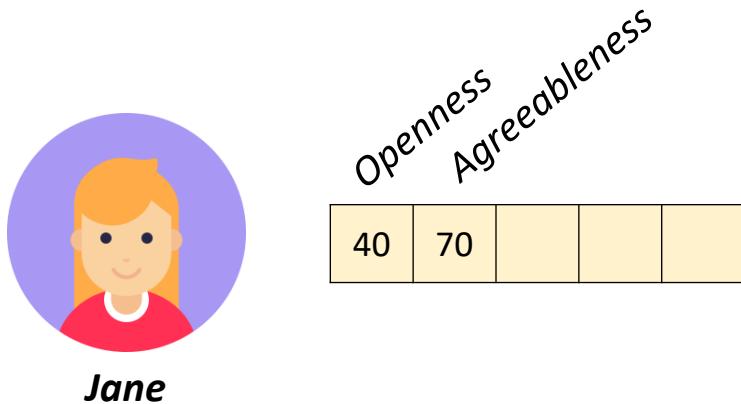
3

Prediction based Word representation

Let's get familiar with using vectors to represent things

Assume that you are taking a personality test (the Big Five Personality Traits test)

1)Openness, 2)Agreeableness, 3)Conscientiousness, 4)Negative emotionality, 5)Extraversion



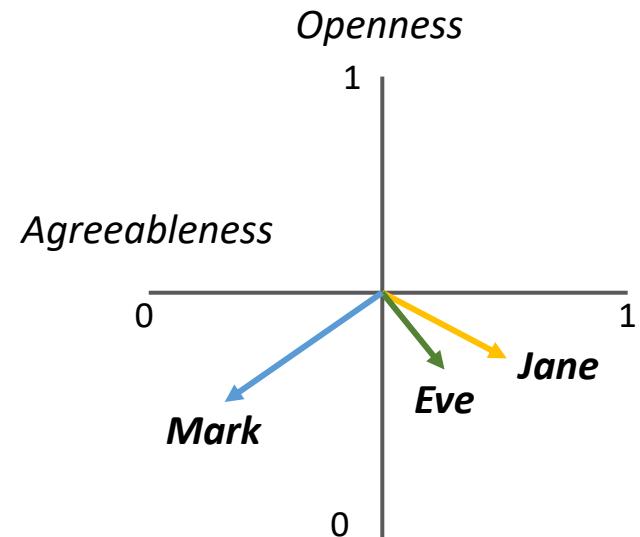
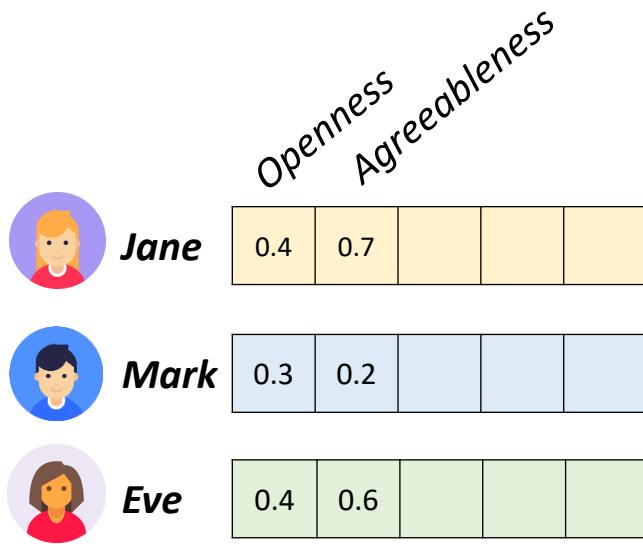
3

Prediction based Word representation

Let's get familiar with using vectors to represent things

Assume that you are taking a personality test (the Big Five Personality Traits test)

1)Openness, 2)Agreeableness, 3)Conscientiousness, 4)Negative emotionality, 5)Extraversion



3

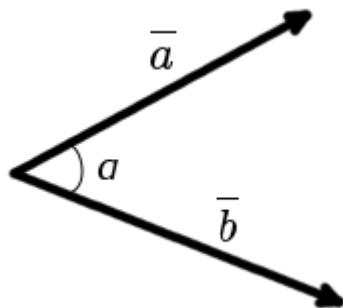
Prediction based Word representation

Let's get familiar with using vectors to represent things

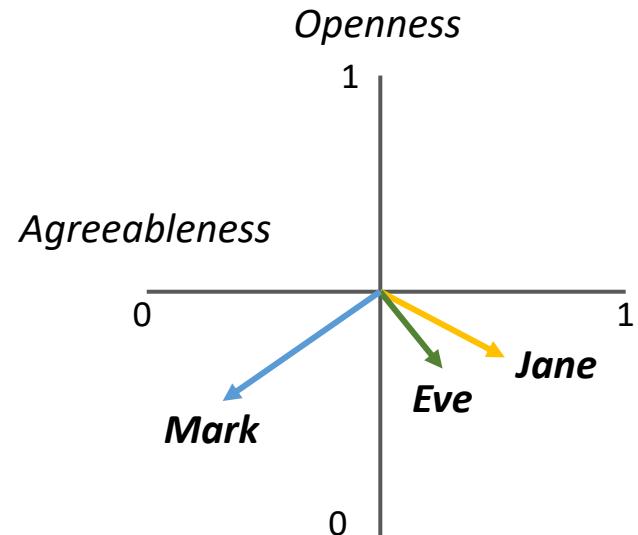
Which of two people (Mark or Eve) is more similar to Jane?

Cosine Similarity

Measure of similarity between two vectors
of inner product space that measures the
cosine of the angle between them



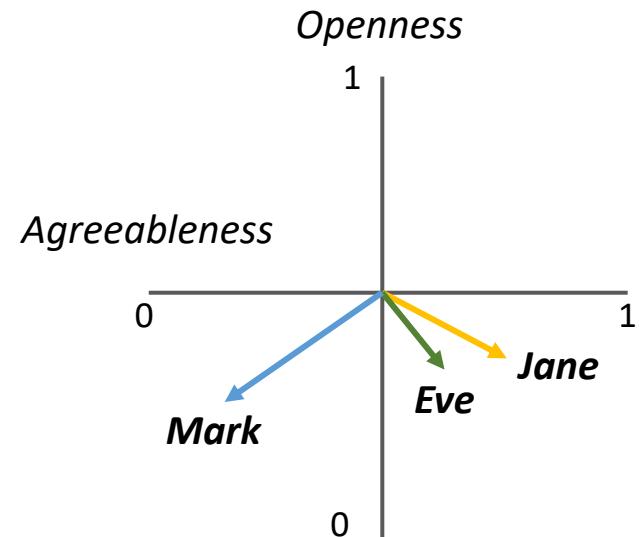
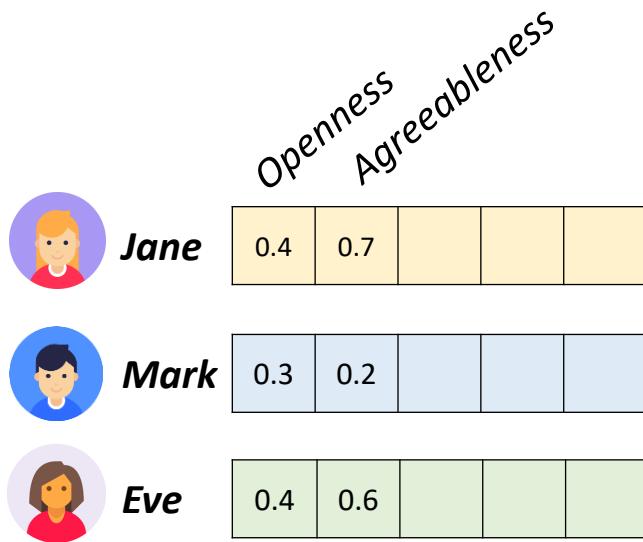
$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}},$$



3 Prediction based Word representation

Let's get familiar with using vectors to represent things

Which of two people (Mark or Eve) is more similar to Jane?



$$\cos\left(\begin{bmatrix} 0.4 & 0.7 \end{bmatrix}, \begin{bmatrix} 0.3 & 0.2 \end{bmatrix}\right) \approx 0.89$$

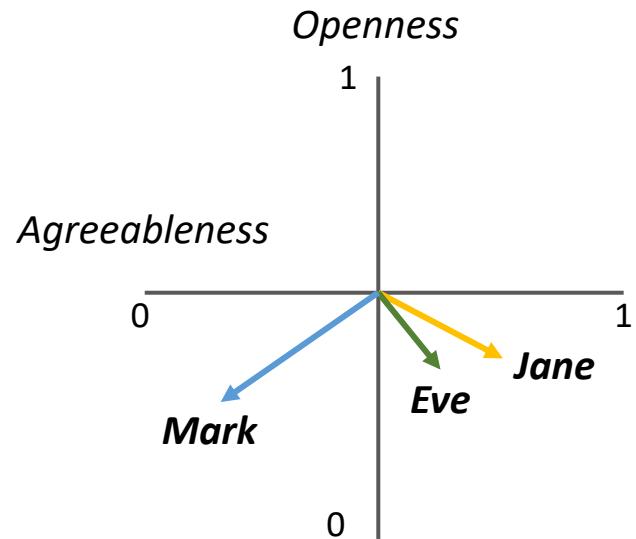
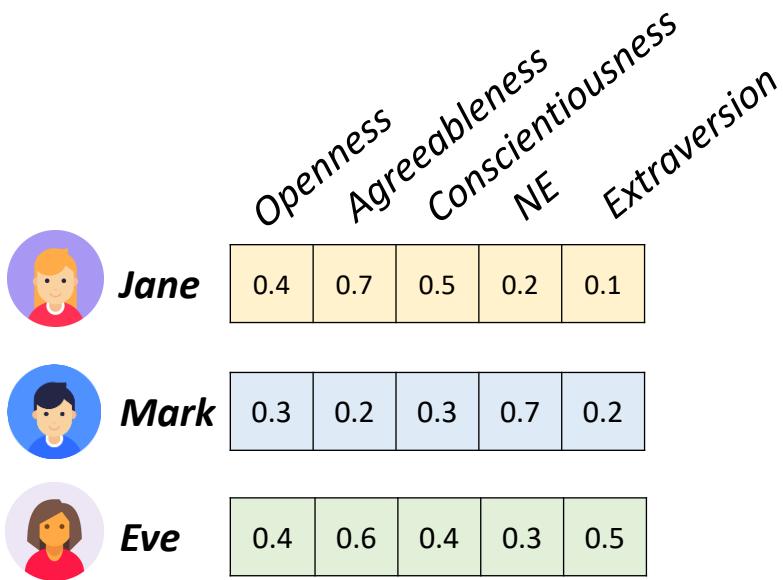
$$\cos\left(\begin{bmatrix} 0.4 & 0.7 \end{bmatrix}, \begin{bmatrix} 0.4 & 0.6 \end{bmatrix}\right) \approx 0.99$$

3

Prediction based Word representation

Let's get familiar with using vectors to represent things

We need all five major factors for represent the personality



With these embeddings,

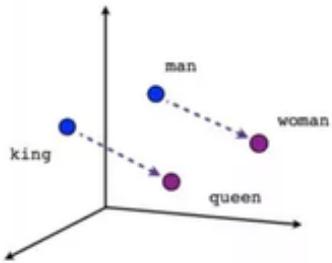
1. Represent things as vectors of fixed numbers!

2. Easily calculate the similarity between vectors

3

Prediction based Word representation

Remember? The Word2Vec Demo!

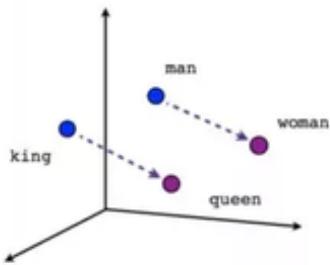


This is a word embedding for the word “king”

3

Prediction based Word representation

Remember? The Word2Vec Demo!



This is a word embedding for the word “king”

* Trained by Wikipedia Data, 50-dimension GloVe Vector

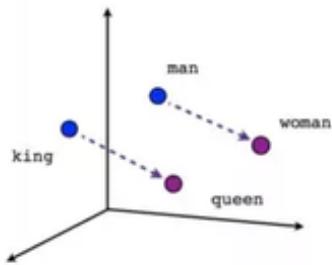
king

```
[0.50451, 0.68607, -0.59517, -0.022801, 0.60046, 0.08813, 0.47377, -0.61798, -0.31012,
-0.066666, 1.493, -0.034173, -0.98173, 0.68229, 0.812229, 0.81722, -0.51722, -744.54
1503, -0.55809, 0.66421, 0.1961, -0.1495, -0.033474, -0.30344, 0.41177, -2.223, -1.0756,
-0.343554, 0.33505, 1.9927, -0.042434, -0.64519, 0.72519, 0.71419, 0.714319, 0.71419
9159, 0.16754, 0.34344, -0.25663, -0.8523, 0.1661, 0.40102, 1.1685, -1.0137, -0.2155,
0.78321, -0.91241, -1.6626, -0.64426, -0.542102]
```

3

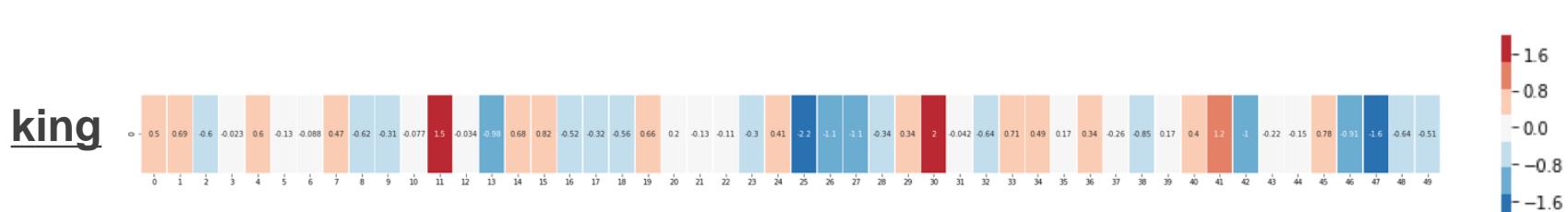
Prediction based Word representation

Remember? The Word2Vec Demo!



This is a word embedding for the word “king”

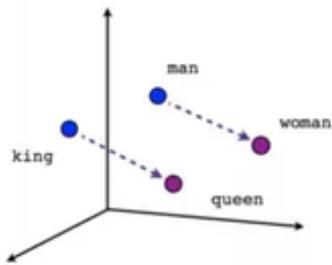
* Trained by Wikipedia Data, 50-dimension GloVe Vector



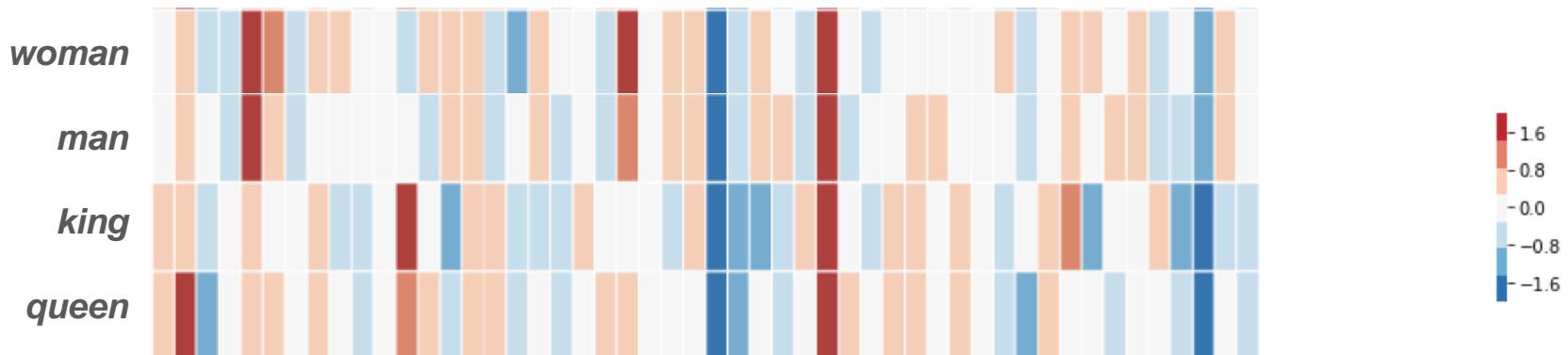
3

Prediction based Word representation

Remember? The Word2Vec Demo!



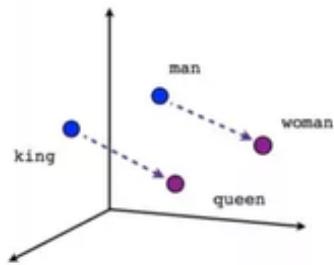
Compare with Woman, Man, King, and Queen



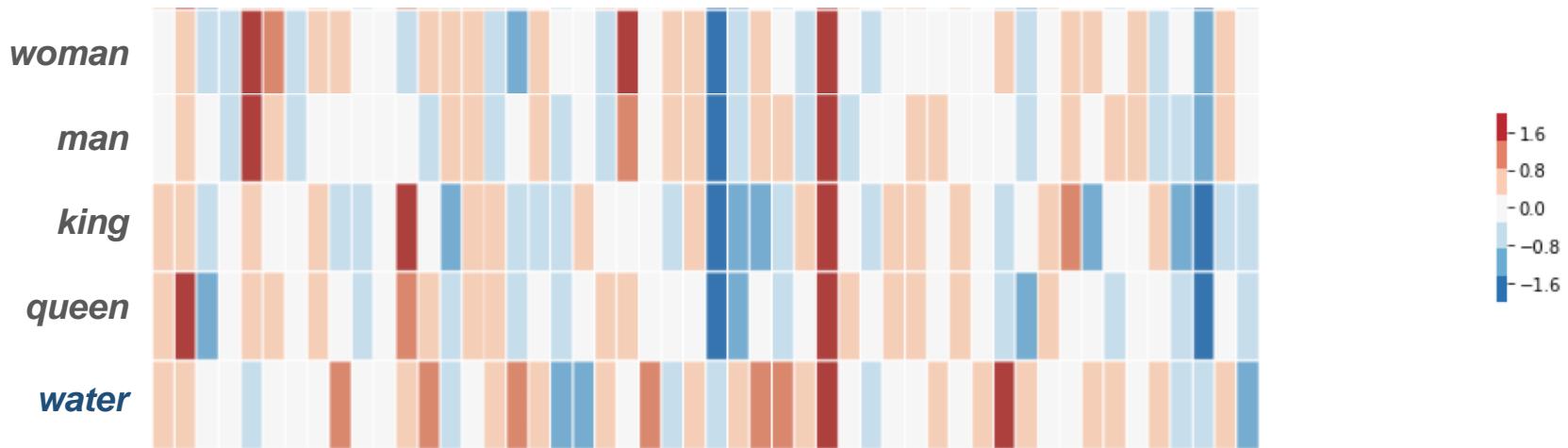
3

Prediction based Word representation

Remember? The Word2Vec Demo!



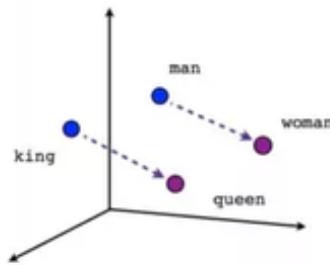
Compare with Woman, Man, King, Queen, and Water



3

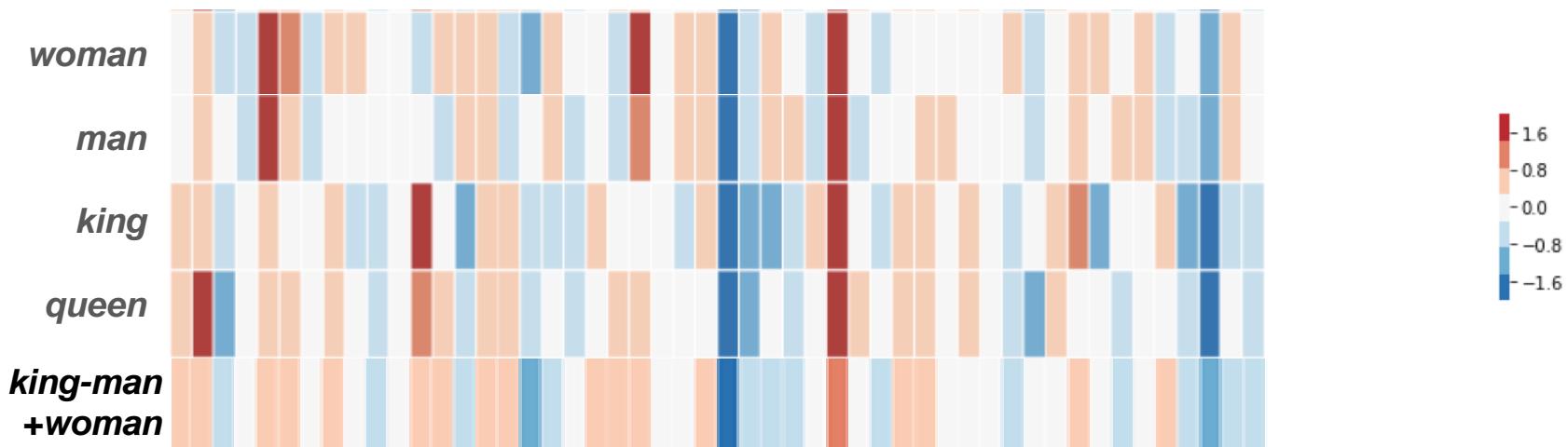
Prediction based Word representation

Remember? The Word2Vec Demo!



$$\text{king} - \text{man} + \text{woman} \approx \text{queen}?$$

Word Algebra



How to make dense vectors for word representation

Prediction based Word representation

How to make dense vectors for word representation



Prof. John Rupert Firth

Distributional Hypothesis

“You shall know a word by the company it keeps”

— (Firth, J. R. 1957:11)

*Prof. Firth is noted for drawing attention to the context-dependent nature of meaning with his notion of '**context of situation**', and his work on **collocational meaning** is widely acknowledged in the field of **distributional semantics**.*

Prediction based Word representation

Word Representations in the context

When a word w appears in a text, its context is the set of words that appear nearby

- Use the surrounding contexts of w to build up a representation of w

Article
Talk
Read
Edit
View history
Se

Sydney

From Wikipedia, the free encyclopedia

This article is about the Australian metropolis. For the local government area, see [City of Sydney](#). For other uses, see [Sydney \(disambiguation\)](#).

Sydney (/'sɪdnɪ/ ( listen) *SID-nee*)^[7] is the state capital of New South Wales and the most populous city in Australia and Oceania.^[8] Located on Australia's east coast, the metropolis surrounds Port Jackson and extends about 70 km (43.5 mi) on its periphery towards the Blue Mountains to the west, Hawkesbury to the north, the Royal National Park to the south and Macarthur to the south-west.^[9] Sydney is made up of 658 suburbs, 40 local government areas and 15 contiguous regions. Residents of the city are known as "Sydneysiders".^[10] As of June 2017, Sydney's estimated metropolitan population was 5,131,326,^[11] and is home to approximately 65% of the state's population.^[12]

Indigenous Australians have inhabited the Sydney area for at least 30,000 years, and thousands of engravings remain throughout the region, making it one of the richest in Australia in terms of Aboriginal archaeological sites. During his first Pacific voyage in 1770, Lieutenant James Cook and his crew became the first Europeans to chart the eastern coast of Australia, making landfall at Botany Bay and inspiring British interest in the area. In 1788, the First Fleet of convicts, led by Arthur Phillip, founded Sydney as a British penal colony, the first European settlement in Australia. Phillip named the city Sydney in recognition of Thomas Townshend, 1st Viscount Sydney.^[13] Penal transportation to New South Wales ended soon after Sydney was incorporated as a city in 1842. A gold rush occurred in the colony in 1851, and over the next century, Sydney transformed from a colonial outpost into a major global cultural and economic centre. After World War II, it experienced mass migration and became one of the most multicultural cities in the world.^[3] At the time of the 2011 census, more than 250 different languages were spoken in Sydney.^[14] In the 2016 Census, about 35.8% of residents spoke a language other than English at home.^[15] Furthermore, 45.4% of the population reported having been born overseas, making Sydney the 3rd largest foreign born population of any city in the world after London and New York City, respectively.^{[16][17]}

3

Prediction based Word representation

How can we train the word representation to machine?

Neural Networks! (Machine Learning)

Article [Talk](#) [Read](#) [Edit](#) [View history](#) [Search](#)

Sydney

From Wikipedia, the free encyclopedia

This article is about the Australian metropolis. For the local government area, see [City of Sydney](#). For other uses, see [Sydney \(disambiguation\)](#).

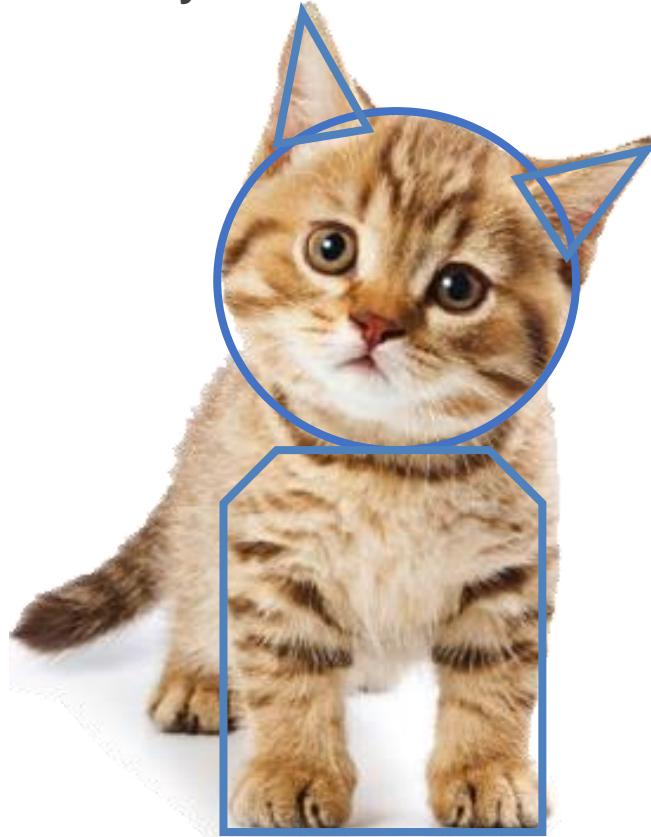
Sydney (/*sɪdnɪ/ (listen) *SID-nee*)^[7] is the state capital of New South Wales and the most populous city in Australia and Oceania.^[8] Located on Australia's east coast, the metropolis surrounds Port Jackson and extends about 70 km (43.5 mi) on its periphery towards the Blue Mountains to the west, Hawkesbury to the north, the Royal National Park to the south and Macarthur to the south-west.^[9] Sydney is made up of 658 suburbs, 40 local government areas and 15 contiguous regions. Residents of the city are known as "Sydneysiders".^[10] As of June 2017, Sydney's estimated metropolitan population was 5,131,326,^[11] and is home to approximately 65% of the state's population.^[12]*

Indigenous Australians have inhabited the Sydney area for at least 30,000 years, and thousands of engravings remain throughout the region, making it one of the richest in Australia in terms of Aboriginal archaeological sites. During his first Pacific voyage in 1770, Lieutenant James Cook and his crew became the first Europeans to chart the eastern coast of Australia, making landfall at Botany Bay and inspiring British interest in the area. In 1788, the First Fleet of convicts, led by Arthur Phillip, founded Sydney as a British penal colony, the first European settlement in Australia. Phillip named the city Sydney in recognition of Thomas Townshend, 1st Viscount Sydney.^[13] Penal transportation to New South Wales ended soon after Sydney was incorporated as a city in 1842. A gold rush occurred in the colony in 1851, and over the next century, Sydney transformed from a colonial outpost into a major global cultural and economic centre. After World War II, it experienced mass migration and became one of the most multicultural cities in the world.^[3] At the time of the 2011 census, more than 250 different languages were spoken in Sydney.^[14] In the 2016 Census, about 35.8% of residents spoke a language other than English at home.^[15] Furthermore, 45.4% of the population reported having been born overseas, making Sydney the 3rd largest foreign born population of any city in the world after London and New York City, respectively.^{[16][17]}

Brief in Machine Learning!

Machine Learning

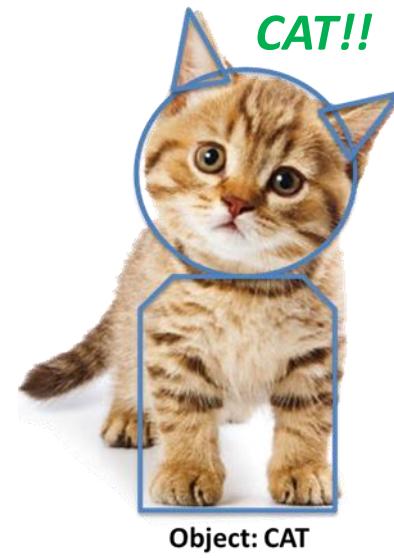
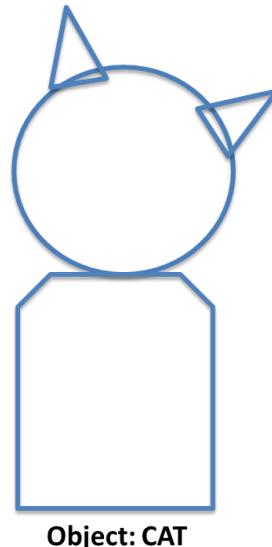
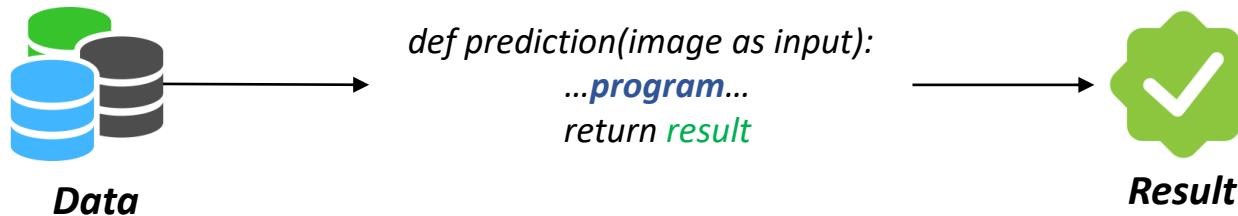
How to classify this with your machine?



Object: CAT

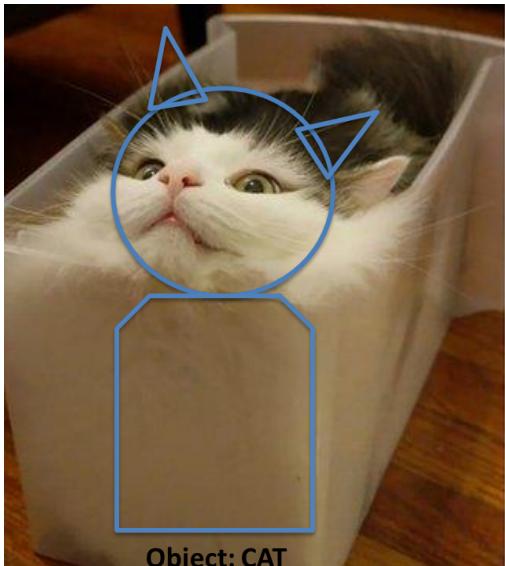
Brief in Machine Learning!

Computer System



Brief in Machine Learning!

Can we classify this with the computer system?



Object: ???



Object: ???

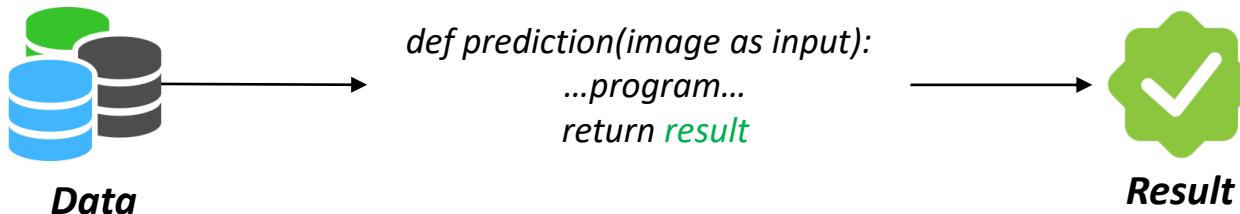


Object: ???

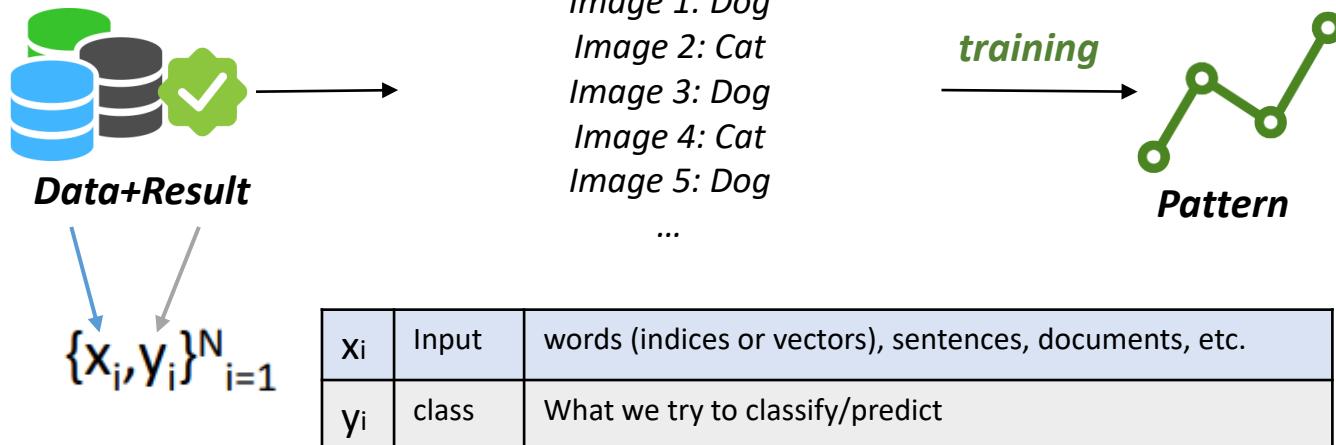
Brief in Machine Learning!

Computer System VS Machine Learning

Computer System



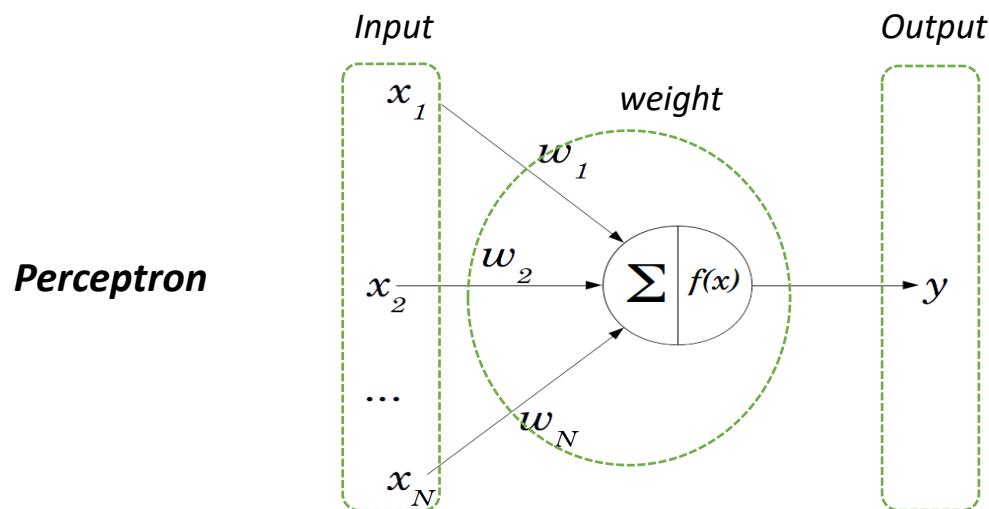
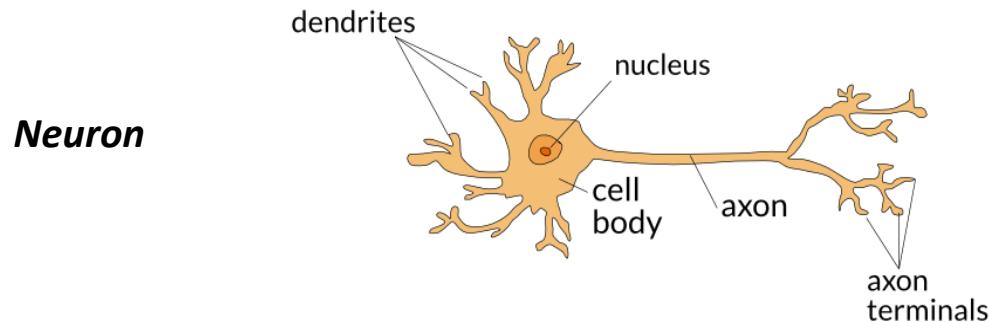
Machine Learning



Brief in Machine Learning!

Neural Network and Deep Learning

Neuron and Perceptron



NOTE: The detailed neural network and deep learning concept will be covered in the Lecture 3

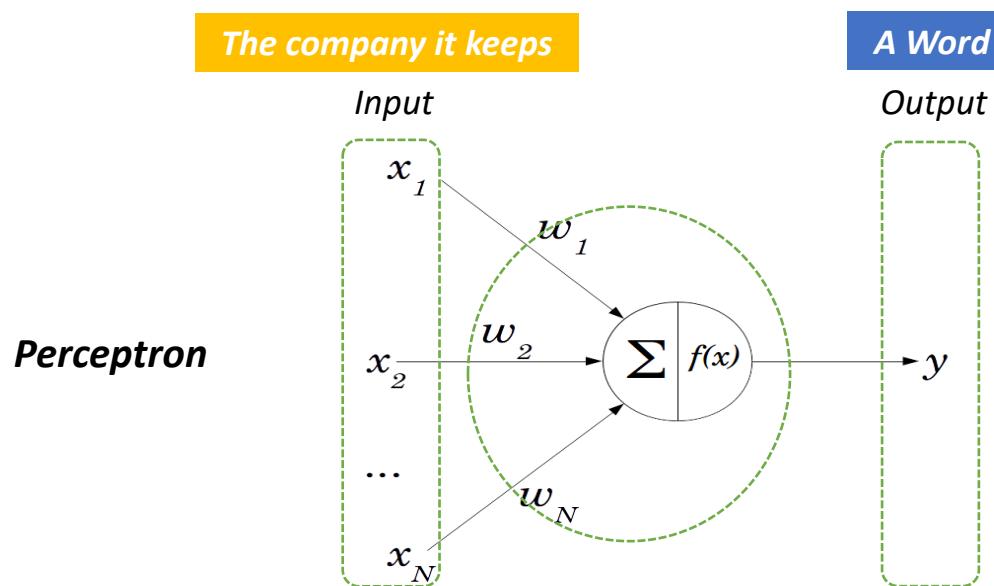
Prediction based Word representation

Neural Network and Deep Learning in Word Representation

“You shall know a word by the company it keeps” (Firth, J. R. 1957:11)

Why don’t we train a word by the company it keeps?

Why don’t we represent a word by the company it keeps?



Prediction based Word representation

Neural Network and Deep Learning in Word Representation

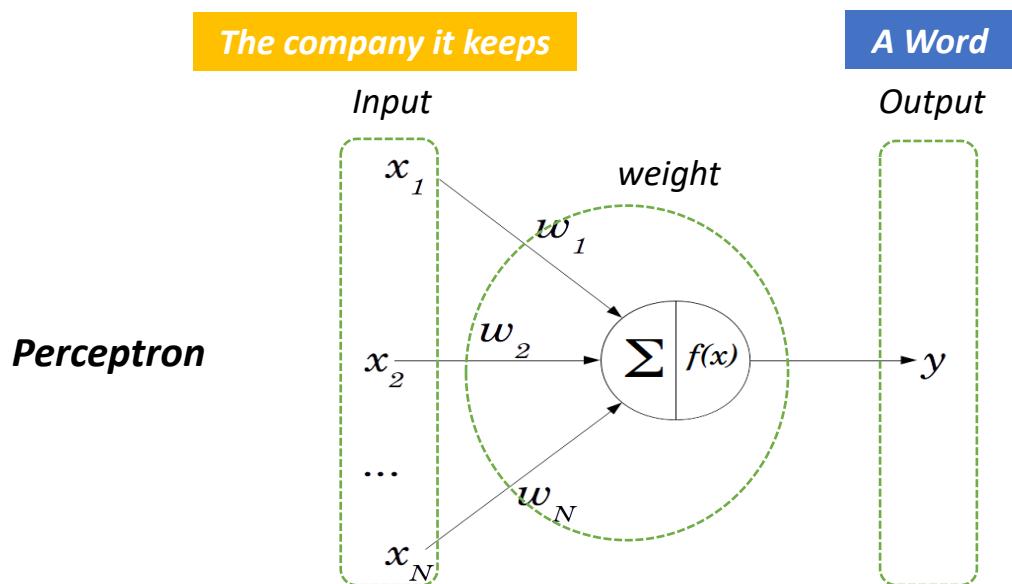
Wikipedia: “Sydney is the state capital of NSW...”

Sydney

From Wikipedia, the free encyclopedia

This article is about the Australian metropolis. For the local government area see Sydney (local government area).

Sydney (/'sɪdnɪ/ (listen) *SID-nee*)^[7] is the state capital of New South Wales



Prediction based Word representation

Neural Network and Deep Learning in Word Representation

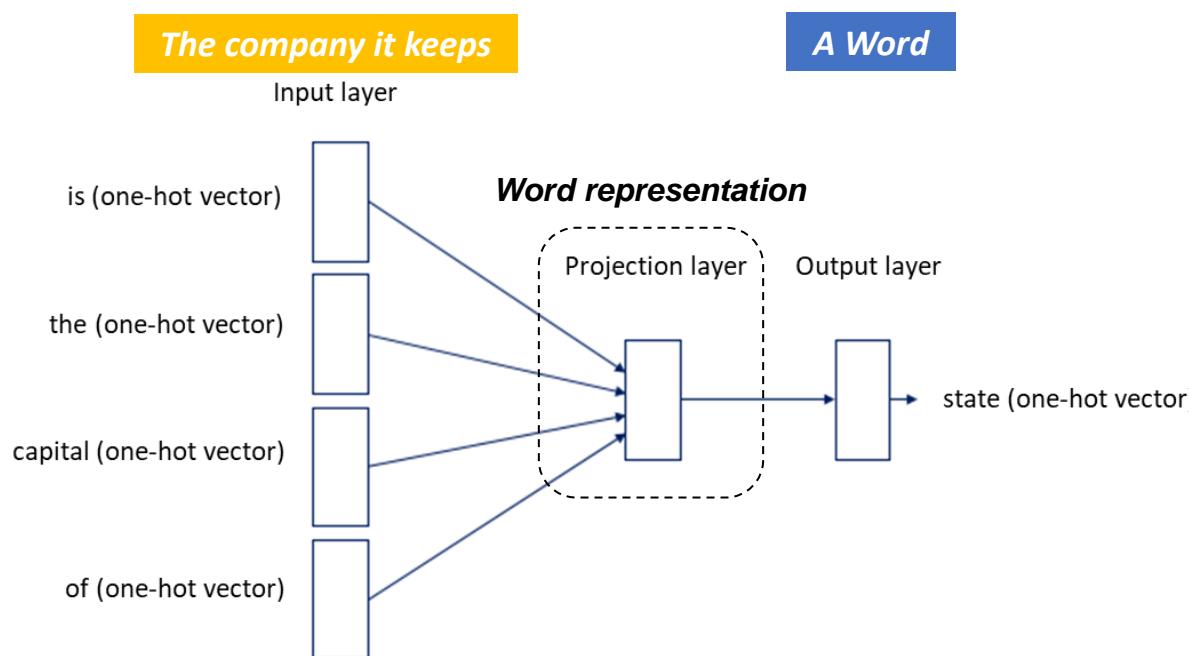
Wikipedia: “Sydney is the state capital of NSW...”

Sydney

From Wikipedia, the free encyclopedia

This article is about the Australian metropolis. For the local government area, see Sydney (local government area). For other uses, see Sydney (disambiguation).

Sydney (/'sɪdni/ (listen) *SID-nee*)^[7] is the state capital of New South Wales



3

Prediction based Word representation

Neural Network and Deep Learning in Word Representation

Wikipedia: “Sydney is the state capital of NSW...”

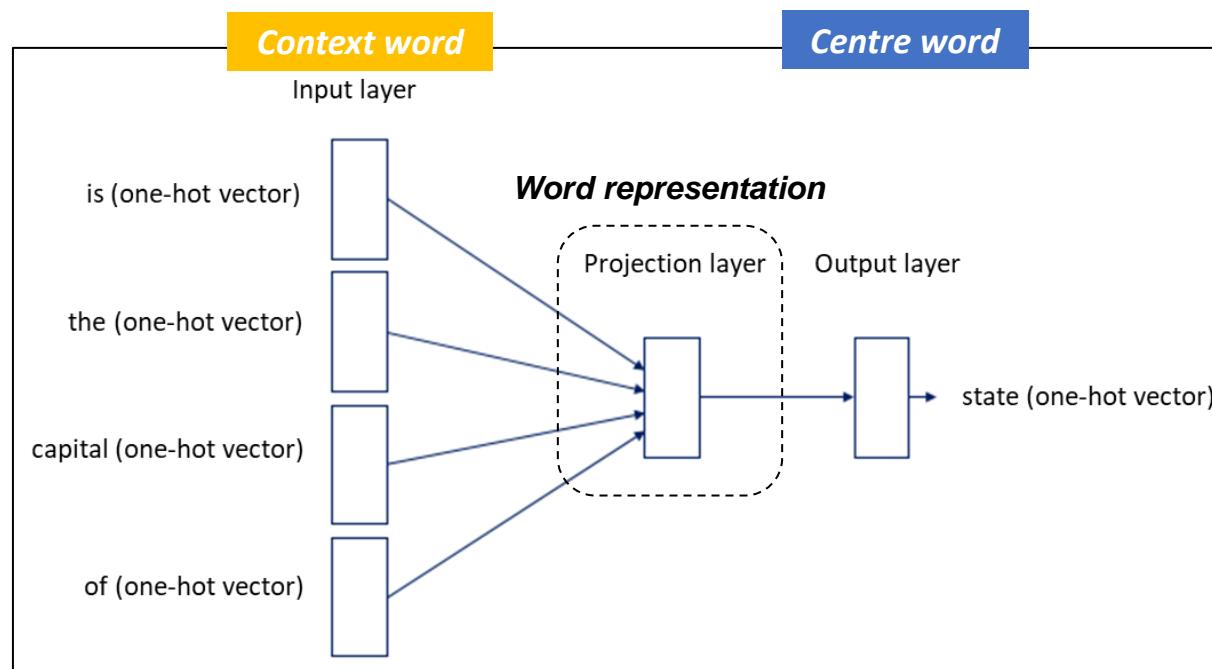
Sydney

From Wikipedia, the free encyclopedia

This article is about the Australian metropolis. For the local government area see Sydney (local government area).

Sydney (/'sɪdni/ (listen) *SID-nee*)^[7] is the state capital of New South Wales

Word2Vec



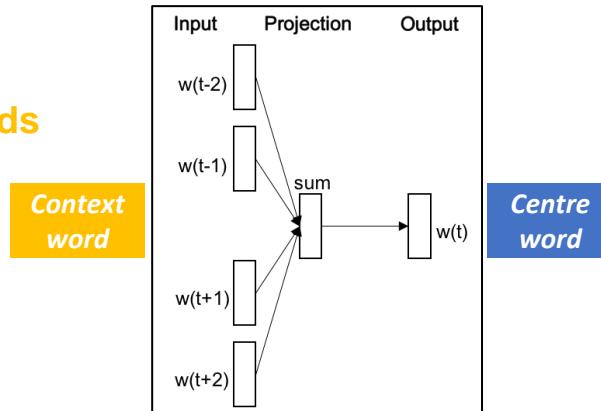
Prediction based Word representation

Word2Vec

Word2vec can utilize either of two model architectures to produce a distributed representation of words:

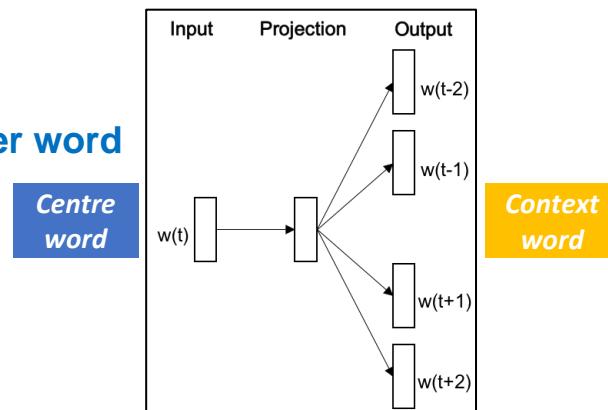
1. Continuous Bag of Words (CBOW)

Predict **center word** from (bag of) **context words**



2. Continuous Skip-gram

Predict **context (“outside”) words** given **center word**



3

Prediction based Word representation

Word2Vec with Continuous Bag of Words (CBOW)

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”

Aim

- Predict the center word

Setup

- Window size
 - Assume that the window size is 2

Sydney	is	the	state	capital	of	NSW
--------	----	-----	-------	---------	----	-----

Sydney	is	the	state	capital	of	NSW
--------	----	-----	-------	---------	----	-----

Sydney	is	the	state	capital	of	NSW
--------	----	-----	-------	---------	----	-----

Sydney	is	the	state	capital	of	NSW
--------	----	-----	-------	---------	----	-----

Sydney	is	the	state	capital	of	NSW
--------	----	-----	-------	---------	----	-----

Sydney	is	the	state	capital	of	NSW
--------	----	-----	-------	---------	----	-----

Sydney	is	the	state	capital	of	NSW
--------	----	-----	-------	---------	----	-----

 Center word

 Context (“outside”) word

3

Prediction based Word representation

Word2Vec with Continuous Bag of Words (CBOW)

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”

Using window slicing, develop the training data

Center word	Context (“outside”) word	Sydney is the state capital of NSW
[1,0,0,0,0,0,0]	[0,1,0,0,0,0,0], [0,0,1,0,0,0,0]	Sydney is the state capital of NSW
[0,1,0,0,0,0,0]	[1,0,0,0,0,0,0], [0,0,1,0,0,0,0], [0,0,0,1,0,0,0]	Sydney is the state capital of NSW
[0,0,1,0,0,0,0]	[1,0,0,0,0,0,0], [0,1,0,0,0,0,0] [0,0,0,1,0,0,0], [0,0,0,0,1,0,0]	Sydney is the state capital of NSW
[0,0,0,1,0,0,0]	[0,1,0,0,0,0,0], [0,0,1,0,0,0,0] [0,0,0,0,1,0,0], [0,0,0,0,0,1,0]	Sydney is the state capital of NSW
[0,0,0,0,1,0,0]	[0,0,1,0,0,0,0], [0,0,0,1,0,0,0] [0,0,0,0,0,1,0], [0,0,0,0,0,0,1]	Sydney is the state capital of NSW
[0,0,0,0,0,1,0]	[0,0,0,1,0,0,0], [0,0,0,0,1,0,0] [0,0,0,0,0,0,1]	Sydney is the state capital of NSW
[0,0,0,0,0,0,1]	[0,0,0,0,1,0,0], [0,0,0,0,0,1,0]	Sydney is the state capital of NSW

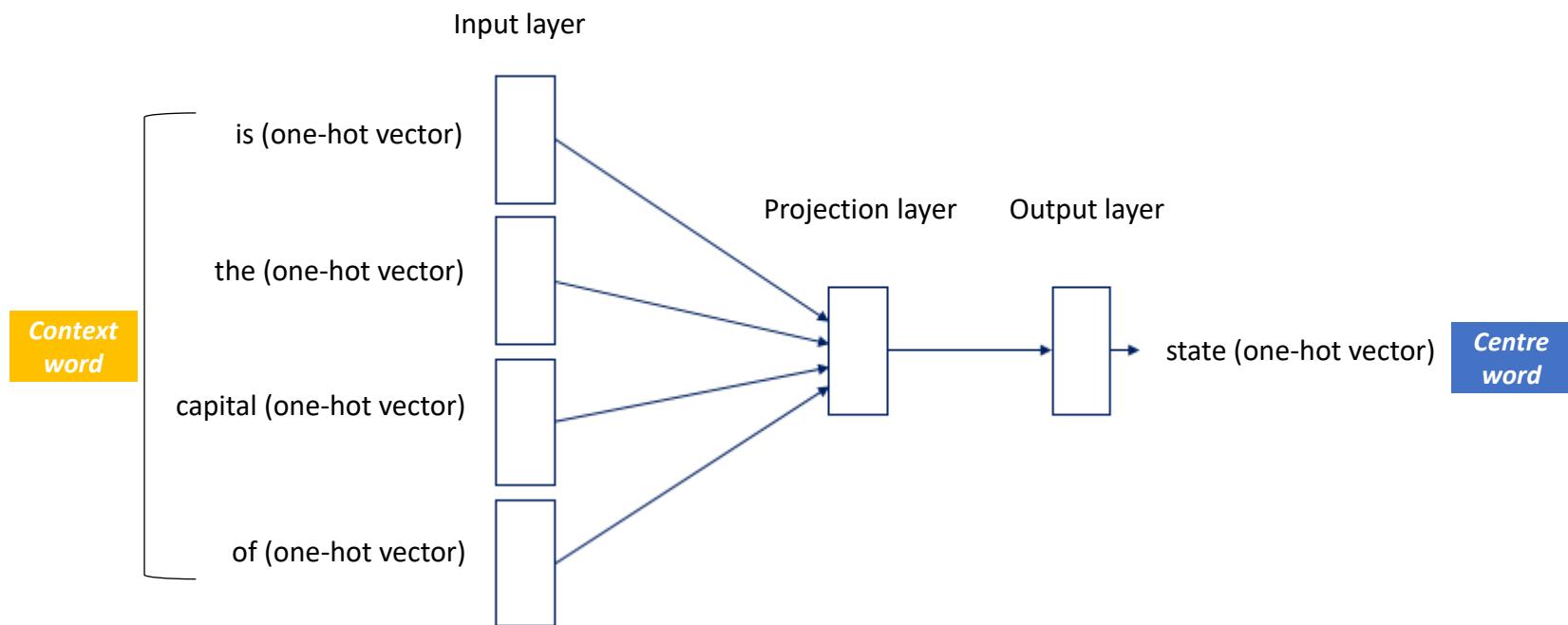
█ Center word
█ Context (“outside”) word

Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”

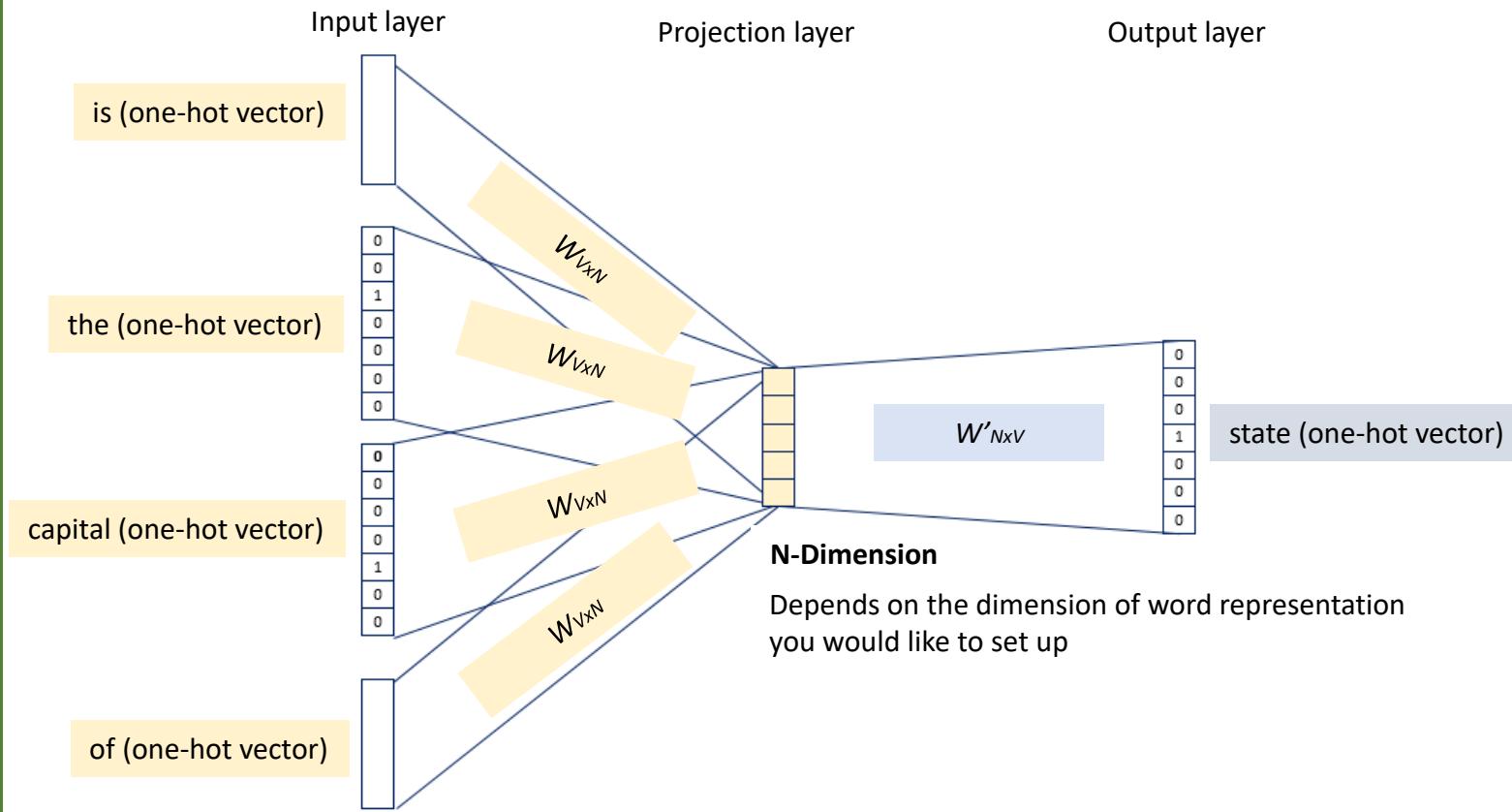


Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”



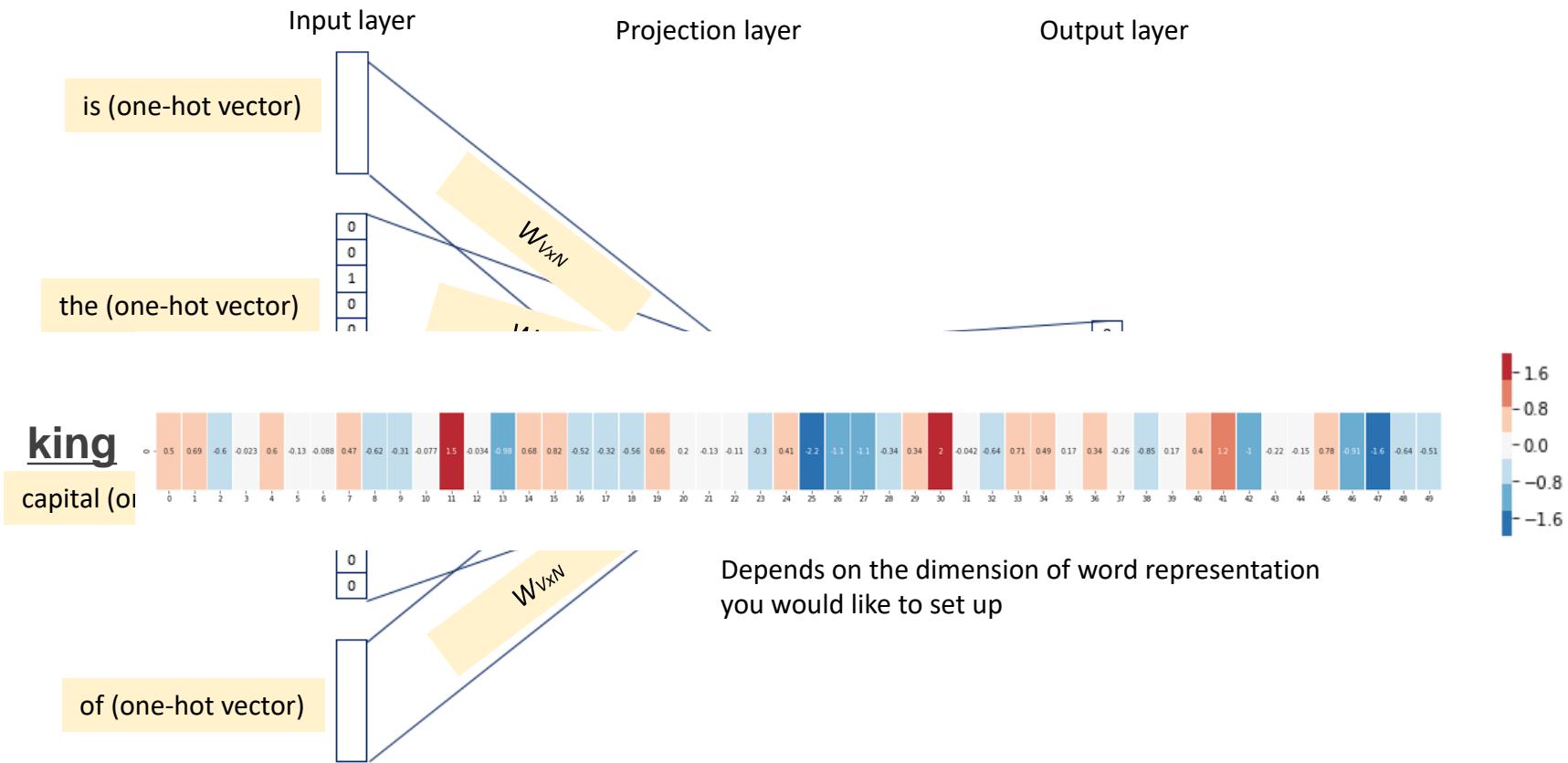
3

Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”

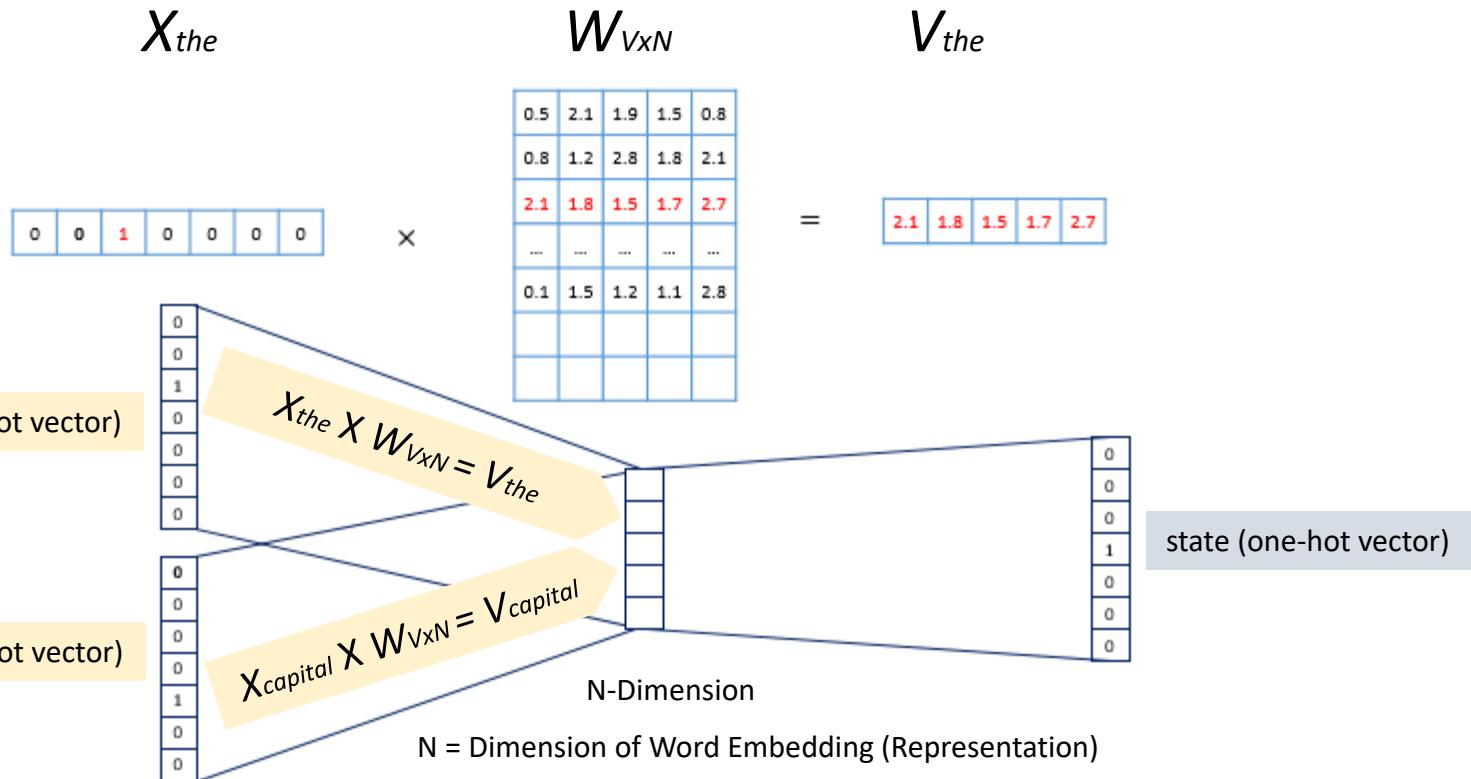


Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”

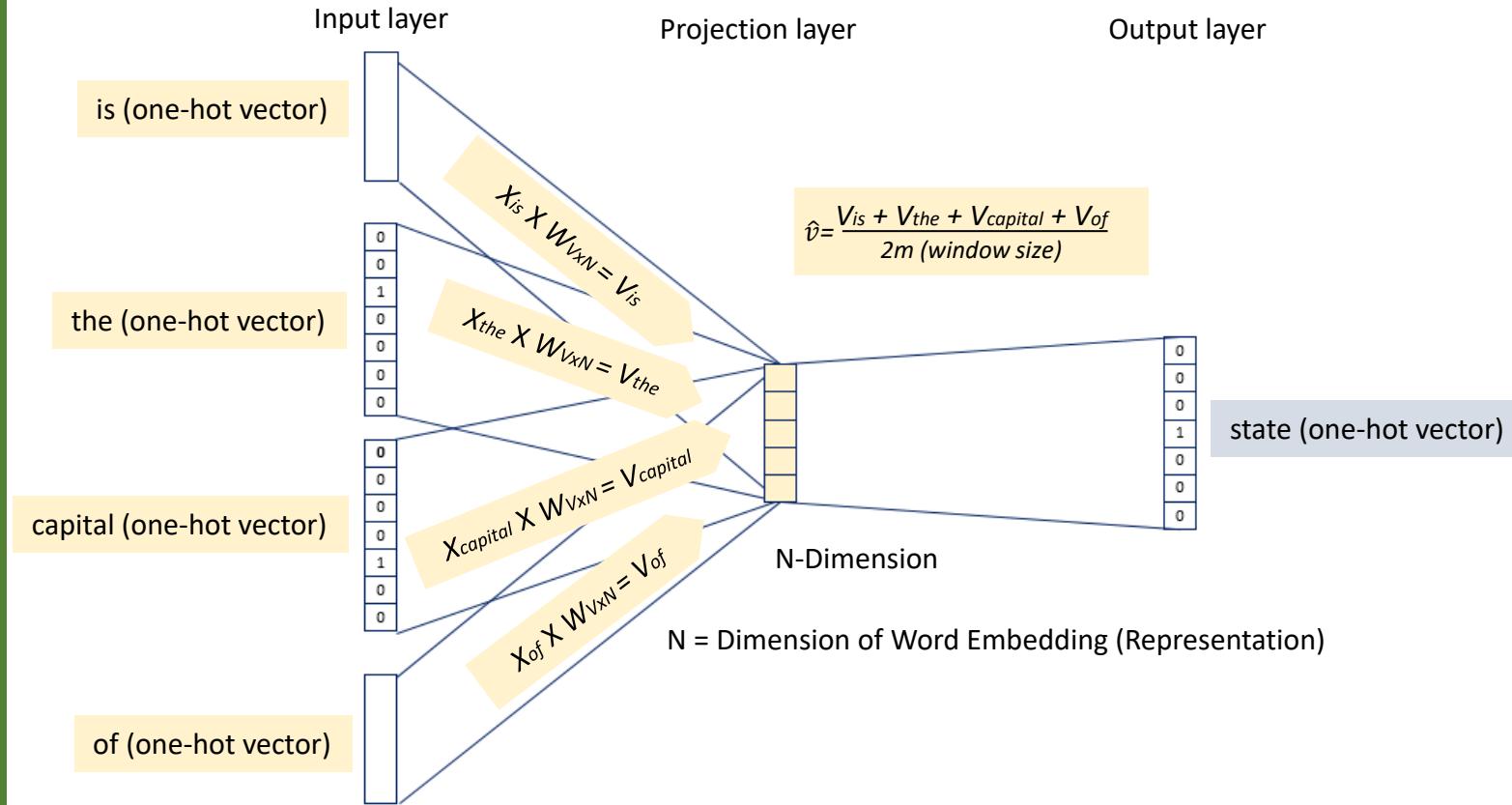


Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”

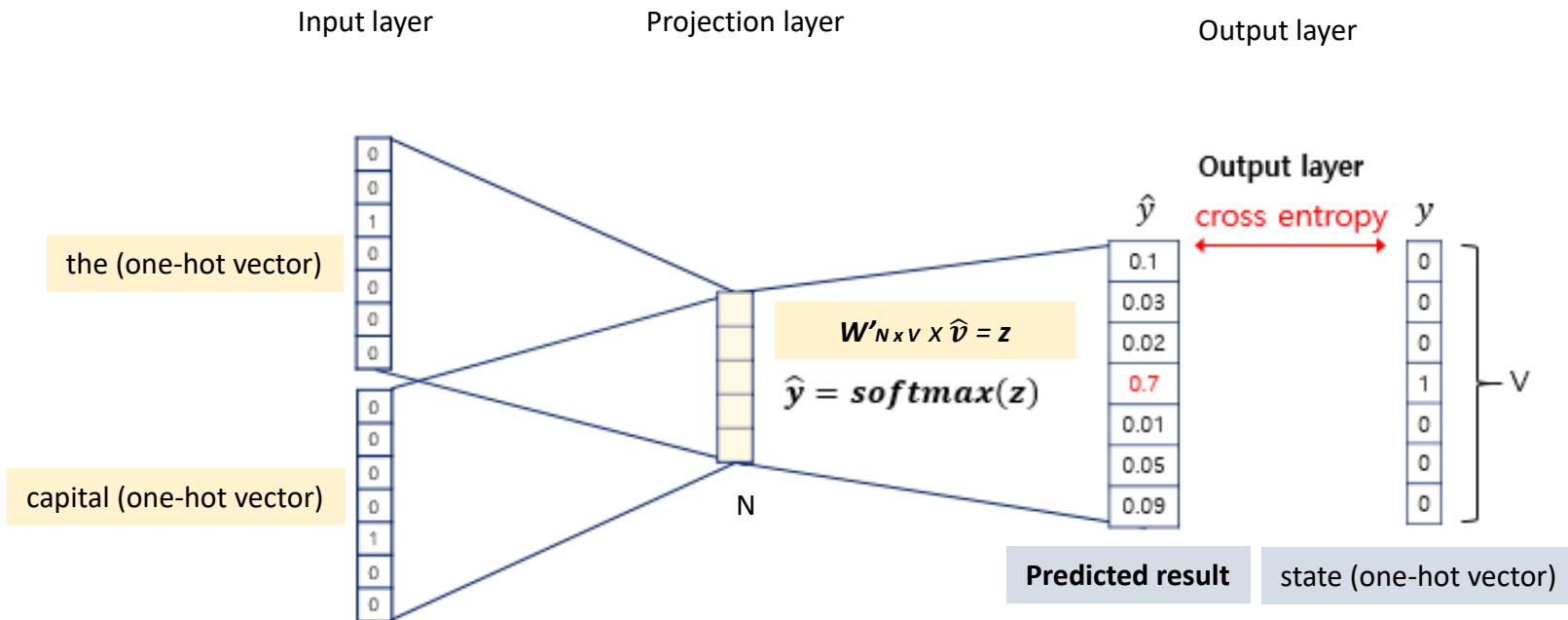


Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”



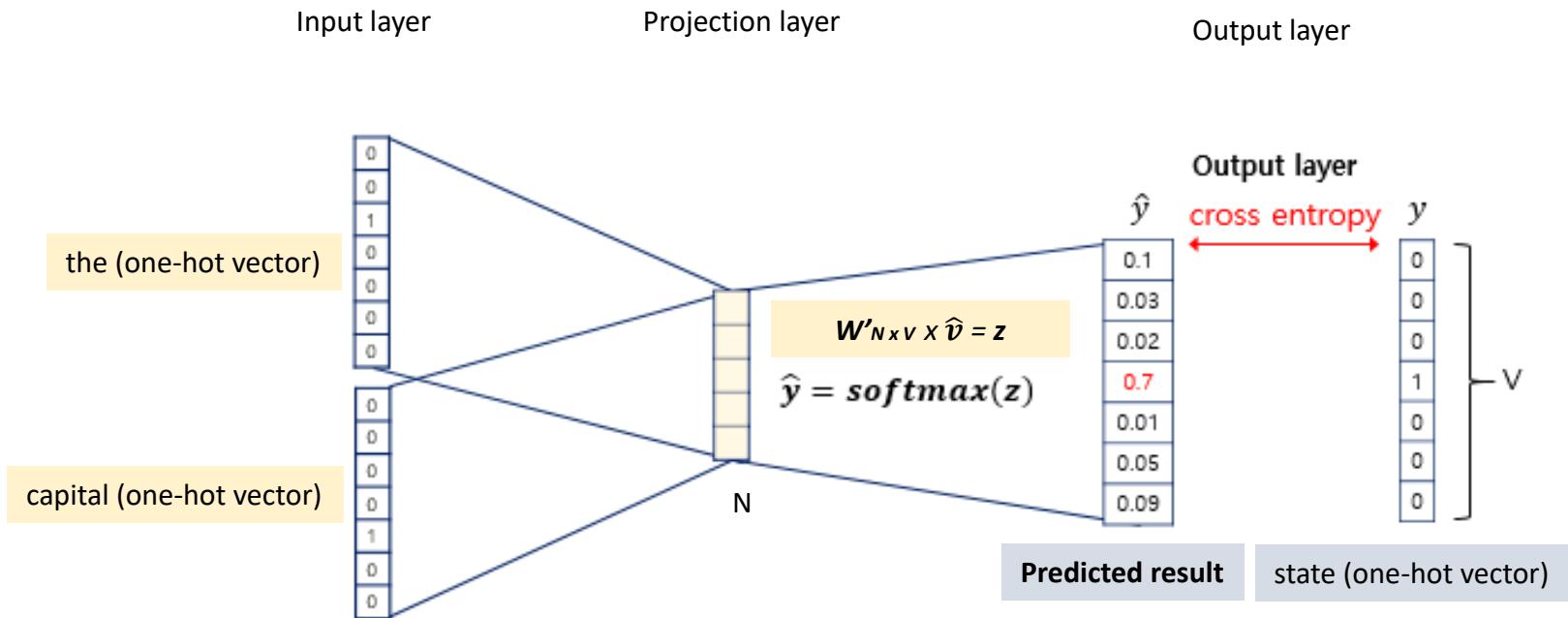
Softmax: outputs a vector that represents the probability distributions (sum to 1) of a list of potential outcome

Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”



Cross Entropy: can be used as a loss function
when optimizing classification

Loss Function (Cross Entropy)

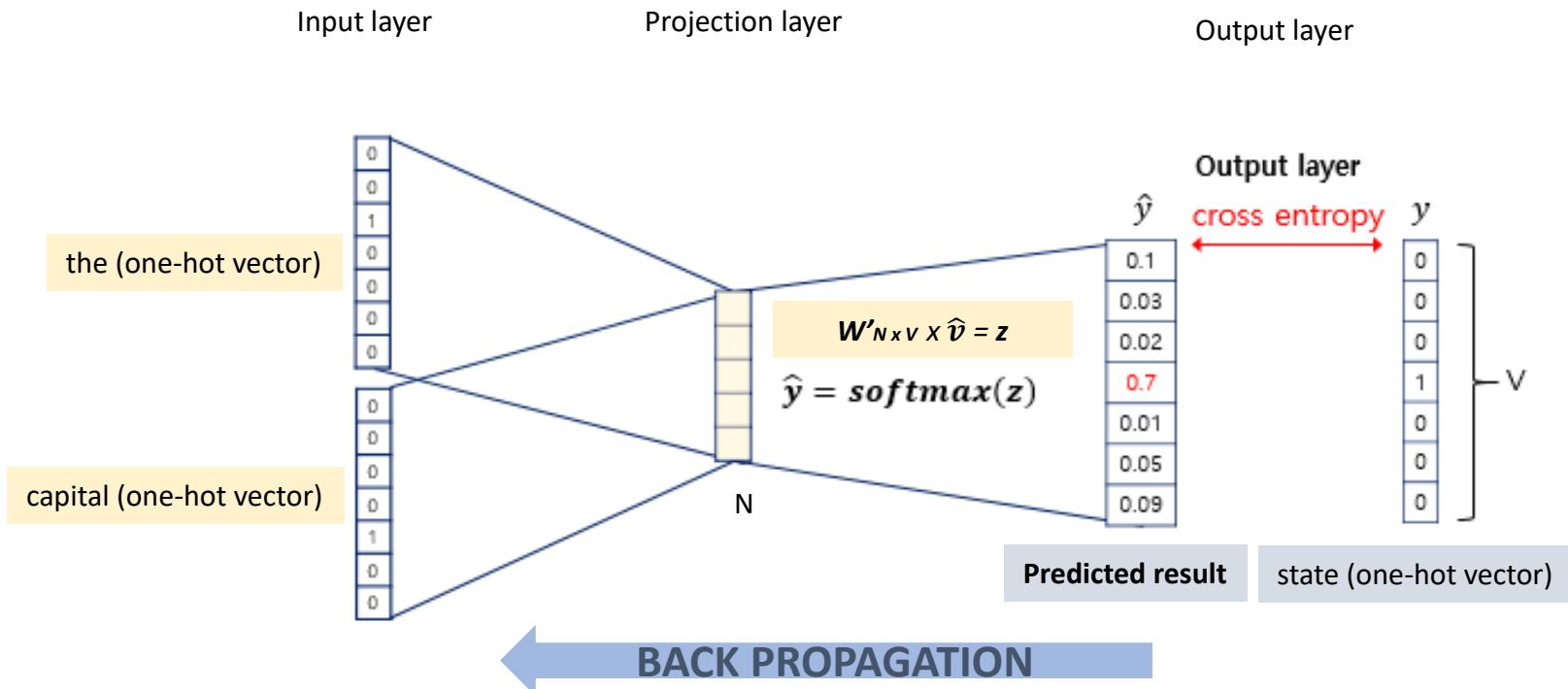
$$H(\hat{y}, y) = - \sum_{j=1}^{|V|} y_j \log(\hat{y}_j)$$

Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”



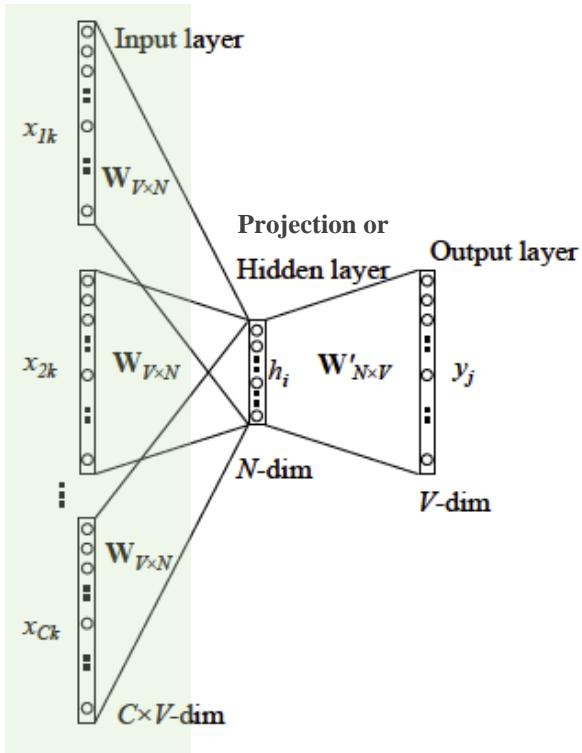
*This back propagation or optimization function will be learned more details in the lecture 3.

Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words.

Summary of CBOW Training (Review your understanding with equations)



1. Initialise each word in a one-hot vector form.

$$x_k = [0, \dots, 0, 1, 0, \dots, 0]$$
2. Use context words ($2m$, based on window size = m) as input of the Word2Vec-CBOW model.

$$(x^{c-m}, x^{c-m+1}, \dots, x^{c-1}, x^{c+1}, \dots, x^{c+m-1}, x^{c+m}) \in \mathbb{R}^{|V|}$$
3. Has two Parameter Matrices:
 - 1) Parameter Matrix (from Input Layer to Hidden/Projection Layer)

$$W \in \mathbb{R}^{V \times N}$$
 - 2) Parameter Matrix (to Output Layer)

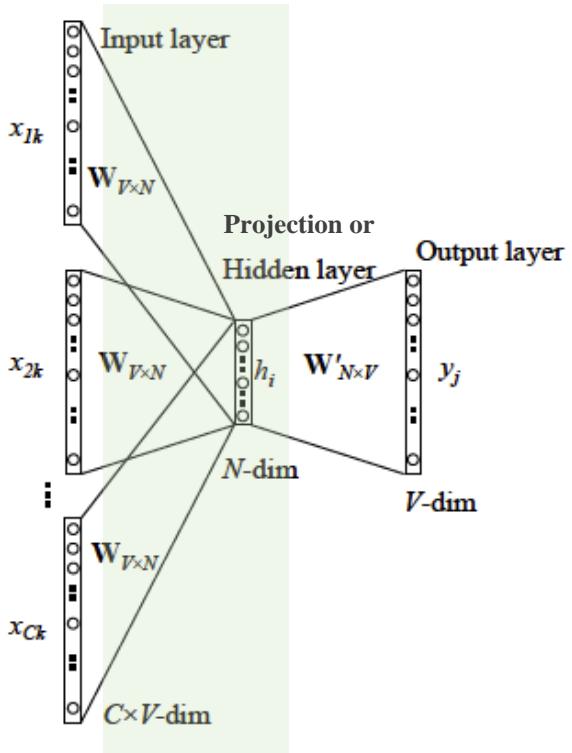
$$W' \in \mathbb{R}^{N \times V}$$

Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words.

Summary of CBOW Training (Review your understanding with equations)



4. Initial words are represented in one hot vector so multiplying a **one hot vector** with $\mathbf{W}_{V \times N}$ will give you a $1 \times N$ (embedded word) vector.

$$(\mathbf{v}_{c-m} = \mathbf{W}x^{c-m}, \dots, \mathbf{v}_{c+m} = \mathbf{W}x^{c+m}) \in \mathbb{R}^n$$

5. Average those **$2m$** embedded vectors to calculate the value of the Hidden Layer.

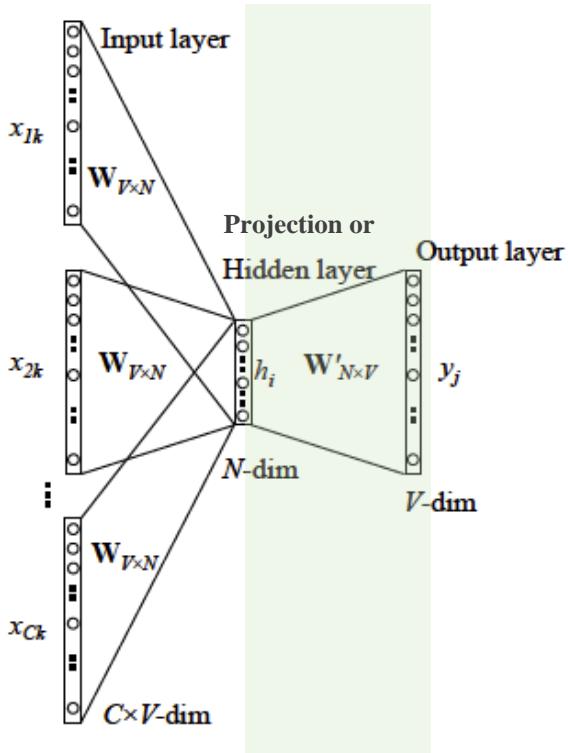
$$\hat{v} = \frac{\mathbf{v}_{c-m} + \mathbf{v}_{c-m+1} + \dots + \mathbf{v}_{c+m}}{2m}$$

Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words.

Summary of CBOW Training (Review your understanding with equations)



6. Calculate the score value for the output layer. The higher score is produced when words are closer.

$$\mathbf{z} = \hat{\mathbf{v}} \mathbf{W}' \in \mathbb{R}^{|V|}$$

7. Calculate the probability using softmax

$$\hat{y} = \text{softmax}(\mathbf{z}) \in \mathbb{R}^{|V|}$$

8. Train the parameter matrix using objective function.

$$H(\hat{y}, y) = - \sum_{j=1}^{|V|} y_j \log(\hat{y}_j)$$

* Focus on minimising the value

We use an one-hot vector (one 1, the rest 0) so it will be calculated in only one.

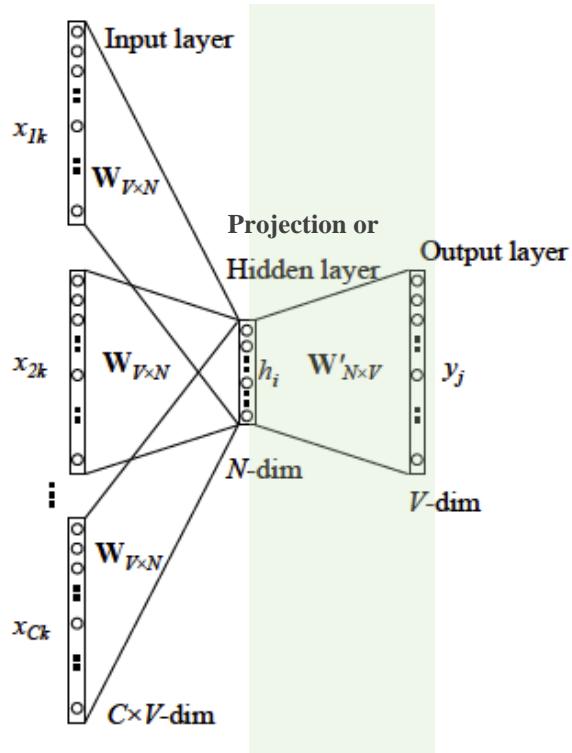
$$H(\hat{y}, y) = -y_j \log(\hat{y}_j)$$

Prediction based Word representation

CBOW – Neural Network Architecture

Predict center word from (bag of) context words.

Summary of CBOW Training (Review your understanding with equations)



8-1. Optimization Objective Function can be presented:

$$\begin{aligned}
 \text{minimize } J &= -\log P(w_c | w_{c-m}, \dots, w_{c+m}) \\
 &= -\log P(u_c | v) \\
 &= -\log \frac{\exp(u_c^\top \hat{v})}{\sum_{j=1}^{|V|} \exp(u_j^\top \hat{v})} \\
 &= -u_c^\top \hat{v} + \log \sum_{j=1}^{|V|} \exp(u_j^\top \hat{v})
 \end{aligned}$$

*This optimization objective will be learned more details in the lecture 3.

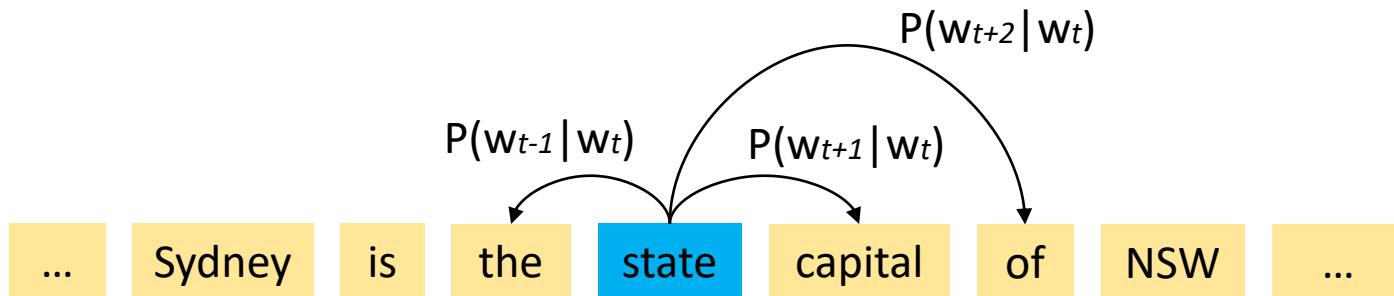
3

Prediction based Word representation

Skip Gram

Predict context (“outside”) words (position independent) given center word

Sentence: “Sydney is the state capital of NSW”

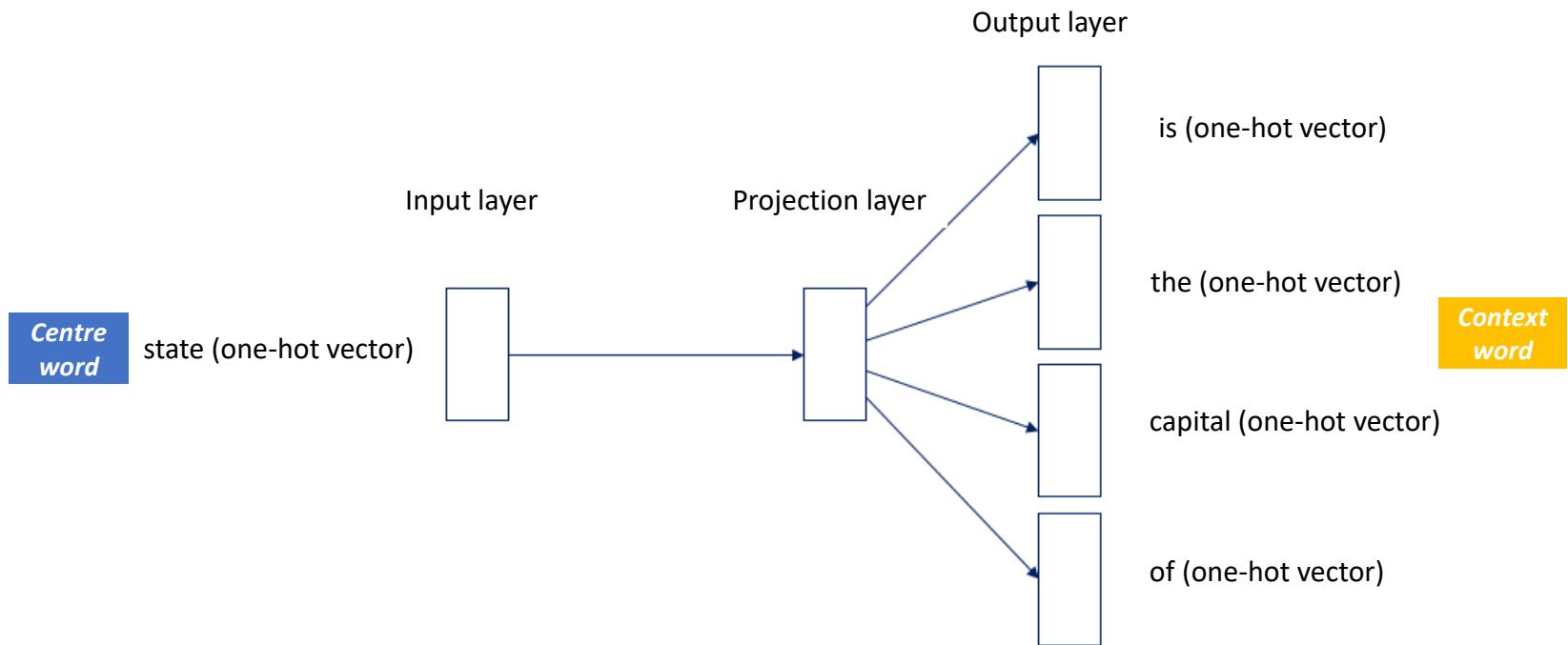


Prediction based Word representation

Skip Gram

Predict context (“outside”) words (position independent) given center word

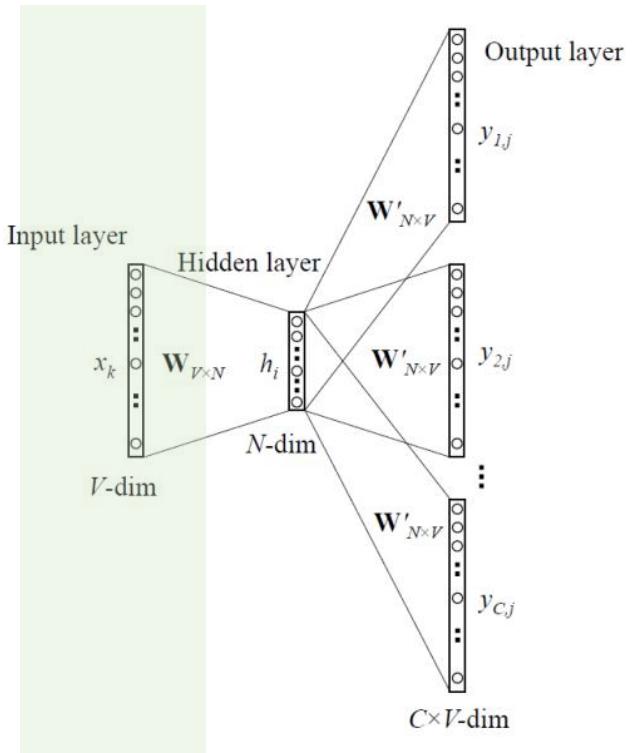
Sentence: “Sydney is the state capital of NSW”



Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given center word

Summary of Skip Gram Training (Review your understanding with equations)



1. Initialise the centre word in a one-hot vector form.

$$\mathbf{x}_k = [0, \dots, 0, 1, 0, \dots, 0]$$

$$\mathbf{x} \in \mathbb{R}^{|V|}$$

2. Has two Parameter Matrices:

- 1) Parameter Matrix (from Input Layer to Hidden/Projection Layer)
 $\mathbf{W} \in \mathbb{R}^{V \times N}$

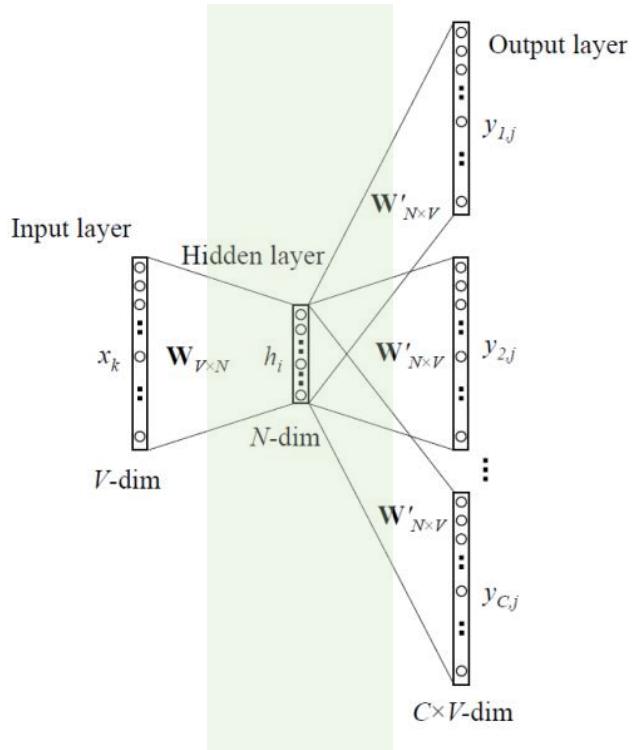
2) Parameter Matrix (to Output Layer)

$$\mathbf{W}' \in \mathbb{R}^{N \times V}$$

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given center word

Summary of Skip Gram Training (Review your understanding with equations)



3. Initial words are represented in one hot vector so multiplying a **one hot vector** with $\mathbf{W}_{V \times N}$ will give you a $1 \times N$ (embedded word) vector.

$$\mathbf{v}_c = \mathbf{W}_x \in \mathbb{R}^n \text{ (as there is only one input)}$$

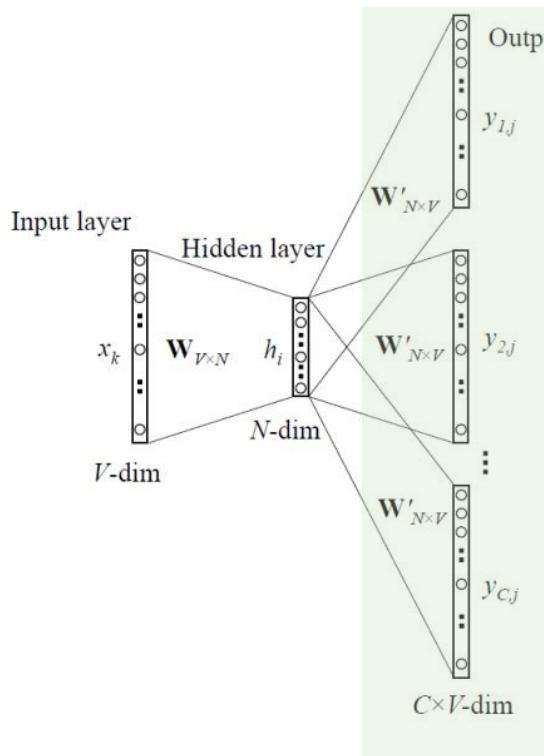
4. Calculate the score value for the output layer by multiplying the parameter matrix \mathbf{W}'
 $\mathbf{z} = \mathbf{W}' \mathbf{v}_c$

Prediction based Word representation

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given center word

Summary of Skip Gram Training (Review your understanding with equations)



5. Calculate the probability using softmax
 $\hat{y} = \text{softmax}(\mathbf{z})$

6. Calculate $2m$ probabilities as we need to predict $2m$ context words.

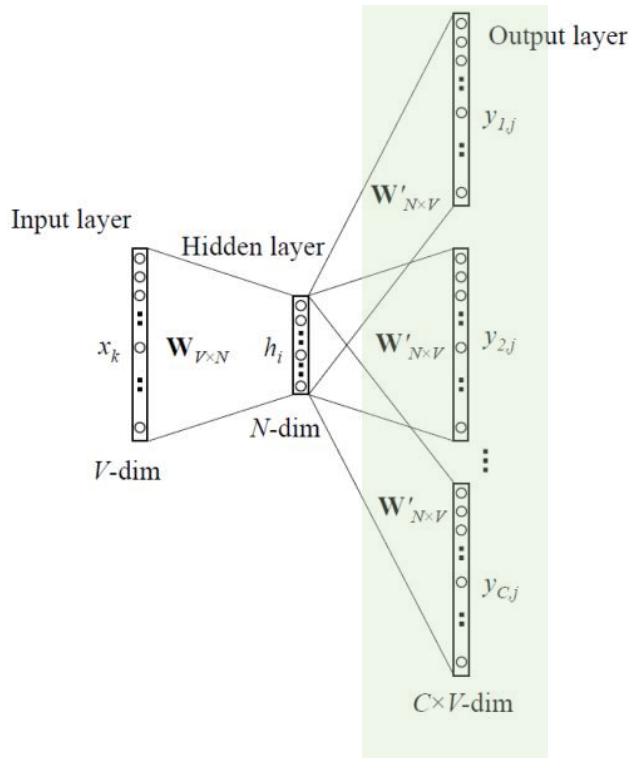
$$\hat{y}_{c-m}, \dots, \hat{y}_{c-1}, \hat{y}_{c+1}, \dots, \hat{y}_{c+m}$$

and compare with the ground truth (one-hot vector)
 $y^{(c-m)}, \dots, y^{(c-1)}, y^{(c+1)}, \dots, y^{(c+m)}$

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given center word

Summary of Skip Gram Training (Review your understanding with equations)



8. As in CBOW, use an objective function for us to evaluate the model. A key difference here is that we invoke a Naïve Bayes assumption to break out the probabilities. It is a strong naïve conditional independence assumption. Given the centre word, all output words are completely independent.

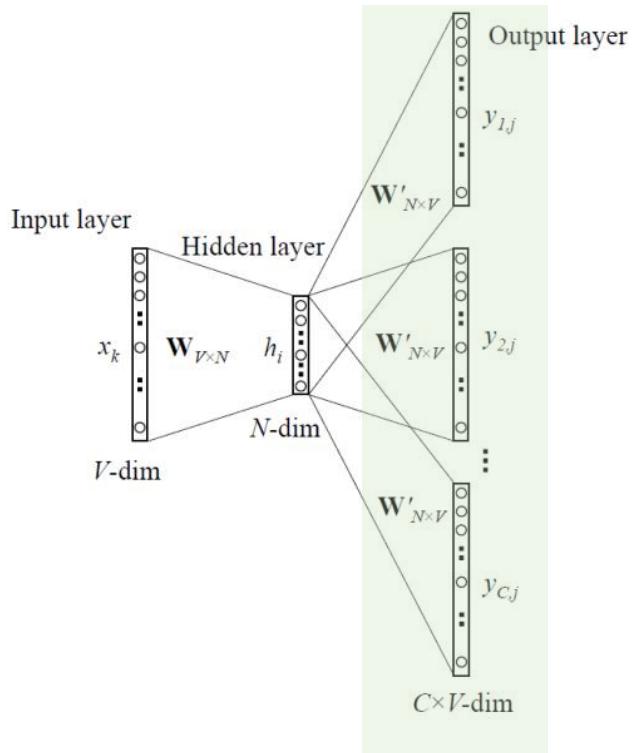
$$\begin{aligned}
 \text{minimize } J &= -\log P(w_{c-m}, \dots, w_{c-1}, w_{c+1}, \dots, w_{c+m} | w_c) \\
 &= -\log \prod_{j=0, j \neq m}^{2m} P(w_{c-m+j} | w_c) \\
 &= -\log \prod_{j=0, j \neq m}^{2m} \frac{\exp(u_{c-m+j}^\top v_c)}{\sum_{k=1}^{|V|} \exp(u_k^\top v_c)} \\
 &= - \sum_{j=0, j \neq m}^{2m} u_{c-m+j}^\top v_c + 2m \log \sum_{k=1}^{|V|} \exp(u_k^\top v_c)
 \end{aligned}$$

*This optimization objective will be learned more details in the lecture 3.

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given center word

Summary of Skip Gram Training (Review your understanding with equations)



8-1. With this objective function, we can compute the gradients with respect to the unknown parameters and at each iteration update them via Stochastic Gradient Descent

$$\begin{aligned}
 J &= - \sum_{j=0, j \neq m}^{2m} \log P(u_{c-m+j} | v_c) \\
 &= \sum_{j=0, j \neq m}^{2m} H(\hat{y}, y_{c-m+j})
 \end{aligned}$$

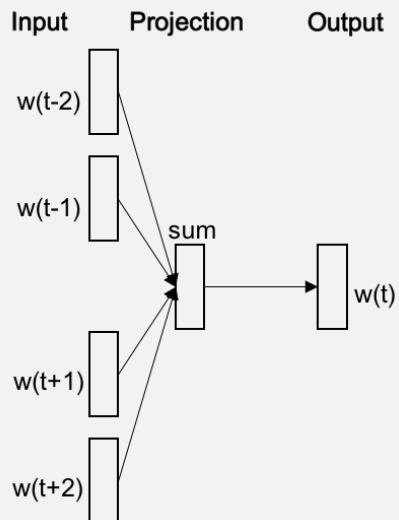
*This Stochastic Gradient Descent will be learned details in the lecture 3.

Prediction based Word representation

CBOW vs Skip Gram Overview

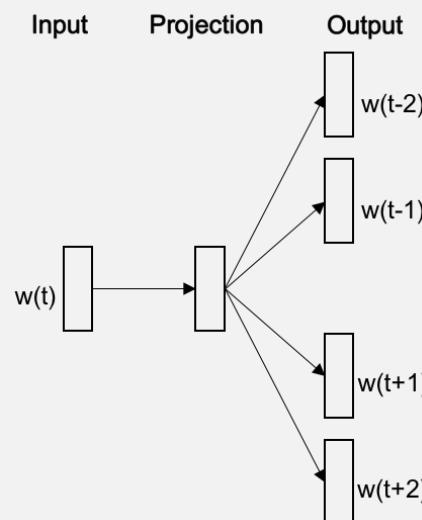
CBOW

Predict center word from (bag of) context words



Skip-gram

Predict context words given center word



3

Prediction based Word representation

Key Parameter (1) for Training methods: Window Size

Different tasks are served better by different window sizes.

Smaller window sizes (2-15) lead to embeddings where high similarity scores between two embeddings indicates that the words are interchangeable.

Larger window sizes (15-50, or even more) lead to embeddings where similarity is more indicative of relatedness of the words

Prediction based Word representation

Key Parameter (2) for Training methods: Negative Samples

The number of negative samples is another factor of the training process.

Negative samples to our dataset – samples of words that are not neighbors

Negative sample: 2

<i>Input word</i>	<i>Output word</i>	<i>Target</i>
eat	mango	1
eat	exam	0
eat	tobacco	0

*1=Appeared, 0=Not Appeared

Negative sample: 5

<i>Input word</i>	<i>Output word</i>	<i>Target</i>
eat	mango	1
eat	exam	0
eat	tobacco	0
eat	pool	0
eat	supervisor	0
eat	building	0

The original paper prescribes **5-20** as being a good number of negative samples. It also states that **2-5** seems to be enough when you have a large enough dataset.

Prediction based Word representation

Key Parameter (2) for Training methods: Negative Samples

The number of negative samples is another factor of the training process.

Negative samples to our dataset – samples of words that are not neighbors

Negative sample: 2

<i>Input word</i>	<i>Output word</i>	<i>Target</i>
eat	mango	1
eat	exam	0
eat	tobacco	0

*1=Appeared, 0=Not Appeared

Negative sample: 5

<i>Input word</i>	<i>Output word</i>	<i>Target</i>
eat	mango	1
eat	exam	0
eat	tobacco	0
eat	pool	0
eat	supervisor	0

How to select the Negative Sample?

The “negative samples” are selected using a “unigram distribution”, where more frequent words are more likely to be selected as negative samples.

$$P(w_i) = \frac{f(w_i)}{\sum_{j=0}^n(f(w_j))}$$

The probability for picking the word (w_i) would be equal to the number of times (w_i) appears in the corpus, divided the total number of word occurs in the corpus.

Prediction based Word representation

Word2Vec Overview

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

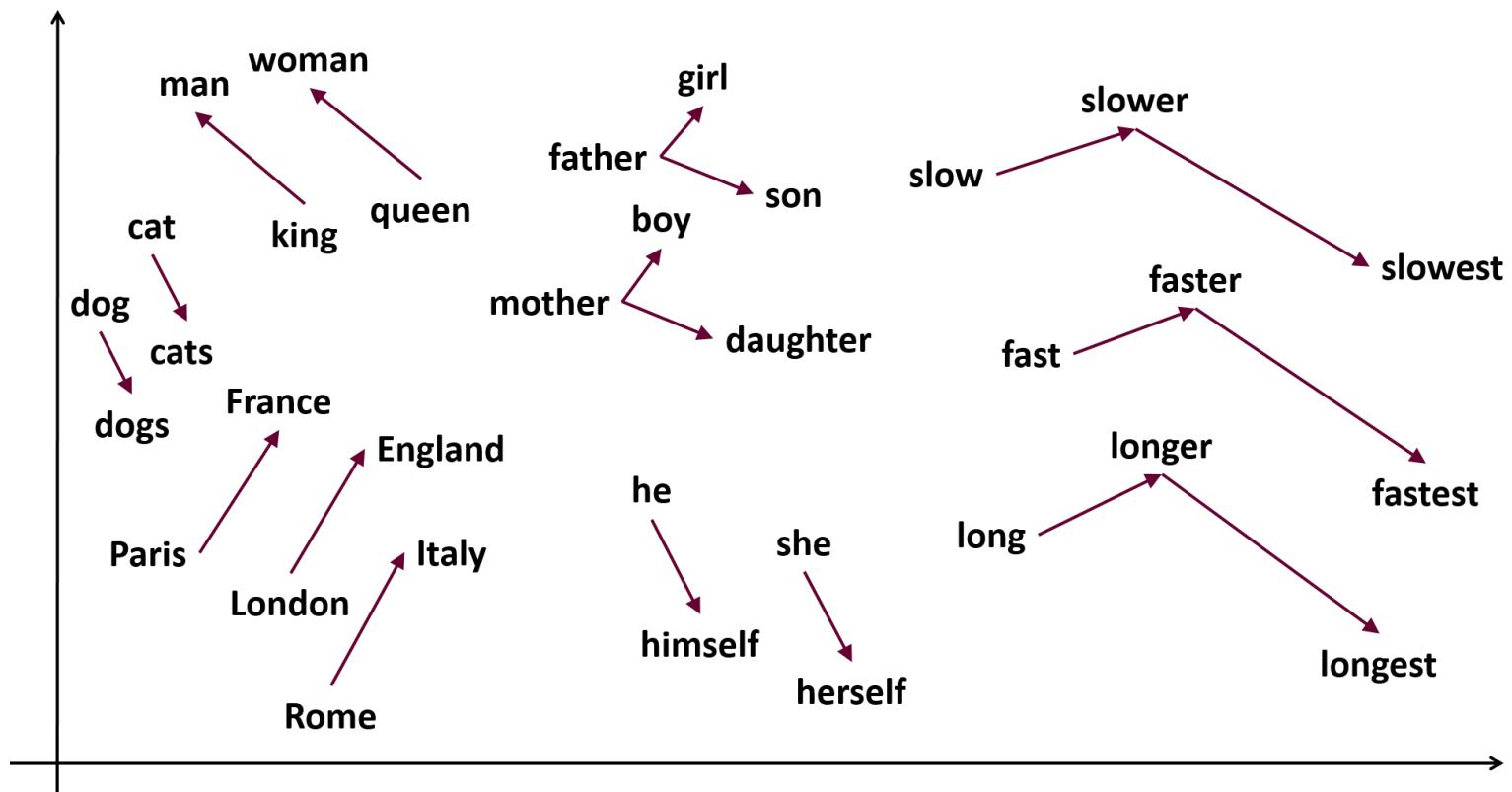
Idea:

- 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 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

3

Prediction based Word representation

Let's try some Word2Vec!



Gensim: <https://radimrehurek.com/gensim/models/word2vec.html>

Resources: <https://wit3.fbk.eu/>

<https://github.com/3Top/word2vec-api#where-to-get-a-pretrained-models>

Prediction based Word representation

Limitation of Word2Vec

Issue#1: Cannot cover the morphological similarity

- Word2vec represents every word as an independent vector, even though many words are morphologically similar, like: teach, teacher, teaching

Issue#2: Hard to conduct embedding for rare words

- Word2vec is based on the Distribution hypothesis. Works well with the frequent words but does not embed the rare words.
(same concept with the under-fitting in machine learning)

Issue#3: Cannot handle the Out-of-Vocabulary (OOV)

- Word2vec does not work at all if the word is not included in the Vocabulary

Prediction based Word representation

FastText

- Deal with this Word2Vec Limitation
- Another Way to transfer *WORDS* to *VECTORS*

fastText

- FastText is a library for learning of word embeddings and text classification created by Facebook's AI Research lab. The model allows to create an unsupervised learning or supervised learning algorithm for obtaining vector representations for words.
- Extension to Word2Vec
 - Instead of feeding individual words into the Neural Network, FastText breaks words into several n-grams (sub-words)

Prediction based Word representation

FastText with N-gram Embeddings

- N-grams are simply all combinations of adjacent words or letters of length n that you can find in your source text. For example, given the word *apple*, all 2-grams (or “bigrams”) are *ap*, *pp*, *pl*, and *le*
- The tri-grams (n=3) for the word *apple* is *app*, *ppl*, and *ple* (ignoring the starting and ending of boundaries of words). The word embedding vector for *apple* will be the sum of all these n-grams.



- After training the Neural Network (either with skip-gram or CBOW), we will have word embeddings for all the n-grams given the training dataset.
- Rare words can now be properly represented since it is highly likely that some of their n-grams also appears in other words.

Prediction based Word representation

Word2Vec VS FastText

Find synonym with Word2vec

```
from gensim.models import Word2Vec
cbow_model = Word2Vec(sentences=result, size=100, window=5, min_count=5, workers=4, sg=0)

a=cbow_model.wv.most_similar("electrofishing")
pprint.pprint(a)
```

Find synonym with FastText

```
from gensim.models import FastText
FT_model = FastText(sentences=result, size=100, window=5, min_count=5, workers=4, sg=0)

a=FT_model.wv.most_similar("electrofishing")
pprint.pprint(a)
```



electrofishing

<https://fasttext.cc/>

Prediction based Word representation

Global Vectors (GloVe)

- Deal with this Word2Vec Limitation

*“Methods like skip-gram may do better on the analogy task, but they poorly utilize the statistics of the corpus since they train on separate local context windows instead of on **global co-occurrence counts**.”*

(PeddingLon et al., 2014)

- Focus on the Co-occurrence

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k ice)/P(k steam)$	8.9	8.5×10^{-2}	1.36	0.96

e.g. $P(k | i)$ $k=$ context words, $i=$ centre words

3

Prediction based Word representation

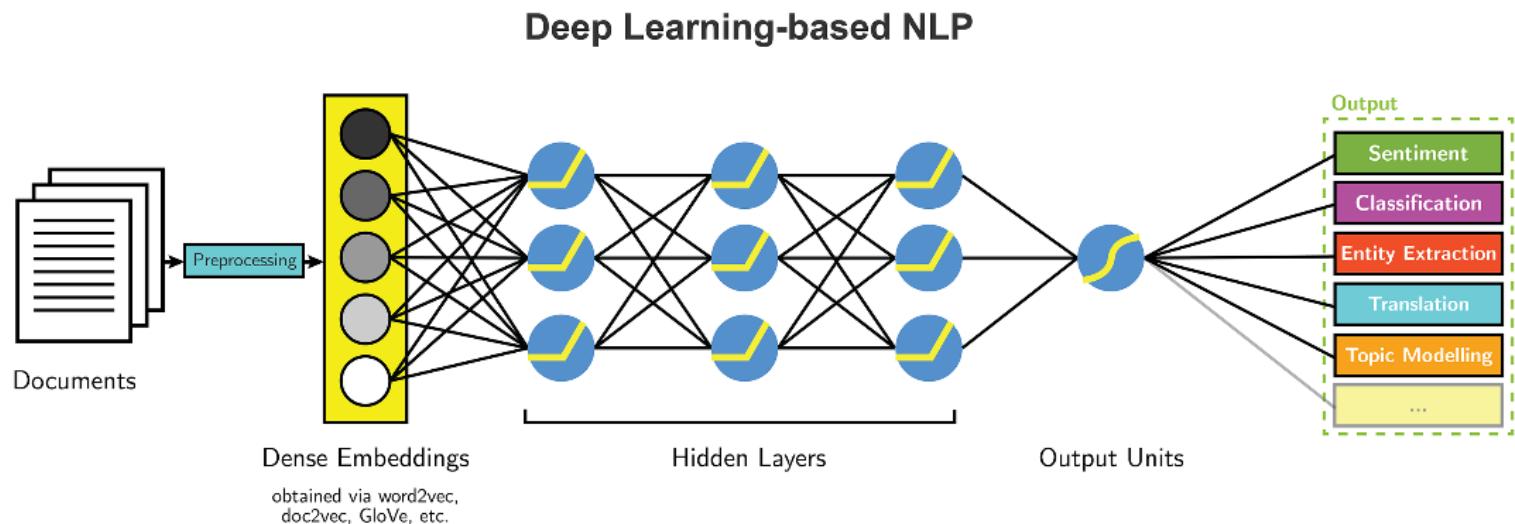
Limitation of Prediction based Word Representation

- I like _____
apple banana fruit
- Training dataset reflect the word representation result
 - The word similarity of the word ‘software’ the model learned by Google News corpus can be different from the one from Twitter.

Word Embeddings

- Finalisation!

Machine Learning/ Deep Learning for Natural Language Processing



/ Reference

Reference for this lecture

- Deng, L., & Liu, Y. (Eds.). (2018). Deep Learning in Natural Language Processing. Springer.
- Rao, D., & McMahan, B. (2019). Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. " O'Reilly Media, Inc.".
- Manning, C. D., Manning, C. D., & Schütze, H. (1999). Foundations of statistical natural language processing. MIT press.
- Manning, C 2017, Introduction and Word Vectors, Natural Language Processing with Deep Learning, lecture notes, Stanford University
- Images: <http://jalammar.github.io/illustrated-word2vec/>
- Goldberg, Lewis R. 1992, "The development of markers for the Big-Five factor structure." Psychological assessment 4.1: 26.

Word2vec

- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

FastText

- Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5, 135-146.
- Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2017). Advances in pre-training distributed word representations. arXiv preprint arXiv:1712.09405.

The lecture will be started at 5:10PM sharply!

COMP5046

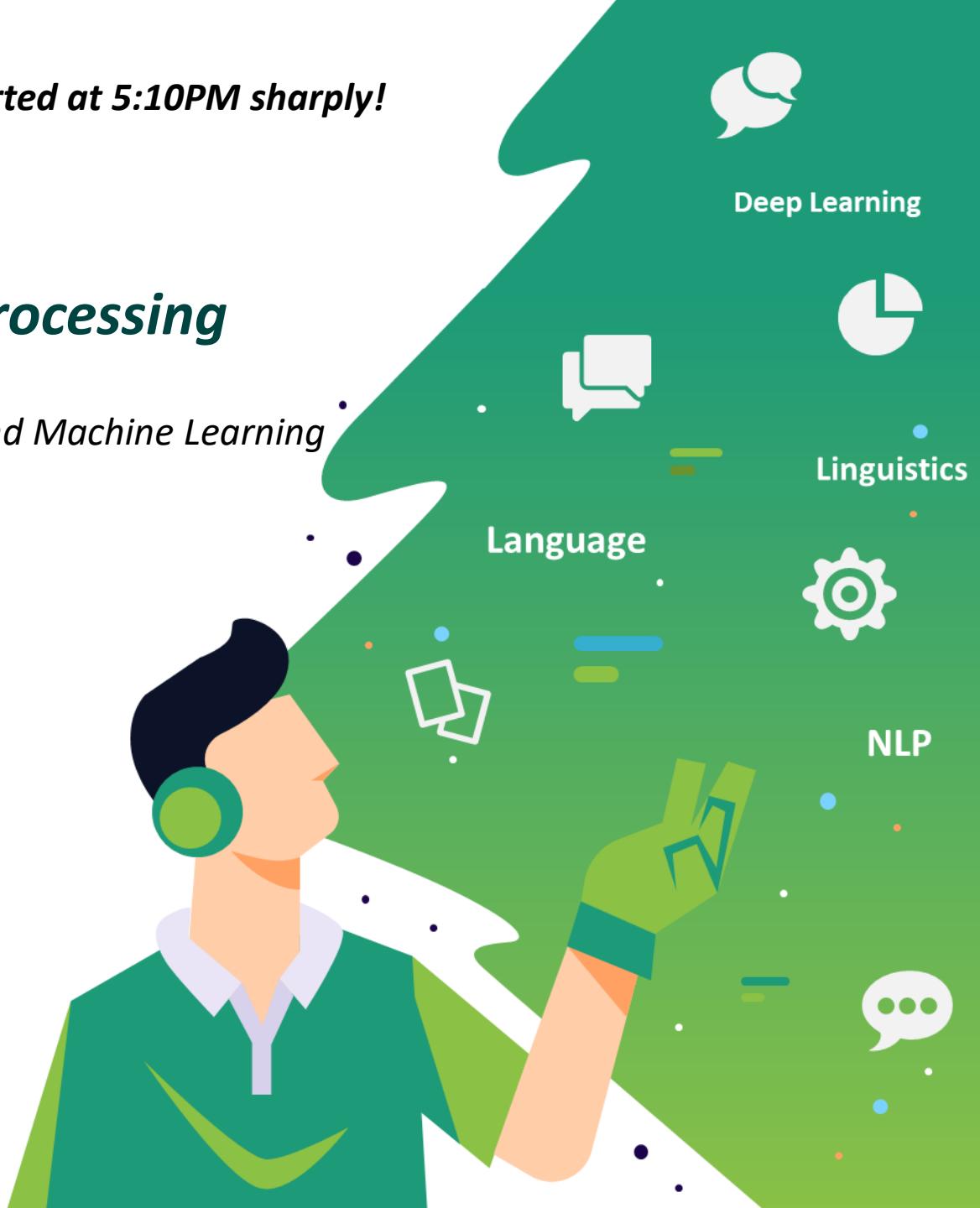
Natural Language Processing

Lecture 3: Word Classification and Machine Learning

Dr. Caren Han

Semester 1, 2022

*School of Computer Science,
University of Sydney*



0 LECTURE PLAN

Lecture 3: Word Classification and Machine Learning

1. Previous Lecture: Word Embedding Review
2. Word Embedding Evaluation
3. Deep Neural Network for Natural Language Processing
 1. Perceptron and Neural Network (NN)
 2. Multilayer Perceptron
 3. Applications
4. Next Week Preview

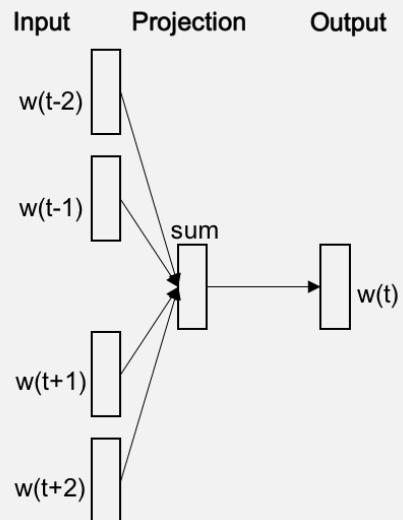
See how the Deep Learning can be used for NLP

 - Text Classification, etc.

Word2Vec Models

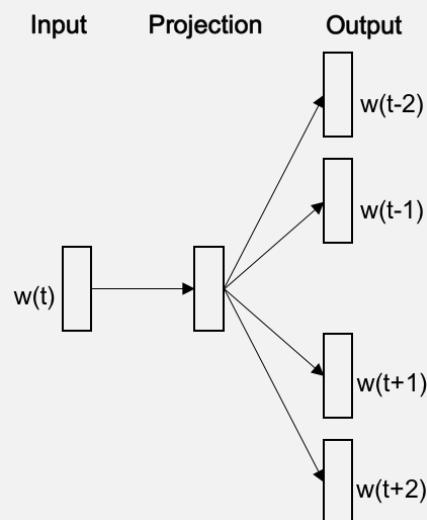
CBOW

Predict center word from (bag of) context words



Skip-gram

Predict context words given center word



1 Previous Lecture Review

Word2Vec with Continuous Bag of Words (CBOW)

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”

Using window slicing, develop the training data

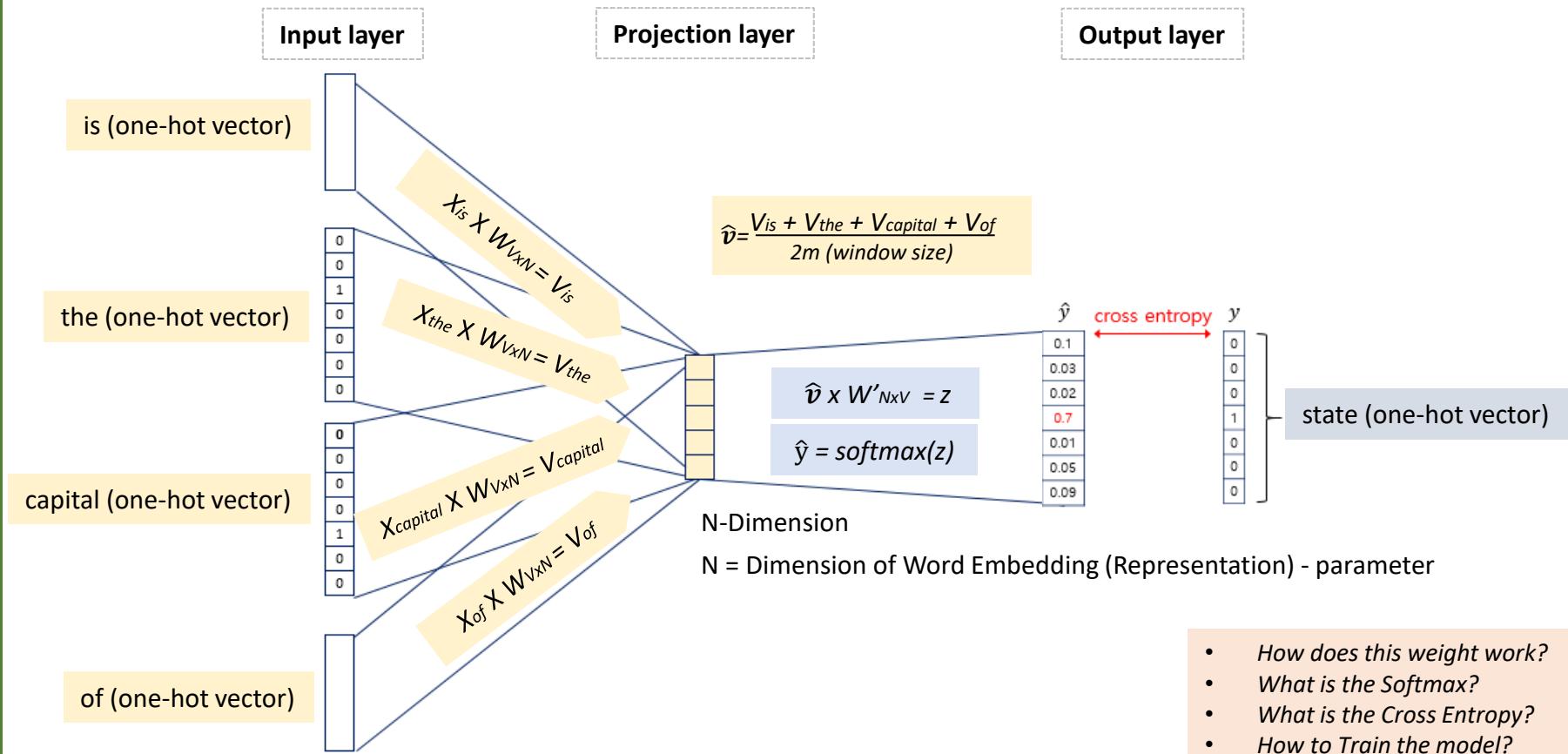
Center word	Context (“outside”) word	Sydney is the state capital of NSW
[1,0,0,0,0,0,0]	[0,1,0,0,0,0,0], [0,0,1,0,0,0,0]	Sydney is the state capital of NSW
[0,1,0,0,0,0,0]	[1,0,0,0,0,0,0], [0,0,1,0,0,0,0], [0,0,0,1,0,0,0]	Sydney is the state capital of NSW
[0,0,1,0,0,0,0]	[1,0,0,0,0,0,0], [0,1,0,0,0,0,0] [0,0,0,1,0,0,0], [0,0,0,0,1,0,0]	Sydney is the state capital of NSW
[0,0,0,1,0,0,0]	[0,1,0,0,0,0,0], [0,0,1,0,0,0,0] [0,0,0,0,1,0,0], [0,0,0,0,0,1,0]	Sydney is the state capital of NSW
[0,0,0,0,1,0,0]	[0,0,1,0,0,0,0], [0,0,0,1,0,0,0] [0,0,0,0,0,1,0], [0,0,0,0,0,0,1]	Sydney is the state capital of NSW
[0,0,0,0,0,1,0]	[0,0,0,1,0,0,0], [0,0,0,0,1,0,0] [0,0,0,0,0,0,1]	Sydney is the state capital of NSW
[0,0,0,0,0,0,1]	[0,0,0,0,1,0,0], [0,0,0,0,0,1,0]	Sydney is the state capital of NSW

█ Center word
█ Context (“outside”) word

CBOV – Neural Network Architecture

Predict center word from (bag of) context words

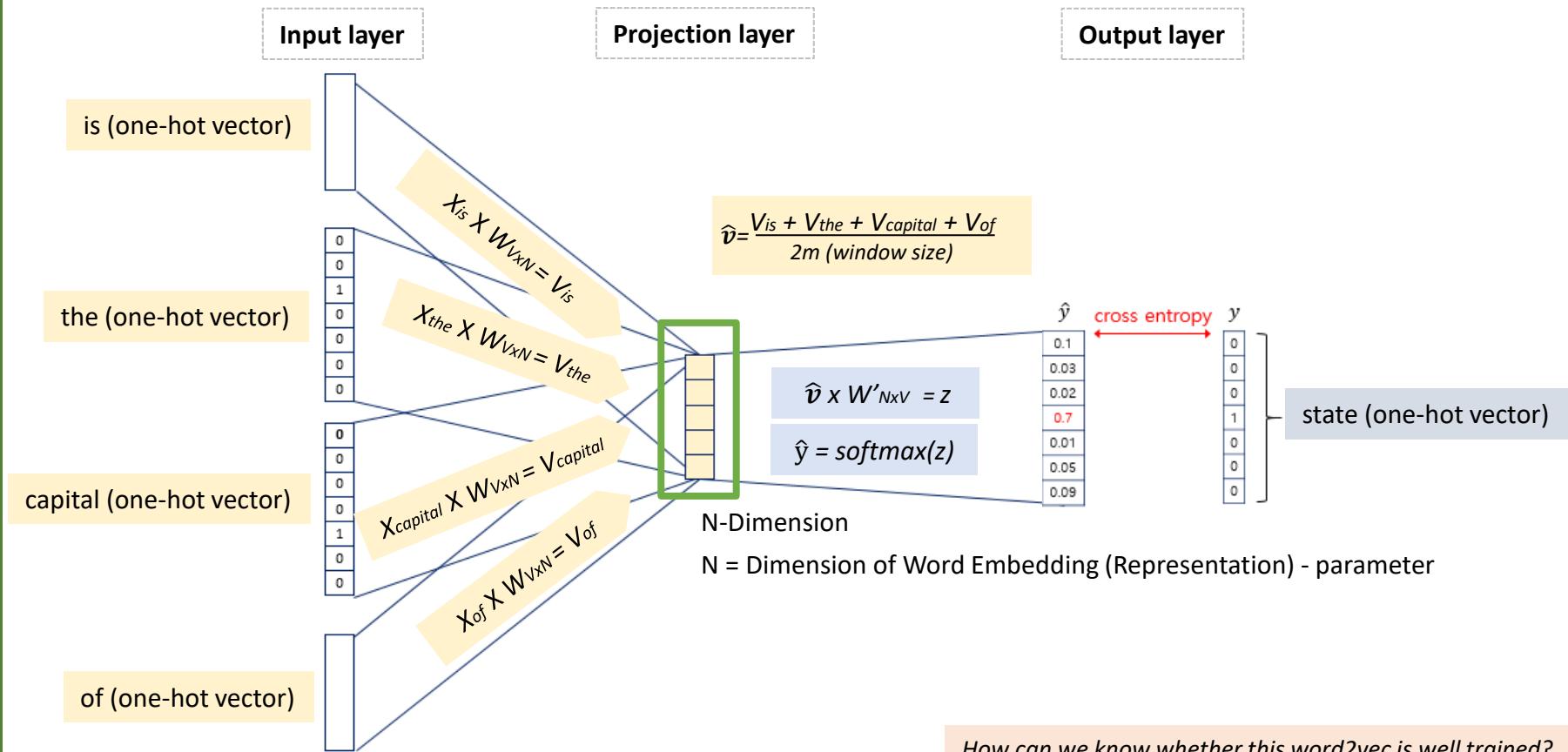
Sentence: “Sydney is the state capital of NSW”



CBOW – Neural Network Architecture

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”



0 LECTURE PLAN

Lecture 3: Word Classification and Machine Learning

1. Previous Lecture: Word Embedding Review
2. **Word Embedding Evaluation**
3. Deep Neural Network for Natural Language Processing
 1. Perceptron and Neural Network (NN)
 2. Multilayer Perceptron
 3. Applications
4. Next Week Preview

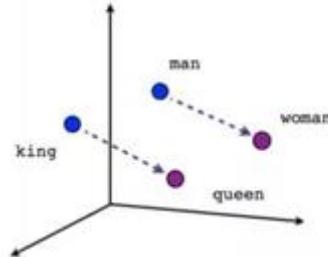
See how the Deep Learning can be used for NLP

 - Text Classification, etc.

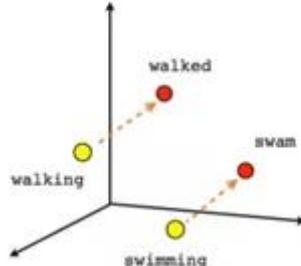
Word Embedding Evaluation

How to evaluate word vectors?

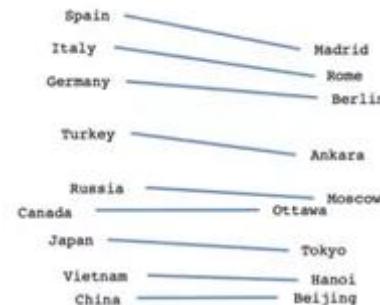
Type	How to work / Benefit
Intrinsic	Evaluation on a specific/intermediate subtask
Intrinsic	<ul style="list-style-type: none"> Fast to compute Helps to understand that system Not clear if really helpful unless correlation to real task is established
Extrinsic	Evaluation on a real task
Extrinsic	<ul style="list-style-type: none"> Can take a long time to compute accuracy Unclear if the subsystem is the problem or its interaction or other subsystems



Male-Female



Verb tense



Country-Capital

2

Word Embedding Evaluation

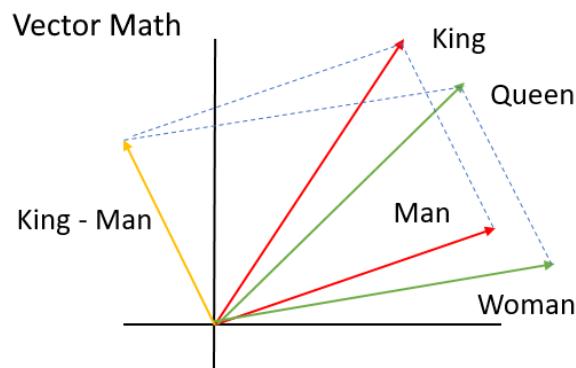
Intrinsic word vector evaluation

Word Vector Analogies

$$a \leftrightarrow b :: c \leftrightarrow ???$$

$$\text{man} \leftrightarrow \text{women} :: \text{king} \leftrightarrow ???$$

- Evaluate word vectors by how well their cosine distance after addition captures intuitive semantic and syntactic analogy questions



Word Embedding Evaluation

Intrinsic word vector evaluation

Word Vector Analogies

King – Man + Woman = ?

No	Training Dataset	Type	Result
1	 TED Script	word2vec CBOW	President
2		word2vec Skip-gram	Luther
3		fastText CBOW	Kidding
4		fastText Skip-gram	Jarring
5	 Google News	word2vec CBOW	queen
6		word2vec Skip-gram	queen

2

Word Embedding Evaluation

Intrinsic word vector evaluation

Evaluation Result Comparison

The Semantic-Syntactic word relationship tests for understanding of a wide variety of relationships as shown below.

Using 640-dimensional word vectors, a skip-gram trained model achieved 55% semantic accuracy and 59% syntactic accuracy.

Table 3: *Comparison of architectures using models trained on the same data, with 640-dimensional word vectors. The accuracies are reported on our Semantic-Syntactic Word Relationship test set, and on the syntactic relationship test set of [20]*

Model Architecture	Semantic-Syntactic Word Relationship test set		MSR Word Relatedness Test Set [20]
	Semantic Accuracy [%]	Syntactic Accuracy [%]	
RNNLM	9	36	35
NNLM	23	53	47
CBOW	24	64	61
Skip-gram	55	59	56

(Original Word2vec Paper - Mikolov et al.2013)

Word Embedding Evaluation

Intrinsic word vector evaluation

Evaluation Result Comparison

The Semantic-Syntactic word relationship tests for understanding of a wide variety of relationships as shown below.

Table 2: Results on the word analogy task, given as percent accuracy. Underlined scores are best within groups of similarly-sized models; bold scores are best overall. HPCA vectors are publicly available²; (i)vLBL results are from (Mnih et al., 2013); skip-gram (SG) and CBOW results are from (Mikolov et al., 2013a,b); we trained SG[†] and CBOW[†] using the word2vec tool³. See text for details and a description of the SVD models.

Model	Dim.	Size	Sem.	Syn.	Tot.
ivLBL	100	1.5B	55.9	50.1	53.2
HPCA	100	1.6B	4.2	16.4	10.8
GloVe	100	1.6B	<u>67.5</u>	<u>54.3</u>	<u>60.3</u>
SG	300	1B	61	61	61
CBOW	300	1.6B	16.1	52.6	36.1
vLBL	300	1.5B	54.2	<u>64.8</u>	60.0
ivLBL	300	1.5B	65.2	63.0	64.0
GloVe	300	1.6B	80.8	61.5	<u>70.3</u>
SVD	300	6B	6.3	8.1	7.3
SVD-S	300	6B	36.7	46.6	42.1
SVD-L	300	6B	56.6	63.0	60.1
CBOW [†]	300	6B	63.6	<u>67.4</u>	65.7
SG [†]	300	6B	73.0	66.0	69.1
GloVe	300	6B	<u>77.4</u>	67.0	71.7
CBOW	1000	6B	57.3	68.9	63.7
SG	1000	6B	66.1	65.1	65.6
SVD-L	300	42B	38.4	58.2	49.2
GloVe	300	42B	81.9	69.3	75.0

(Original Glove Paper - Pennington et al.2014)

Word Embedding Evaluation

Intrinsic word vector evaluation

Evaluation Result Comparison

The Semantic-Syntactic word relationship tests for understanding of a wide variety of relationships as shown below.

Window-Size (m) and Vector Dimension (N)

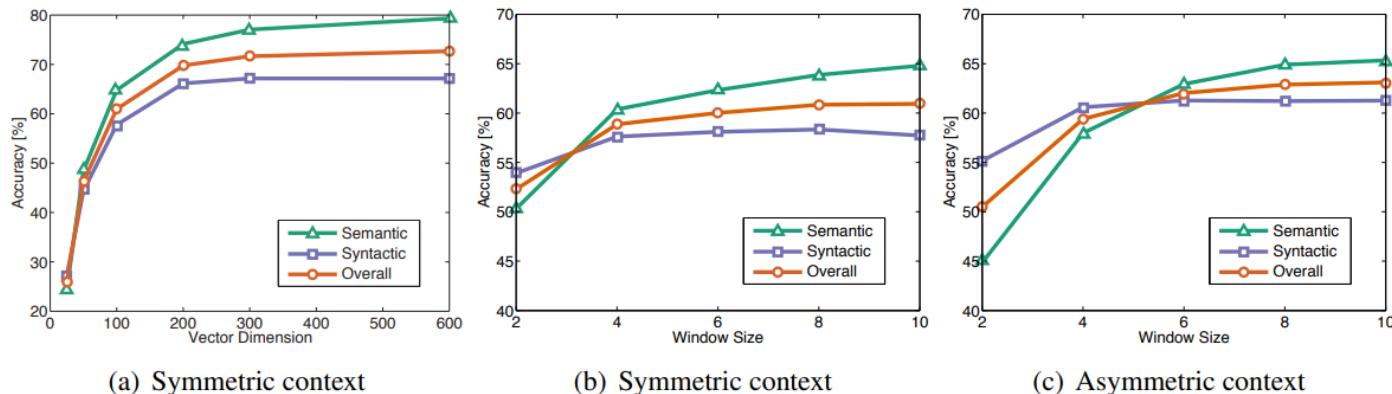


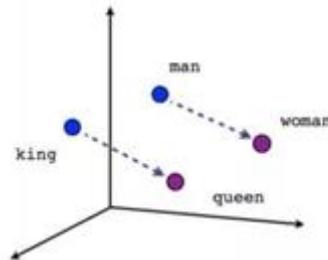
Figure 2: Accuracy on the analogy task as function of vector size and window size/type. All models are trained on the 6 billion token corpus. In (a), the window size is 10. In (b) and (c), the vector size is 100.

(Original Glove Paper - Pennington et al.2014)

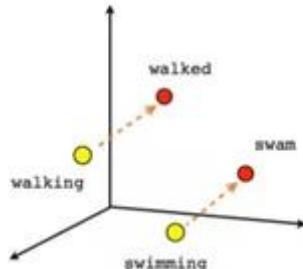
Word Embedding Evaluation

How to evaluate word vectors?

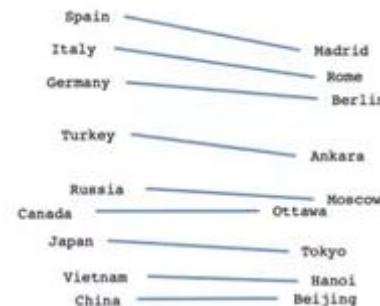
Type	How to work / Benefit
Intrinsic	<p>Evaluation on a specific/intermediate subtask</p> <ul style="list-style-type: none"> • Fast to compute • Helps to understand that system • Not clear if really helpful unless correlation to real task is established
Extrinsic	<p>Evaluation on a real task</p> <ul style="list-style-type: none"> • Can take a long time to compute accuracy • Unclear if the subsystem is the problem or its interaction or other subsystems



Male-Female



Verb tense



Country-Capital

0 LECTURE PLAN

Lecture 3: Word Classification and Machine Learning

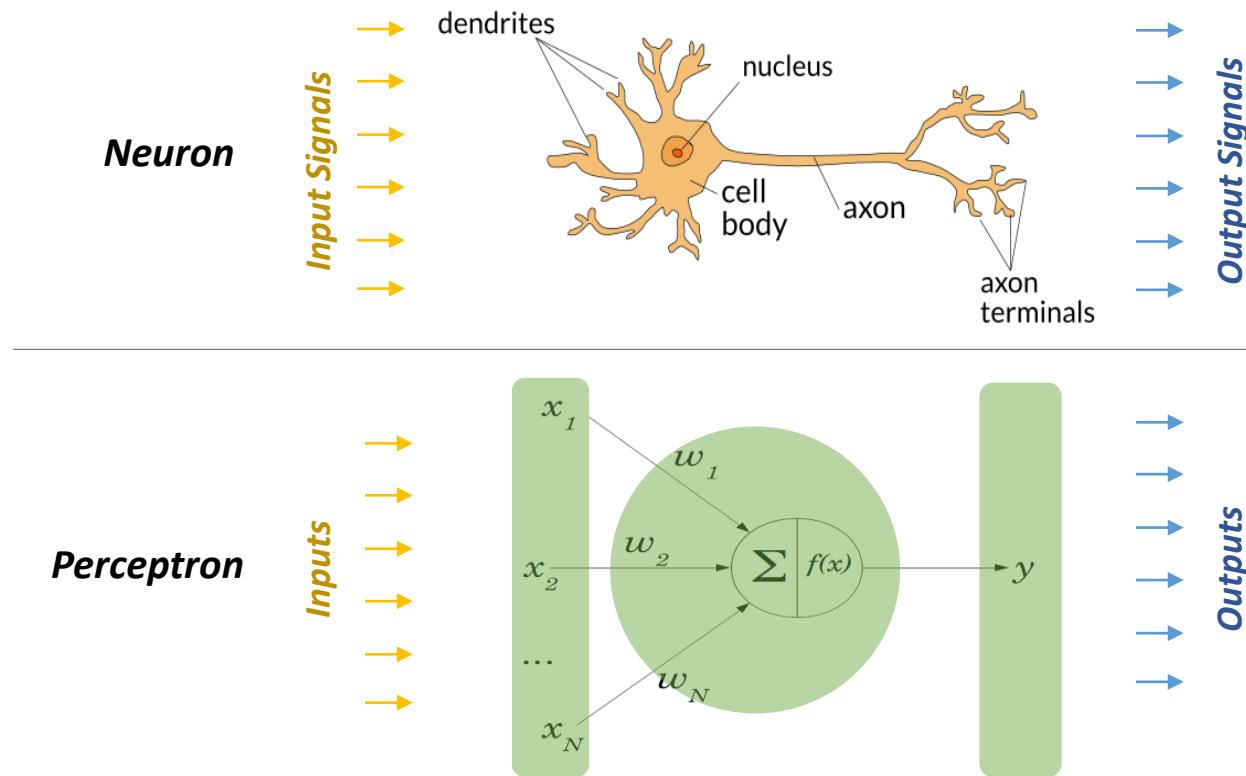
1. Previous Lecture: Word Embedding Review
2. Word Embedding Evaluation
3. **Deep Neural Network for Natural Language Processing**
 1. Perceptron and Neural Network (NN)
 2. Multilayer Perceptron
 3. Applications
4. Next Week Preview

See how the Deep Learning can be used for NLP

 - Text Classification, etc.

Deep Learning with Neural Network

Neuron and Perceptron

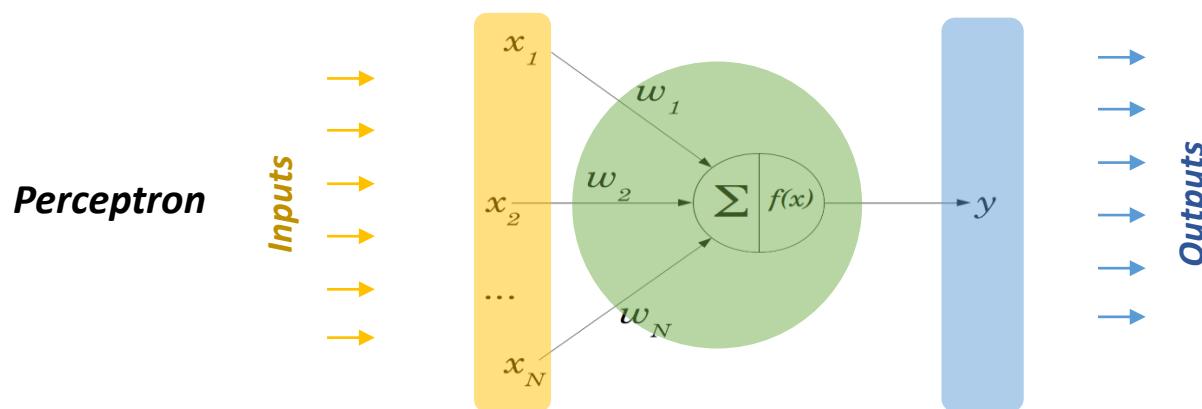


3 Deep Learning for NLP

Deep Learning with Neural Network

Inputs and Outputs (Labels) for Natural Language Processing

x_i	Inputs	Features words (indices or vectors!), context windows, sentences, documents, etc.
y_i	Outputs (labels)	What we try to predict/classify <ul style="list-style-type: none"> E.g. word meaning, sentiment, name entity



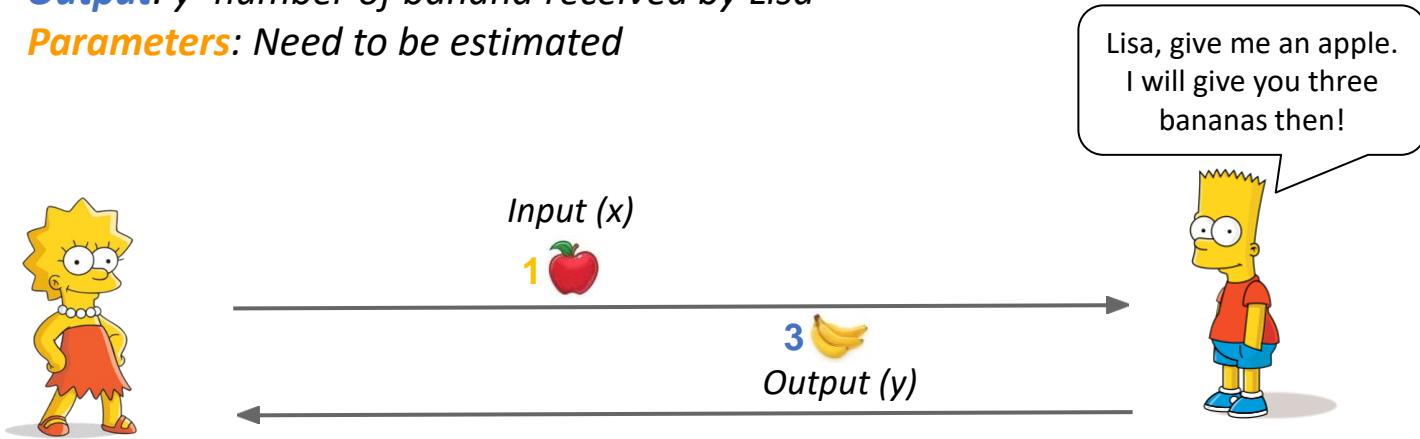
3 Deep Learning for NLP

Deep Learning with Neural Network

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



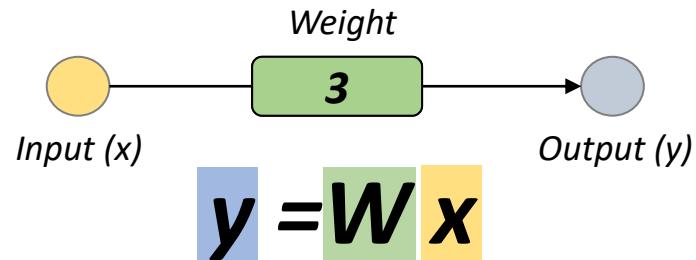
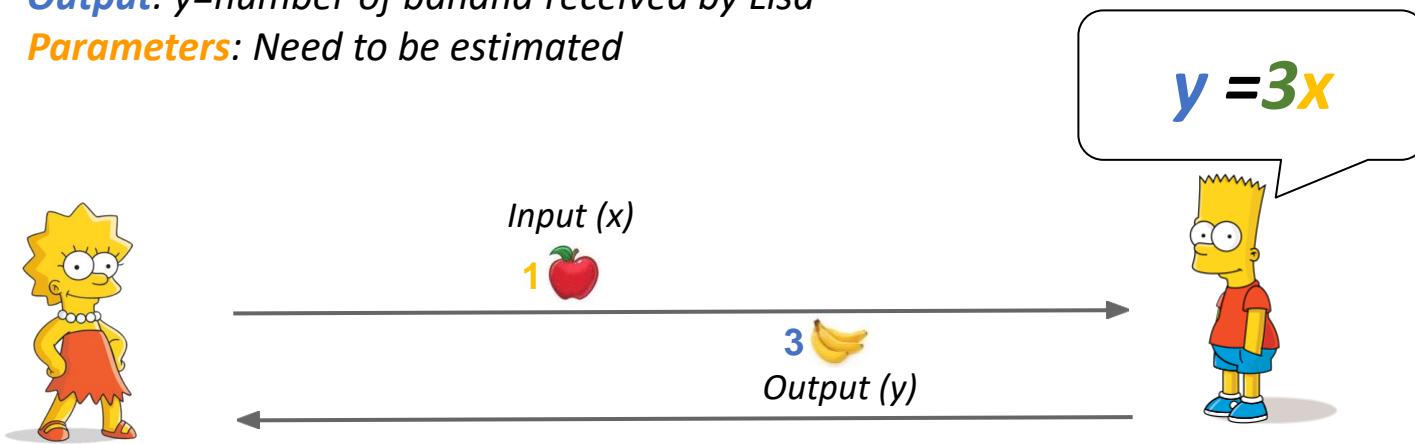
3 Deep Learning for NLP

Deep Learning with Neural Network - Model

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



3 Deep Learning for NLP



3 Deep Learning for NLP

Deep Learning with Neural Network - Model

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



Guess how much I will
give you back!



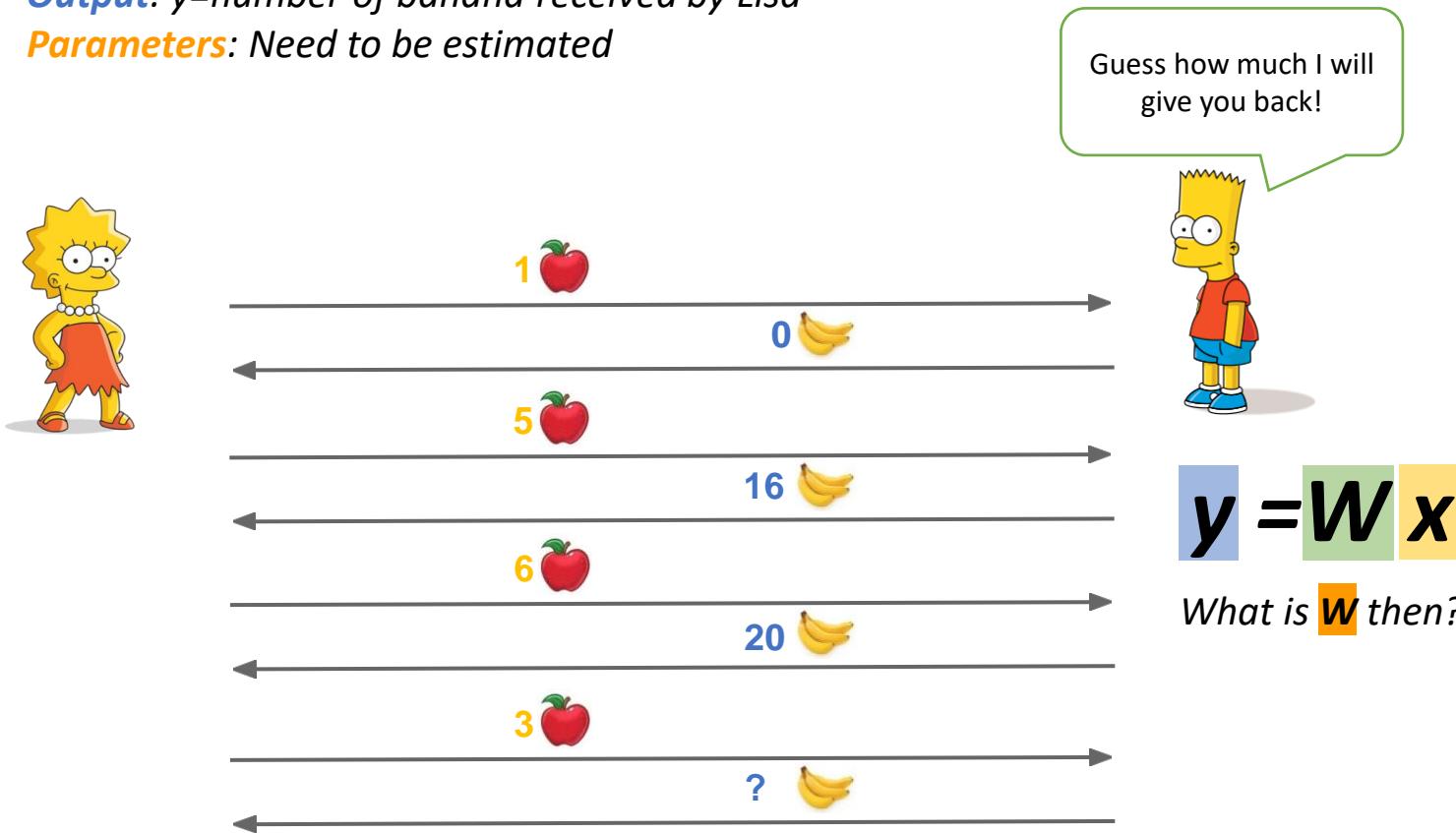
3 Deep Learning for NLP

Deep Learning with Neural Network - Parameter

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



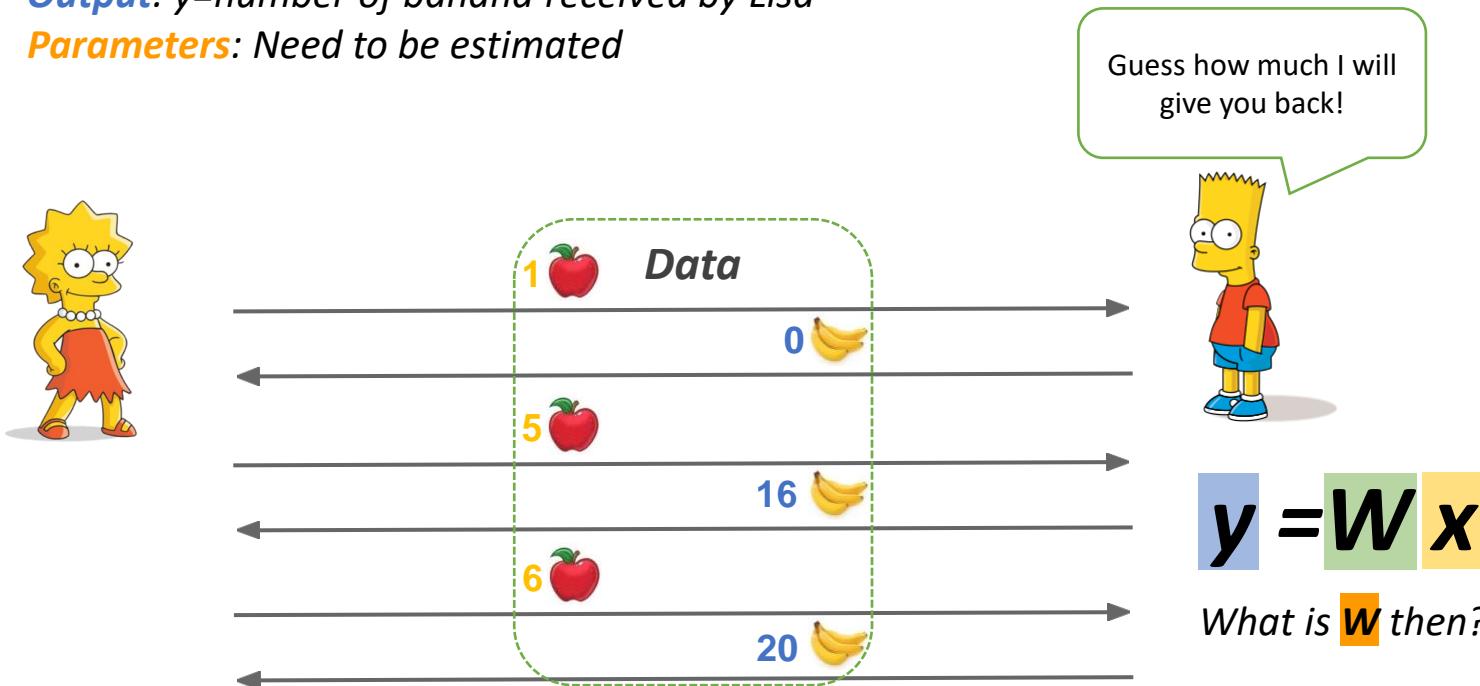
3 Deep Learning for NLP

Deep Learning with Neural Network - Parameter

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



3 Deep Learning for NLP

Deep Learning with Neural Network - Parameter

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated

Data

x 	y 
1	0
5	16
6	20

$$\mathbf{y} = \mathbf{W} \mathbf{x}$$

What is \mathbf{W} then?

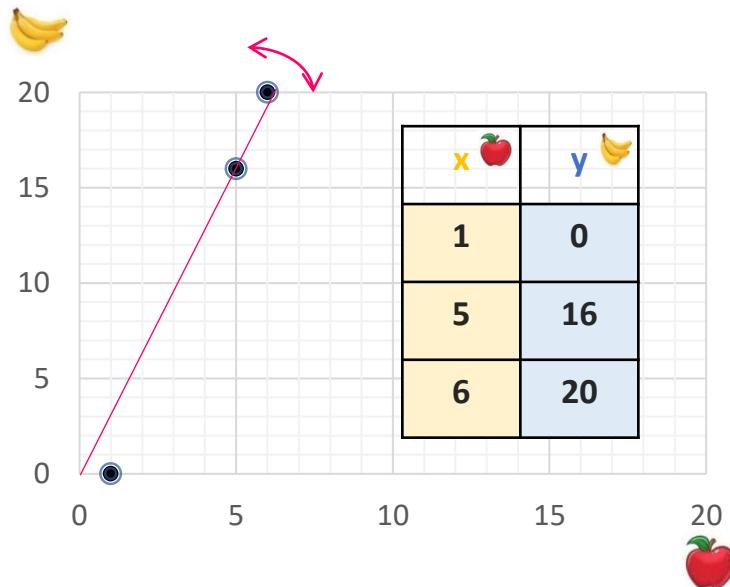
3 Deep Learning for NLP

Deep Learning with Neural Network - Parameter

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



$$y = Wx$$

What is W then?

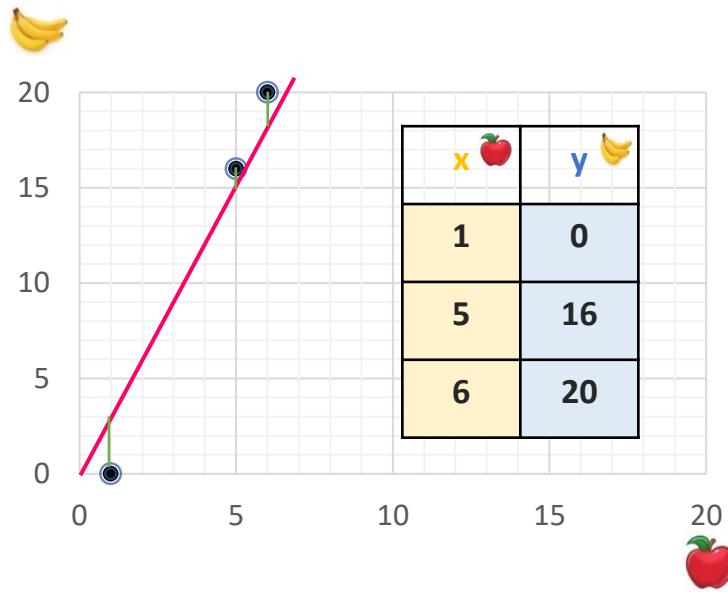
3 Deep Learning for NLP

Deep Learning with Neural Network - Parameter

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



$$\mathbf{y} = \mathbf{W} \mathbf{x}$$

What if W is 3?

$$3 = 3 \times 1$$

$$15 = 3 \times 5$$

$$20 = 3 \times 6$$

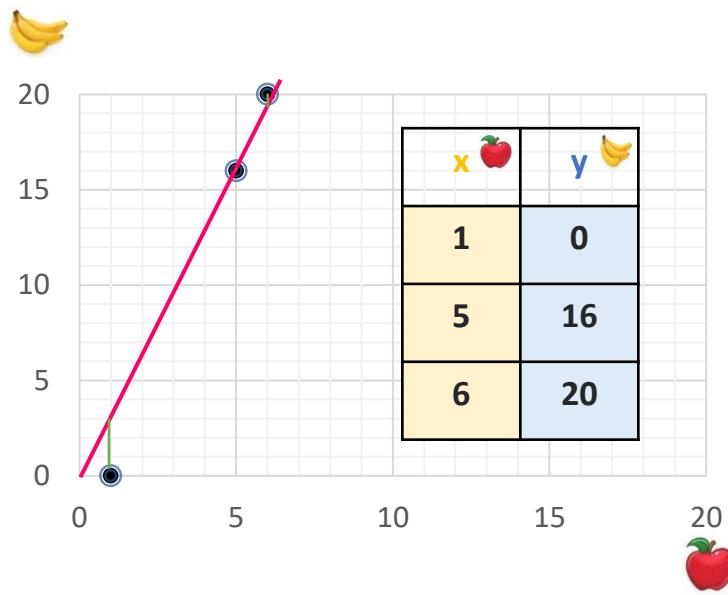
3 Deep Learning for NLP

Deep Learning with Neural Network - Parameter

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



$$y = Wx$$

What if W is 3.2?

$$3.2 = 3.2 \times 1$$

$$16 = 3.2 \times 5$$

$$19.2 = 3.2 \times 6$$

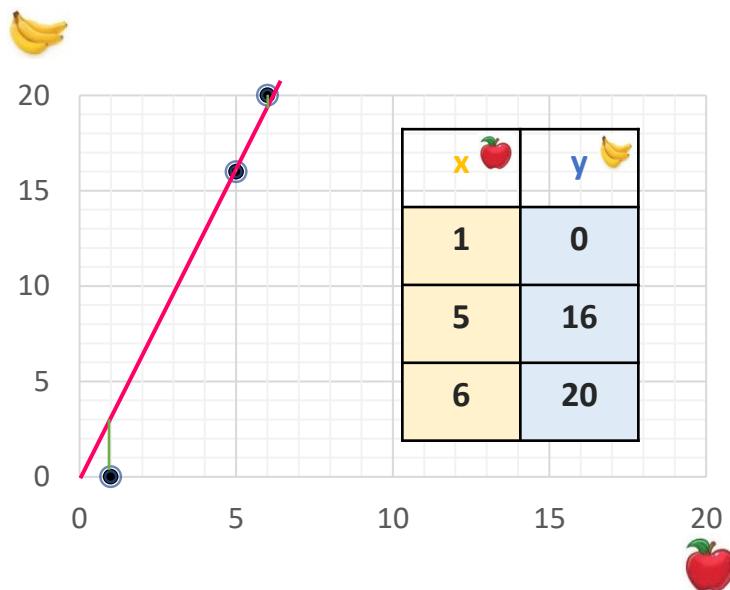
3 Deep Learning for NLP

Deep Learning with Neural Network - Parameter

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



$$y = \underset{\text{weight}}{W}x + \underset{\text{bias}}{b}$$

Weight is not enough...

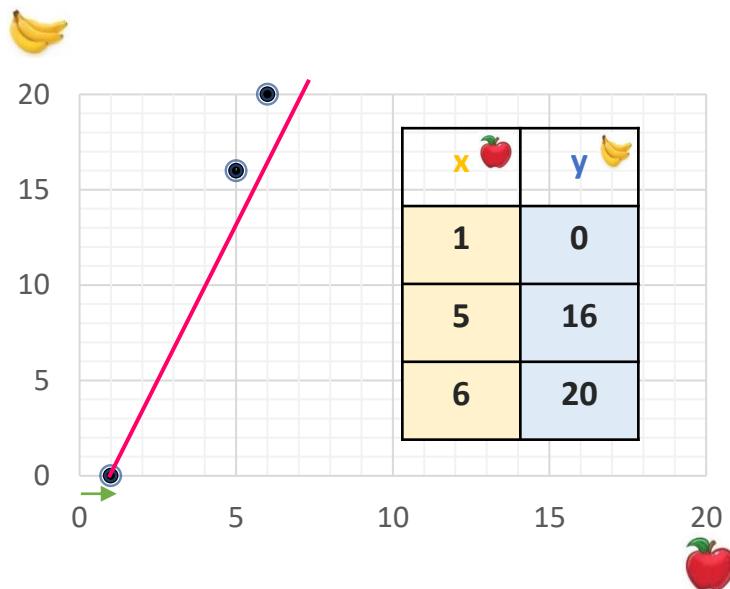
3 Deep Learning for NLP

Deep Learning with Neural Network - Parameter

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



$$y = \text{weight} \times x + \text{bias}$$

How can we find the parameters, w and b ?

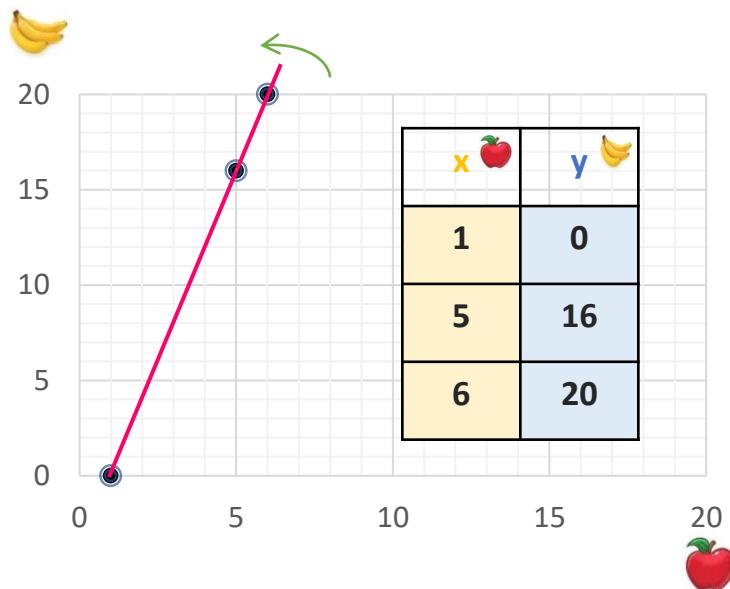
3 Deep Learning for NLP

Deep Learning with Neural Network - Parameter

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



$$y = \text{weight} \cdot x + \text{bias}$$

How can we find the parameters, w and b ?

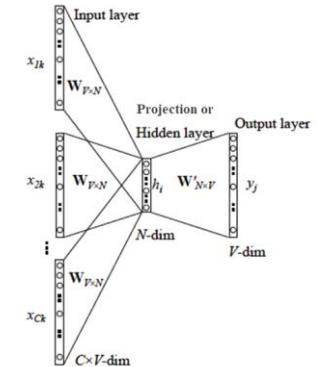
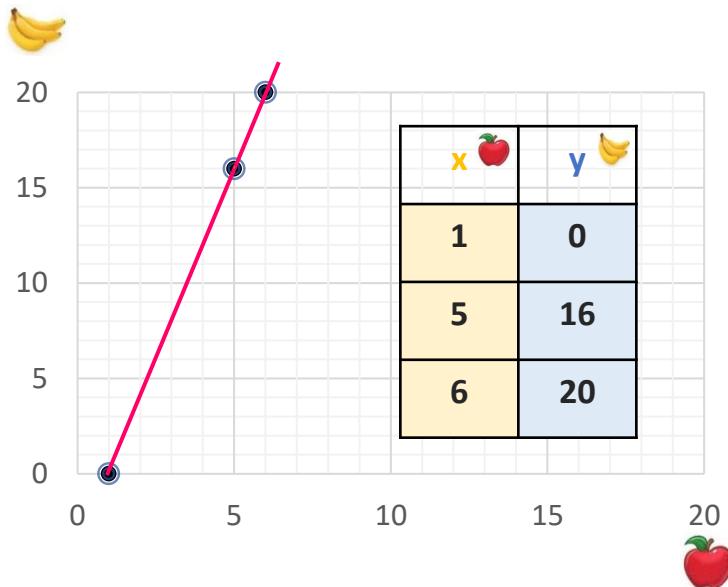
3 Deep Learning for NLP

Deep Learning with Neural Network - Parameter

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



Model $\textcolor{blue}{y} = \textcolor{green}{W} \textcolor{orange}{x} + \textcolor{green}{b}$

How can we find the parameters, w and b ?

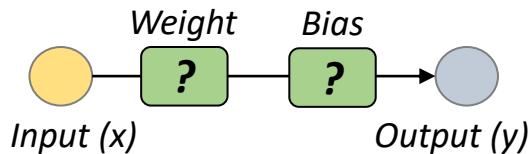
3 Deep Learning for NLP

Deep Learning with Neural Network - Cost

Actual Data

$$y = ? \boxed{x} + ?$$

x 🍎	y 🍌
1	0
5	16
6	20



Model Ex#1

$$\hat{y} = \boxed{1} \boxed{x} + \boxed{0}$$

predicted		actual
x 🍎	ŷ 🍌	y 🍌
1	1	0
5	5	16
6	6	20

Model Ex#2

$$\hat{y} = \boxed{2} \boxed{x} + \boxed{2}$$

predicted		actual
x 🍎	ŷ 🍌	y 🍌
1	4	0
5	12	16
6	14	20



Which one is closer?

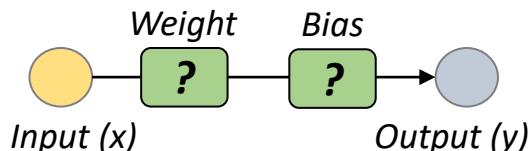
3 Deep Learning for NLP

Deep Learning with Neural Network – Cost (loss)

Actual Data

$$y = ? \boxed{x} + ?$$

x 🍎	y 🍌
1	0
5	16
6	20



Model Ex#1

$$\hat{y} = \boxed{1} \boxed{x} + \boxed{0}$$

predicted	actual	cost
x 🍎	ŷ 🍔	y 🍌
1	1	0
5	5	16
6	6	20

Model Ex#2

$$\hat{y} = \boxed{2} \boxed{x} + \boxed{2}$$

predicted	actual	cost
x 🍎	ŷ 🍔	y 🍌
1	4	0
5	12	16
6	14	20

😎 Let's calculate the cost(loss)!

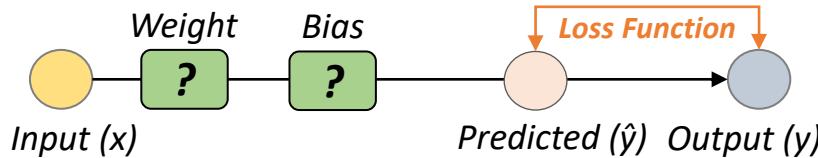
Mean Squared Error
(MSE)

$$C(w, b) = \sum (y_n - \hat{y}_n)^2$$

n ∈ {0, 1, 2}

3 Deep Learning for NLP

WAIT! Loss Function? Cost Calculation?

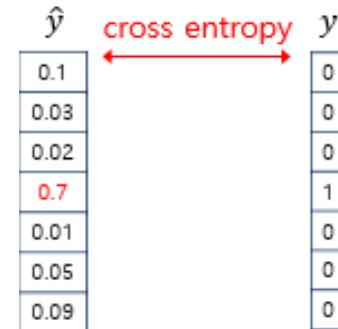


1) Mean Squared Error (MSE): measures the average of the squares of the errors

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

2) Cross Entropy: calculating the difference between two probability distributions

$$L_{\text{cross-entropy}}(\hat{\mathbf{y}}, \mathbf{y}) = - \sum_i y_i \log(\hat{y}_i)$$



The diagram shows two vertical vectors representing probability distributions. The left vector, labeled $\hat{\mathbf{y}}$, contains the predicted probabilities: 0.1, 0.03, 0.02, 0.7 (highlighted in red), 0.01, 0.05, and 0.09. The right vector, labeled \mathbf{y} , contains the target values: 0, 0, 0, 1, 0, 0, and 0. A red double-headed arrow labeled "cross entropy" connects the two vectors, indicating the calculation of the cross-entropy loss between them.

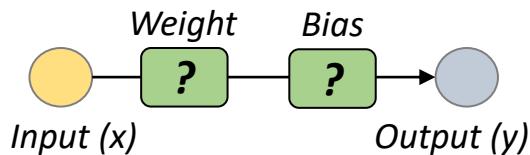
3 Deep Learning for NLP

Deep Learning with Neural Network - Cost (loss)

Actual Data

$$y = ? \boxed{x} + ?$$

x 🍎	y 🍌
1	0
5	16
6	20



Model Ex#1

$$\hat{y} = \boxed{1} \boxed{x} + \boxed{0}$$

predicted		actual	cost
x 🍎	ŷ 🍔	y 🍌	$(y - \hat{y})^2$
1	1	0	1
5	5	16	121
6	6	20	196

$$C(1,0) = 318$$

Model Ex#2

$$\hat{y} = \boxed{2} \boxed{x} + \boxed{2}$$

predicted		actual	cost
x 🍎	ŷ 🍔	y 🍌	$(y - \hat{y})^2$
1	4	0	16
5	12	16	16
6	14	20	36

$$C(2,2) = 68$$



😎 Let's calculate the cost!

$$C(w, b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)$$

3 Deep Learning for NLP

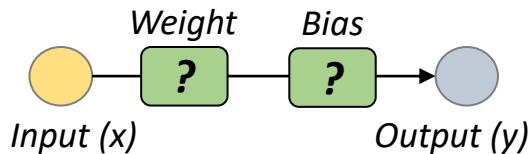
Deep Learning with Neural Network - Cost (loss)

Actual Data

$$y = ? \ x + ?$$

weight bias

x 🍎	y 🍌
1	0
5	16
6	20



Model Ex#1

$$\hat{y} = 1 \ x + 0$$

weight bias

x 🍎	predicted \hat{y}	actual y	$(y-\hat{y})^2$
1	1	0	1
5	5	16	121
6	6	20	196

$$C(1,0) = 318$$

Model Ex#2

$$\hat{y} = 2 \ x + 2$$

weight bias

x 🍎	predicted \hat{y}	actual y	$(y-\hat{y})^2$
1	4	0	16
5	12	16	16
6	14	20	36

$$C(2,2) = 68$$



😎 Let's calculate the costs and get the lowest one!

$$\arg \min C(w,b)$$

$$w,b \in [-\infty, \infty]$$

3 Deep Learning for NLP

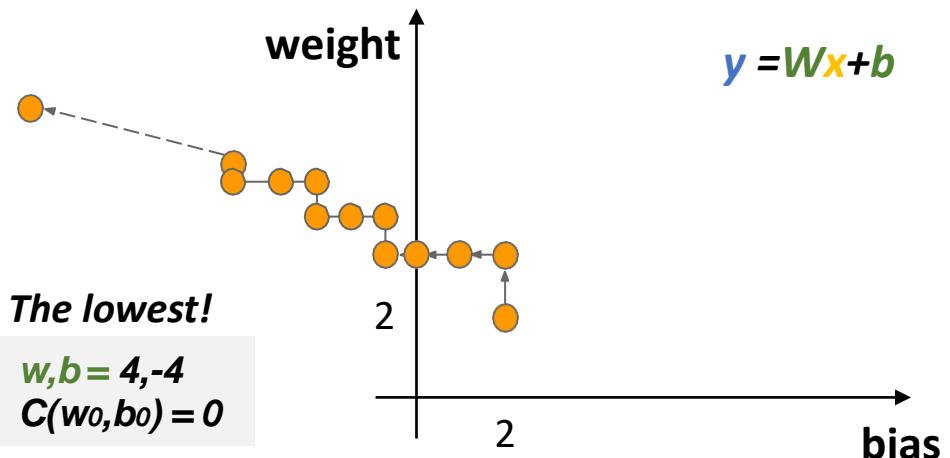
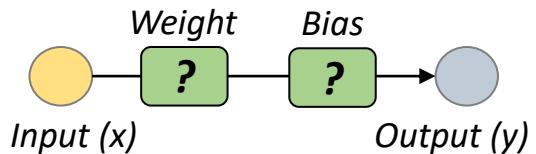
Deep Learning with Neural Network - Optimizer

Actual Data

$$y = ? \times x + ?$$

weight bias

x 🍎	y 🍌
1	0
5	16
6	20



😎 Let's calculate the costs and get the lowest one!

$$\arg \min C(w, b)$$

$$w, b \in [-\infty, \infty]$$

3 Deep Learning for NLP

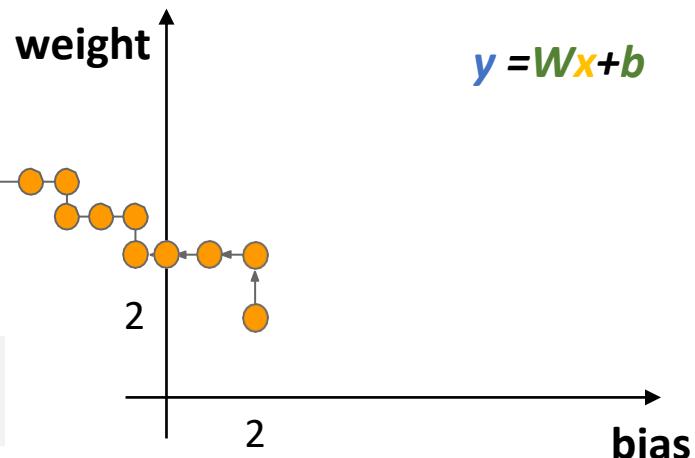
Backpropagation (weight update)

Deep Learning with Neural Network - Optimizer

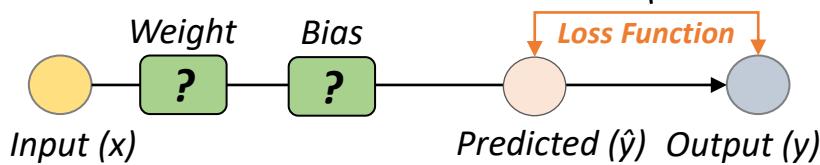
$$y = 4 \text{ } \boxed{x} - 4$$

x 🍎	y 🍌
1	0
5	16
6	20

The lowest!
 $w, b = 4, -4$
 $C(w_0, b_0) = 0$



Backpropagation (weight update)



$\arg \min C(w, b)$

$w, b \in [-\infty, \infty]$

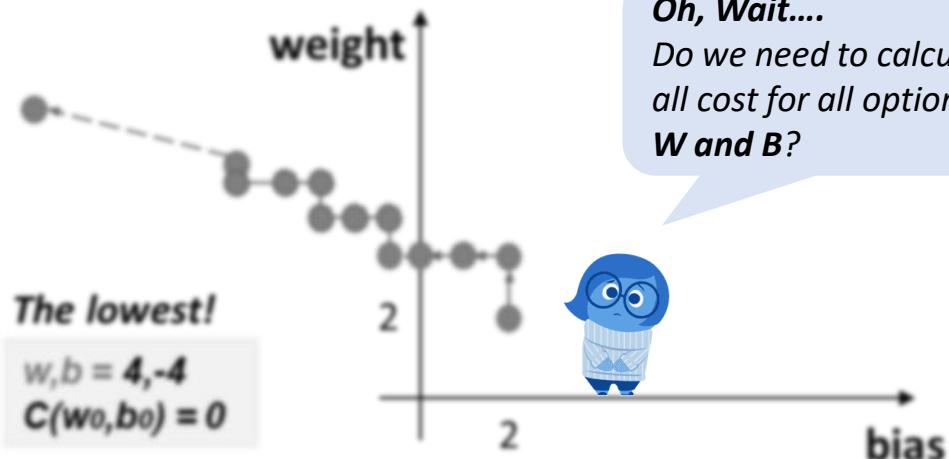
3 Deep Learning for NLP

Backpropagation (weight update)

Deep Learning with Neural Network - Optimizer

$$y = 4 \text{ weight} | x - 4 \text{ bias}$$

x 🍎	y 🍌
1	0
5	16
6	20



Expensive to compute
(hours or days)

$$\arg \min C(w, b)$$

$$w, b \in [-\infty, \infty]$$

3 Deep Learning for NLP

Finding the Optimal weight and bias – Gradient Descent



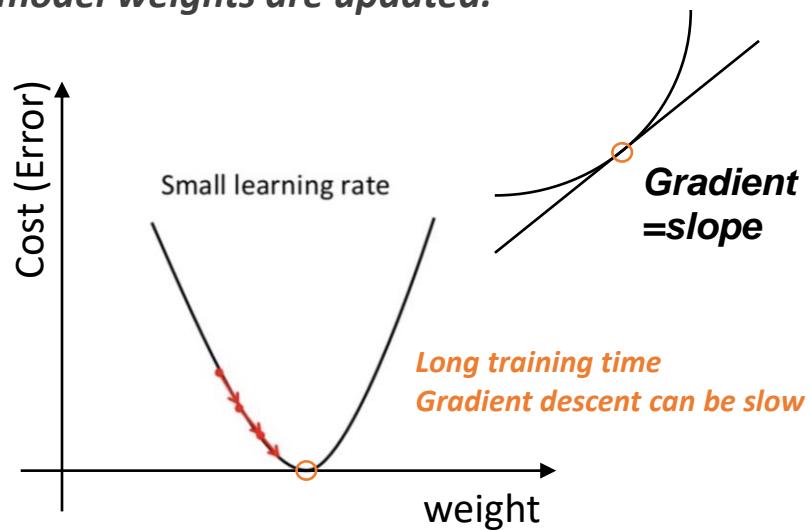
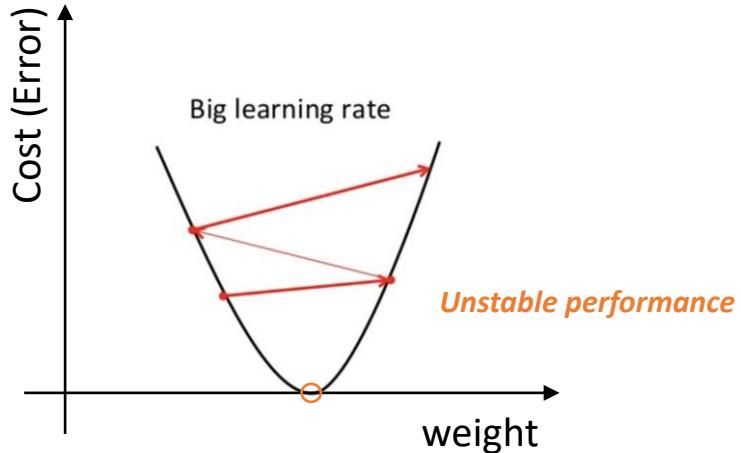
There are different types of Gradient descent optimization algorithms:

Batch Gradient Descent, Stochastic Gradient Descent, Momentum, Adam, etc.

3 Deep Learning for NLP

Choose the optimal Learning Rate!

Learning Rate: a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated.



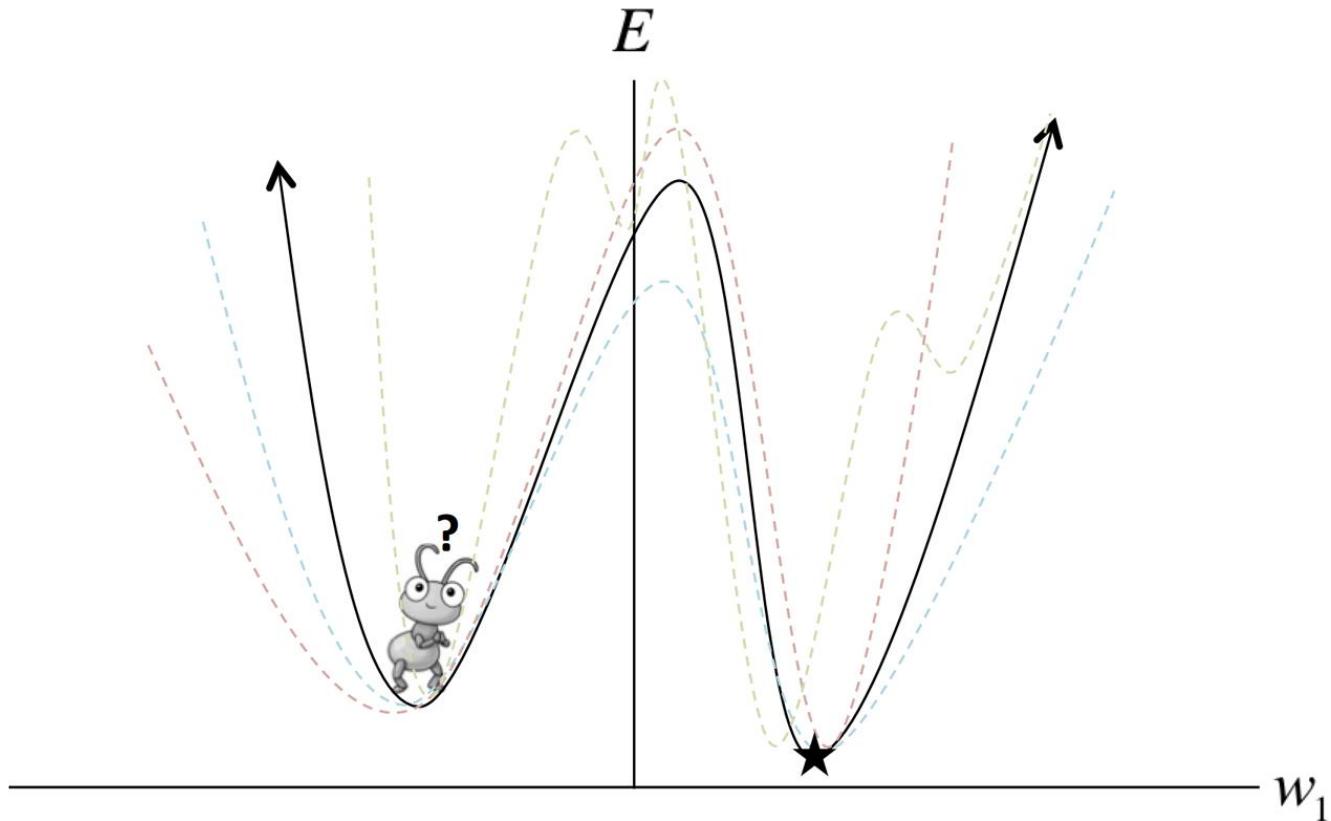
$$\text{new_weight} = \text{existing_weight} - \text{learning_rate} * \text{gradient}$$

$$\text{new_weight} = \text{existing_weight} - \text{learning_rate} * (\text{current_output} - \text{desired output})$$

$$* \text{gradient}(\text{current output}) * \text{existing_input}$$

3 Deep Learning for NLP

Finding the Optimal weight and bias – Gradient Descent



There are different types of Gradient descent optimization algorithms:
Batch Gradient Descent, Stochastic Gradient Descent, Momentum, Adam, etc.

3 Deep Learning for NLP

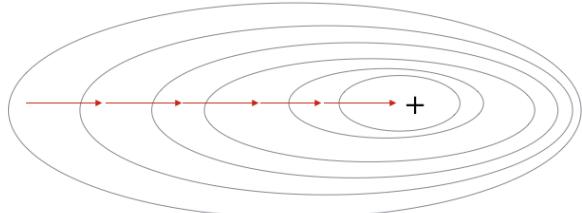
Stochastic Gradient Descent

***The cost would be very expensive if we calculate it for all windows in the corpus!
You would wait a very long time before making a single update!***

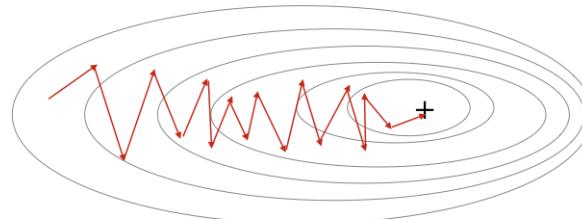
*The Solution can be used different Gradient Descent Method.
The most common – “**Stochastic Gradient Descent (SGD)**”*

*Vanilla (Batch) gradient descent performs redundant computations for large datasets, as it **recomputes gradients for similar examples before each parameter update**. SGD does away with this redundancy by performing one update at a time. It is therefore usually much faster and can also be used to learn online.*

Gradient Descent

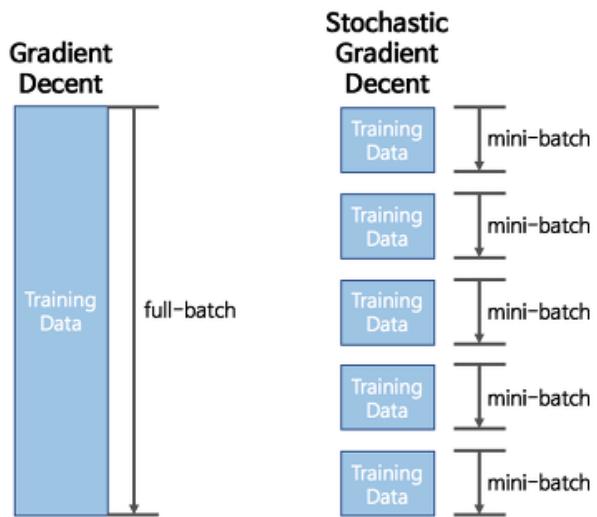


Stochastic Gradient Descent



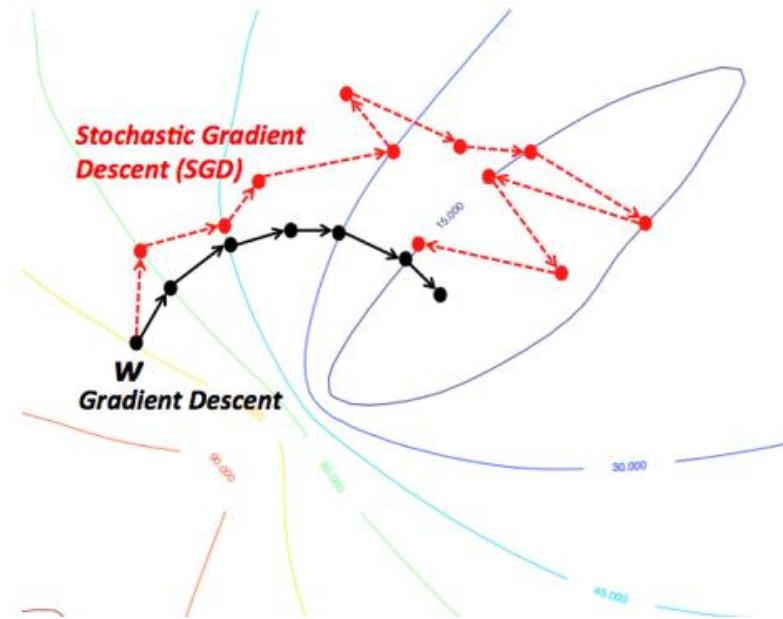
3 Deep Learning for NLP

Stochastic Gradient Descent



Gradient Descent

- Calculate with all training data
- Forward the optimal one step (at a time)
- Accurate but very slow
- e.g. 6 steps * 1 hour = 6 hours

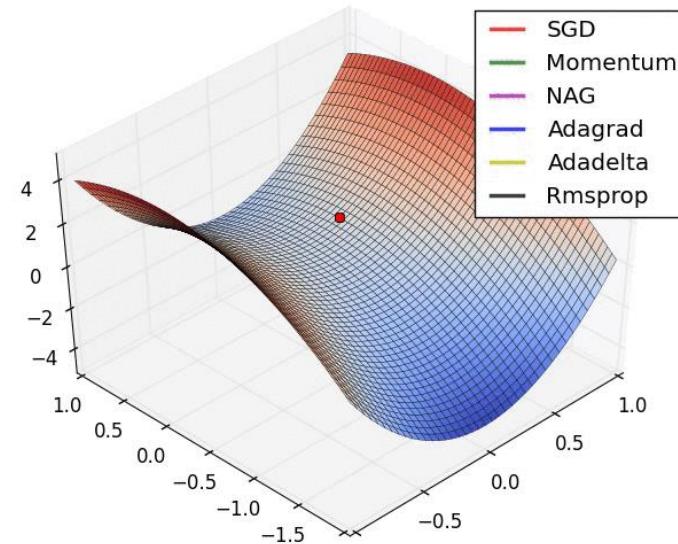
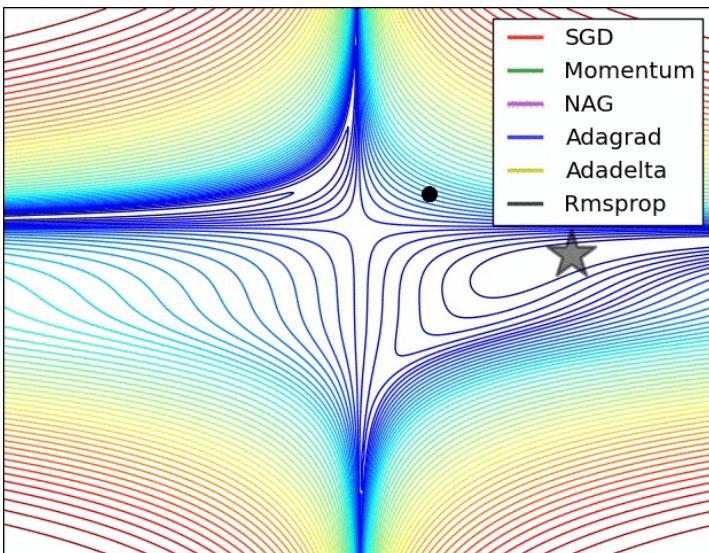


Stochastic Gradient Descent

- Calculate with mini-batch data
- Forward Faster
- A bit fluctuated but faster
- e.g. 10 steps * 5 mins = 50 mins

3 Deep Learning for NLP

Finding the Optimal weight and bias – Gradient Descent



There are different types of Gradient descent optimization algorithms:
Batch Gradient Descent, Stochastic Gradient Descent, Momentum, Adam, etc.

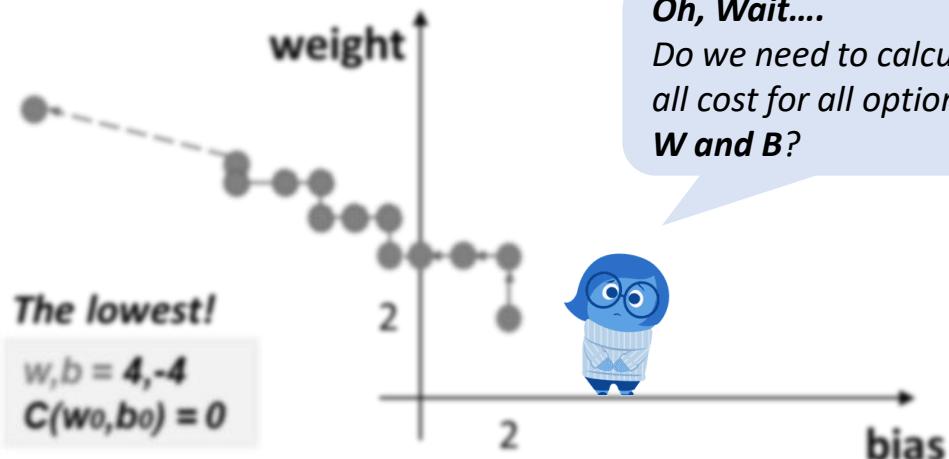
3 Deep Learning for NLP

Backpropagation (weight update)

Deep Learning with Neural Network - Optimizer

$$y = 4 \text{ weight} | x - 4 \text{ bias}$$

x 🍎	y 🍌
1	0
5	16
6	20



Expensive to compute
(hours or days)

$$\arg \min C(w, b)$$

$$w, b \in [-\infty, \infty]$$

3 Deep Learning for NLP

Deep Learning with Neural Network - Optimizer

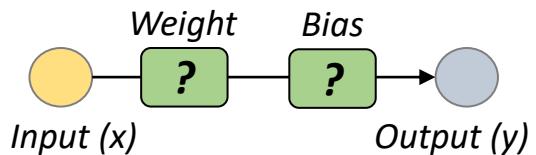
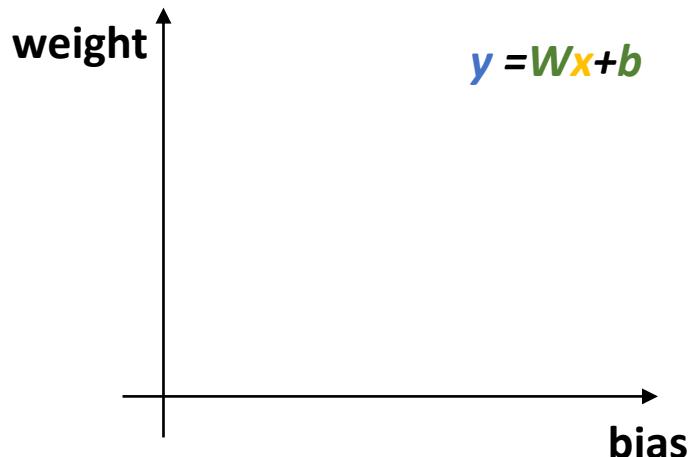
Gradient!

Actual Data

$$y = ? \times x + ?$$

weight bias

x	y
1	0
5	16
6	20



Should be used sparingly

$$\arg \min C(w, b)$$

$$w, b \in [-\infty, \infty]$$

3 Deep Learning for NLP

Deep Learning with Neural Network - Optimizer

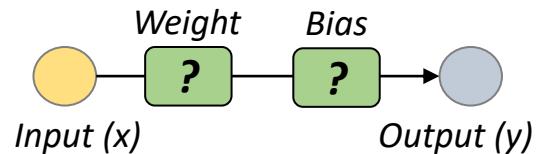
Gradient!

Actual Data

$$y = ? \times x + ?$$

weight bias

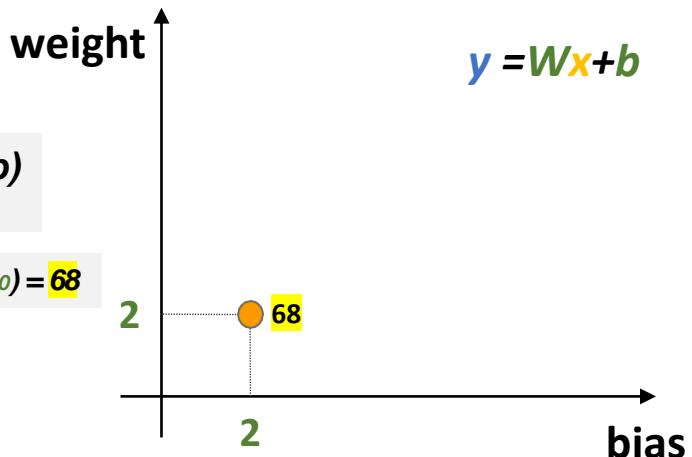
x	y
1	0
5	16
6	20



$$\arg \min C(w,b)$$

$$w,b \in [-\infty, \infty]$$

$w,b = 2,2 \quad C(w_0,b_0) = 68$



3 Deep Learning for NLP

Deep Learning with Neural Network - Optimizer

Gradient!

Actual Data

$$y = ? \times x + ?$$

weight bias

x 🍎	y 🍌
1	0
5	16
6	20

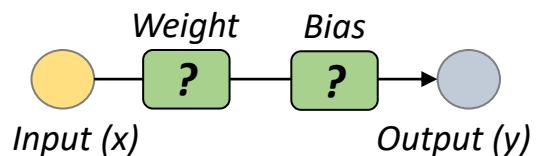
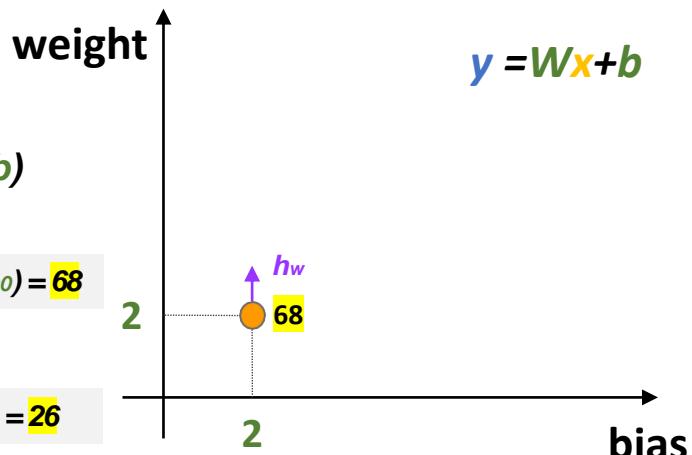
$$\arg \min_{w,b} C(w,b)$$

$$w,b \in [-\infty, \infty]$$

$$w,b = 2,2 \quad C(w_0,b_0) = 68$$

$$h_w = 1$$

$$C(w_0+h_w,b_0) = C(3,2) = 26$$



3 Deep Learning for NLP

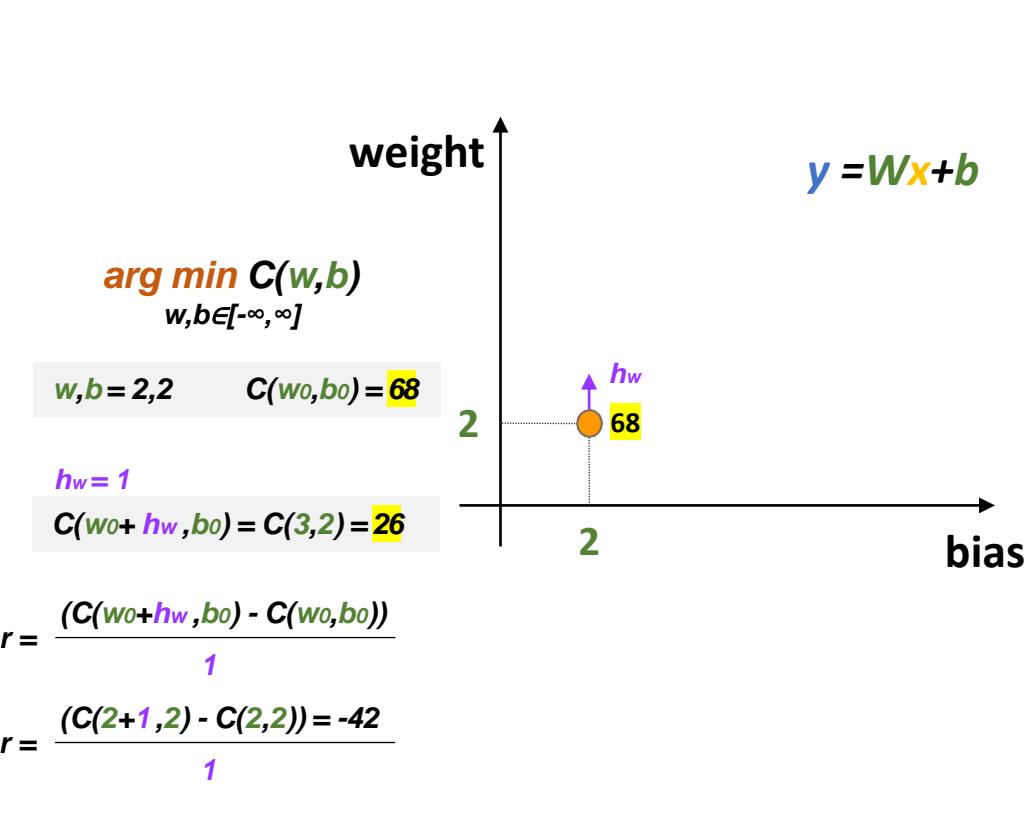
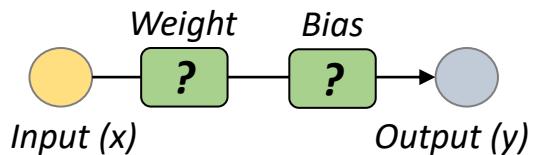
Deep Learning with Neural Network - Optimizer

Gradient!

Actual Data

$$y = ? \boxed{x} + ?$$

x 🍎	y 🍌
1	0
5	16
6	20



3 Deep Learning for NLP

Deep Learning with Neural Network - Optimizer

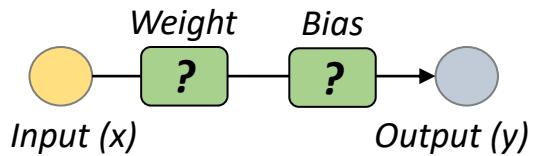
Gradient!

Actual Data

$$y = ? \times x + ?$$

weight bias

x	y
1	0
5	16
6	20

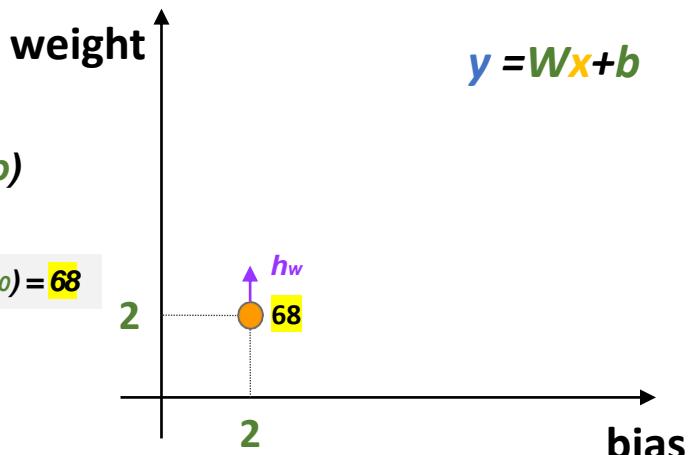


$$\arg \min_{w,b} C(w,b)$$

$$w,b \in [-\infty, \infty]$$

$$w,b = 2,2 \quad C(w_0,b_0) = 68$$

$$\begin{aligned} h_w &= 1, r = -42 \\ h_w &= 0.1, r = -98 \\ h_w &= 0.01, r = -104 \\ h_w &= 0.001, r = -104 \end{aligned}$$



3 Deep Learning for NLP

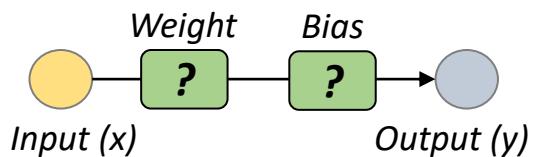
Deep Learning with Neural Network - Optimizer

Gradient!

Actual Data

$$y = ? \boxed{x} + ?$$

x 🍎	y 🍌
1	0
5	16
6	20



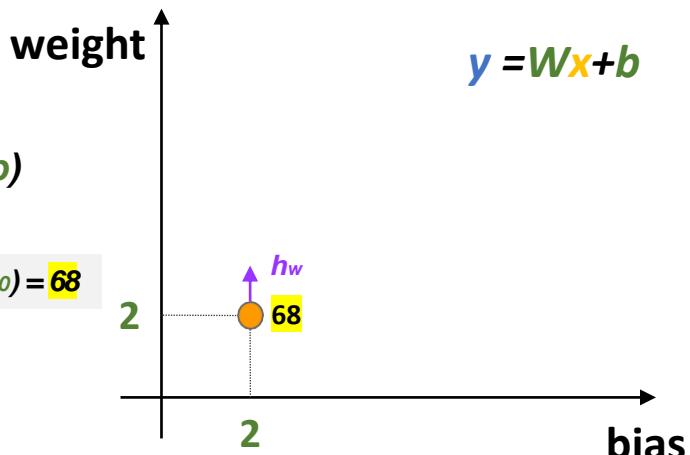
$$\arg \min_{w,b} C(w,b)$$

$$w,b \in [-\infty, \infty]$$

$$w,b = 2,2 \quad C(w_0,b_0) = 68$$

$$\begin{aligned}
 h_w &= 1, r = -42 \\
 h_w &= 0.1, r = -98 \\
 h_w &= 0.01, r = -104 \\
 h_w &= 0.001, r = -104
 \end{aligned}$$

$$h_w \rightarrow 0, \quad r = \frac{\partial C}{\partial w}(w_0,b_0)$$



3 Deep Learning for NLP

Deep Learning with Neural Network - Optimizer

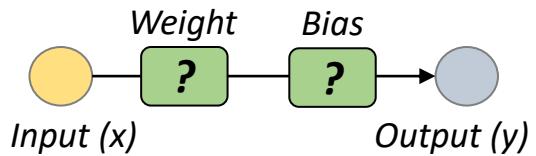
Gradient!

Actual Data

$$y = ? \times x + ?$$

weight bias

x	y
1	0
5	16
6	20

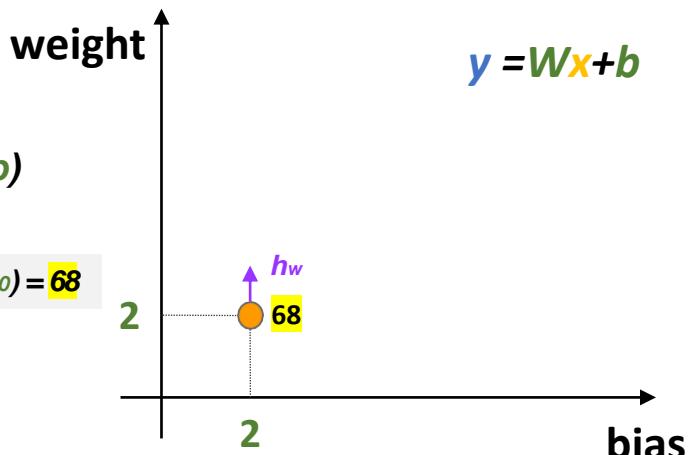


$$\arg \min_{w,b} C(w,b)$$

$$w,b \in [-\infty, \infty]$$

$$w,b = 2,2 \quad C(w_0,b_0) = 68$$

$$\frac{\partial C}{\partial w} = \frac{\partial \sum_n (y_n - \hat{y}_n)^2}{\partial w}$$



3 Deep Learning for NLP

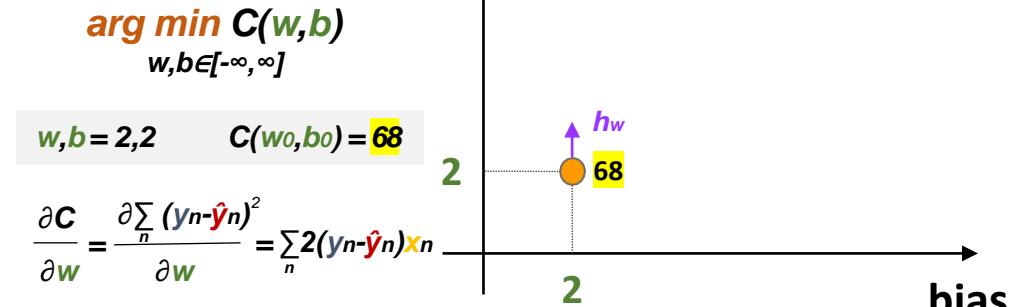
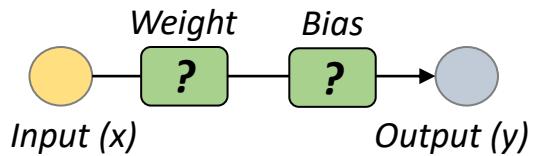
Gradient!

Deep Learning with Neural Network - Optimizer

Actual Data

$$y = ? \times x + ?$$

x	y
1	0
5	16
6	20



3 Deep Learning for NLP

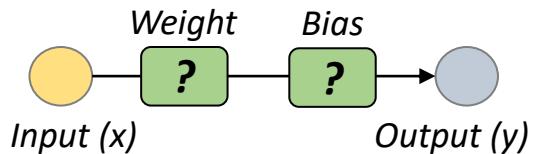
Gradient!

Deep Learning with Neural Network - Optimizer

Actual Data

$$y = ? \times x + ?$$

x	y
1	0
5	16
6	20



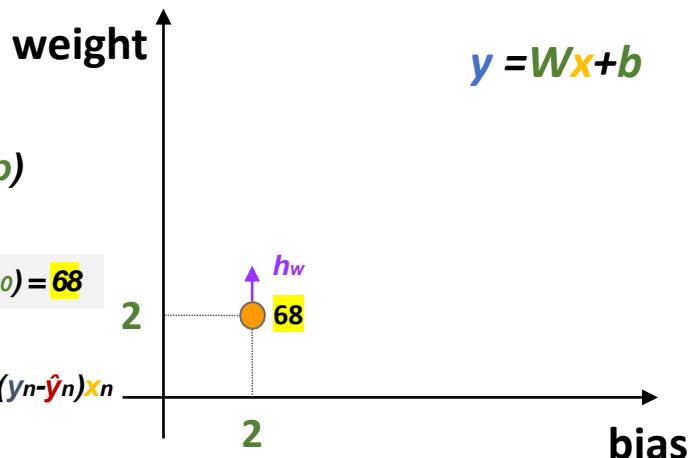
$$\arg \min_{w,b} C(w,b)$$

$$w,b \in [-\infty, \infty]$$

$$w,b = 2,2 \quad C(w_0,b_0) = 68$$

$$\frac{\partial C}{\partial w} = \frac{\partial \sum_n (y_n - \hat{y}_n)^2}{\partial w} = \sum_n 2(y_n - \hat{y}_n)x_n$$

$$h_w \rightarrow 0, r = \frac{\partial C}{\partial w}(w_0,b_0)$$



3 Deep Learning for NLP

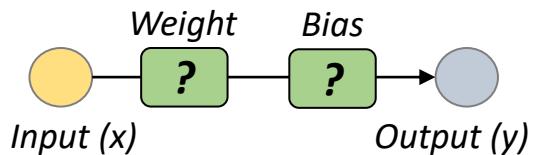
Deep Learning with Neural Network - Optimizer

Gradient!

Actual Data

$$y = ? \boxed{x} + ?$$

x 🍎	y 🍌
1	0
5	16
6	20



$$\arg \min_{w,b \in [-\infty, \infty]} C(w,b)$$

$$w,b = 2,2 \quad C(w_0, b_0) = 68$$

$$\frac{\partial C}{\partial w} = \frac{\partial \sum_n (y_n - \hat{y}_n)^2}{\partial w} = \sum_n 2(y_n - \hat{y}_n)x_n$$

$$h_w \rightarrow 0, r = \frac{\partial C}{\partial w}(w_0, b_0) = -104$$

$$y = \boxed{2} \boxed{x} + \boxed{2}$$

	<i>predicted</i>	<i>actual</i>		
x 🍎	\hat{y} 🍚	y 🍌	$(y - \hat{y})$	$2(y - \hat{y})x$
1	4	0	-4	-8
5	12	16	4	40
6	14	20	6	72

3 Deep Learning for NLP

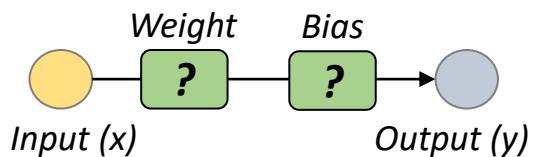
Gradient!

Deep Learning with Neural Network - Optimizer

Actual Data

$$y = ? \times x + ?$$

x	y
1	0
5	16
6	20



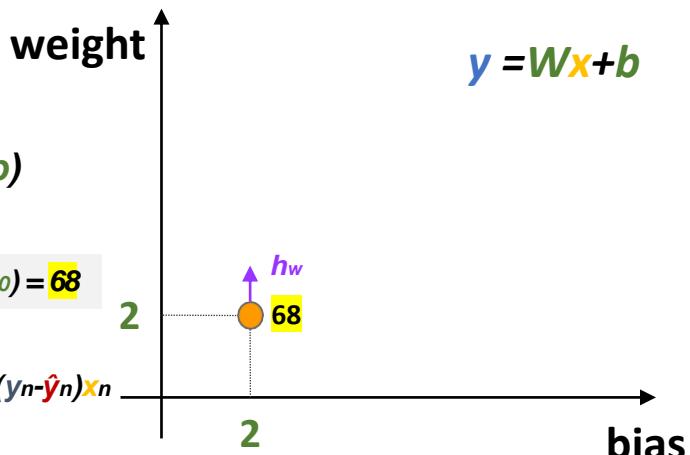
$$\arg \min_{w,b} C(w,b)$$

$$w,b \in [-\infty, \infty]$$

$$w,b = 2,2 \quad C(w_0,b_0) = 68$$

$$\frac{\partial C}{\partial w} = \frac{\partial \sum_n (y_n - \hat{y}_n)^2}{\partial w} = \sum_n 2(y_n - \hat{y}_n) x_n$$

$$\frac{\partial C}{\partial b} = \frac{\partial \sum_n (y_n - \hat{y}_n)^2}{\partial b} = \sum_n 2(y_n - \hat{y}_n)$$



3 Deep Learning for NLP

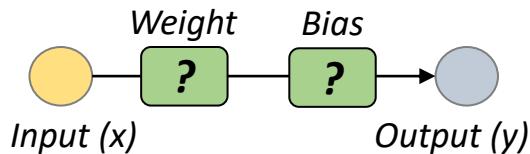
Deep Learning with Neural Network - Optimizer

Gradient!

Actual Data

$$y = ? \boxed{x} + ?$$

x 🍎	y 🍌
1	0
5	16
6	20



$$\arg \min_{w,b \in [-\infty, \infty]} C(w,b)$$

$$w,b=2,2 \quad C(w_0,b_0)=68$$

$$h_w \rightarrow 0, r = \frac{\partial C}{\partial w}(w_0,b_0) = -104$$

$$h_b \rightarrow 0, r = \frac{\partial C}{\partial b}(w_0,b_0) = -12$$

$$y = \boxed{2} \boxed{x} + \boxed{2}$$

	<i>predicted</i>	<i>actual</i>		
x 🍎	ŷ 🍚	y 🍌	(y-ŷ)	2(y-ŷ)
1	4	0	-4	-8
5	12	16	4	8
6	14	20	6	12

3 Deep Learning for NLP

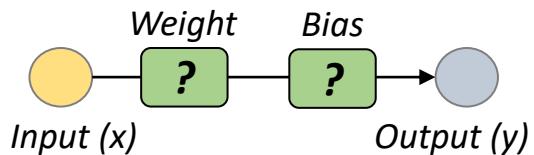
Deep Learning with Neural Network - Optimizer

Gradient!

Actual Data

$$y = ? \boxed{x} + ?$$

x 🍎	y 🍌
1	0
5	16
6	20



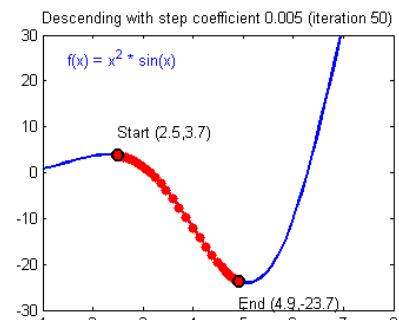
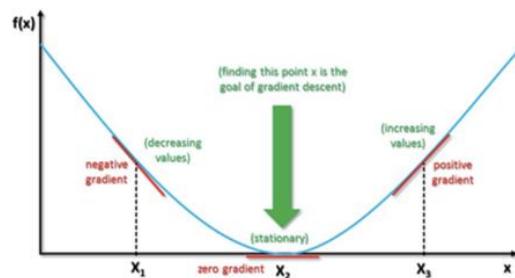
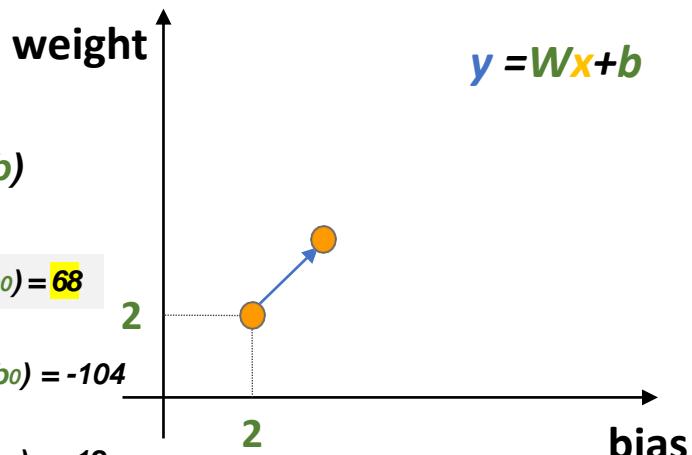
$$\arg \min C(w, b)$$

$$w, b \in [-\infty, \infty]$$

$$w, b = 2, 2 \quad C(w_0, b_0) = 68$$

$$h_w \rightarrow 0, r = \frac{\partial C}{\partial w} (w_0, b_0) = -104$$

$$h_b \rightarrow 0, r = \frac{\partial C}{\partial b} (w_0, b_0) = -12$$



Deep Learning with Neural Network

Data

x	y
1	0
5	16
6	20

Model

$$y = ? \underset{\text{weight}}{x} + ? \underset{\text{bias}}$$

Cost

$$C(w, b) = \sum (y_n - \hat{y}_n)$$

$n \in \{0, 1, 2\}$

Optimizer

$$\arg \min C(w, b)$$

$w, b \in [-\infty, \infty]$



System

$$y = \underset{\text{weight}}{4} \underset{x}{x} - \underset{\text{bias}}{4}$$

3 Deep Learning for NLP



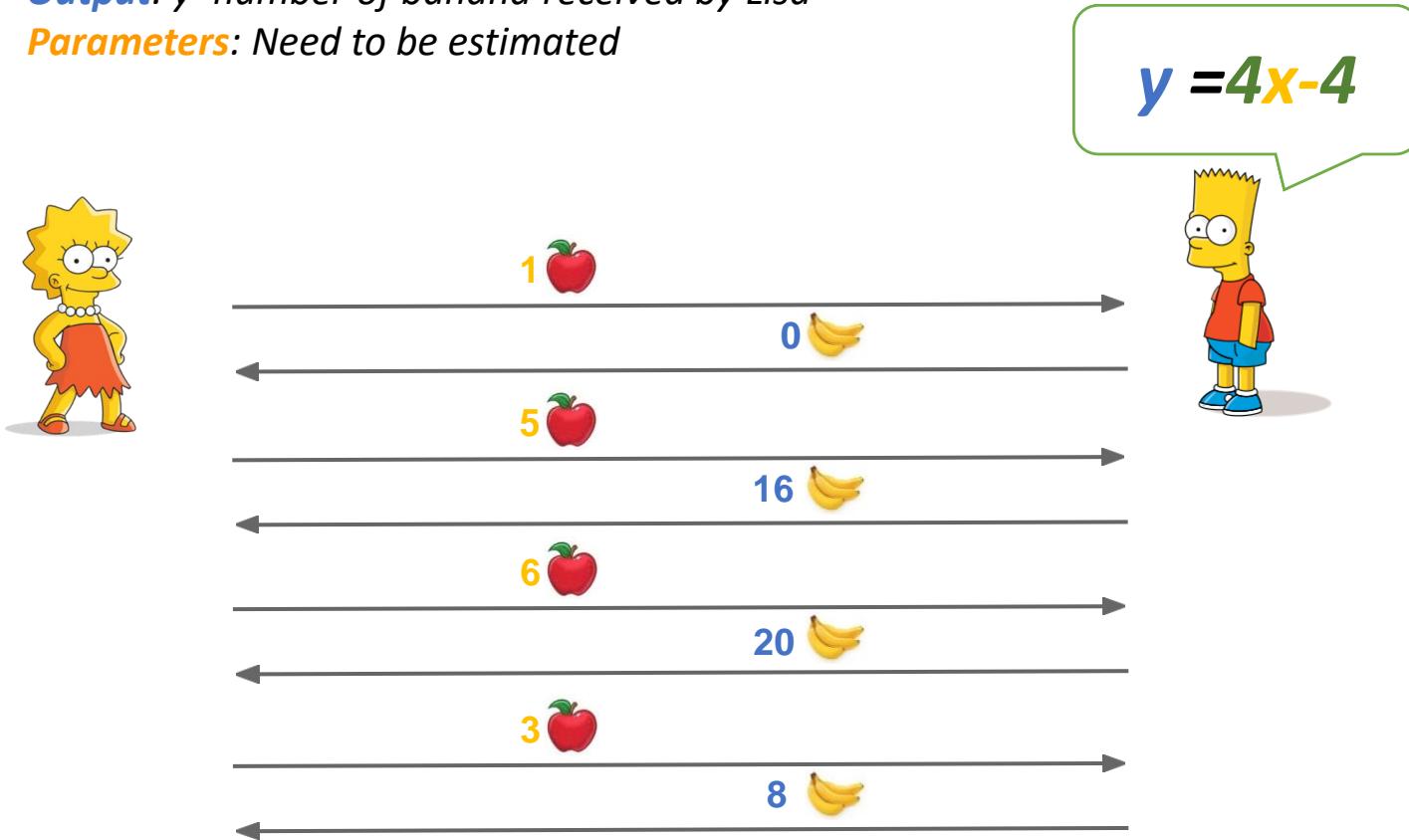
3 Deep Learning for NLP

Deep Learning with Neural Network

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

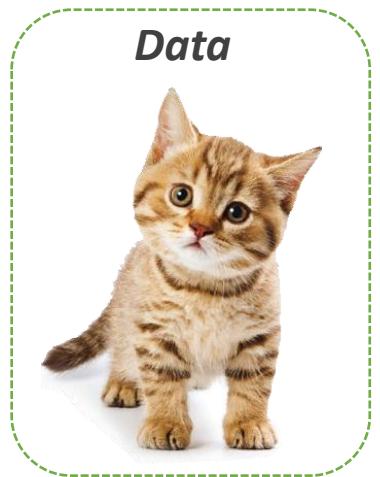
Parameters: Need to be estimated



3 Deep Learning for NLP

Deep Learning with Neural Network

$$y = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + \dots + w_nx_n + b$$



Millions of Parameters
Millions of Samples

3 Deep Learning for NLP

Deep Learning with Neural Network

$$y = W_1 \overset{\text{Vector1}}{X_1} + W_2 \overset{\text{Vector2}}{X_2} + W_3 X_3 + W_4 X_4 + \dots + W_n X_n + b$$



Millions of Parameters
Millions of Samples

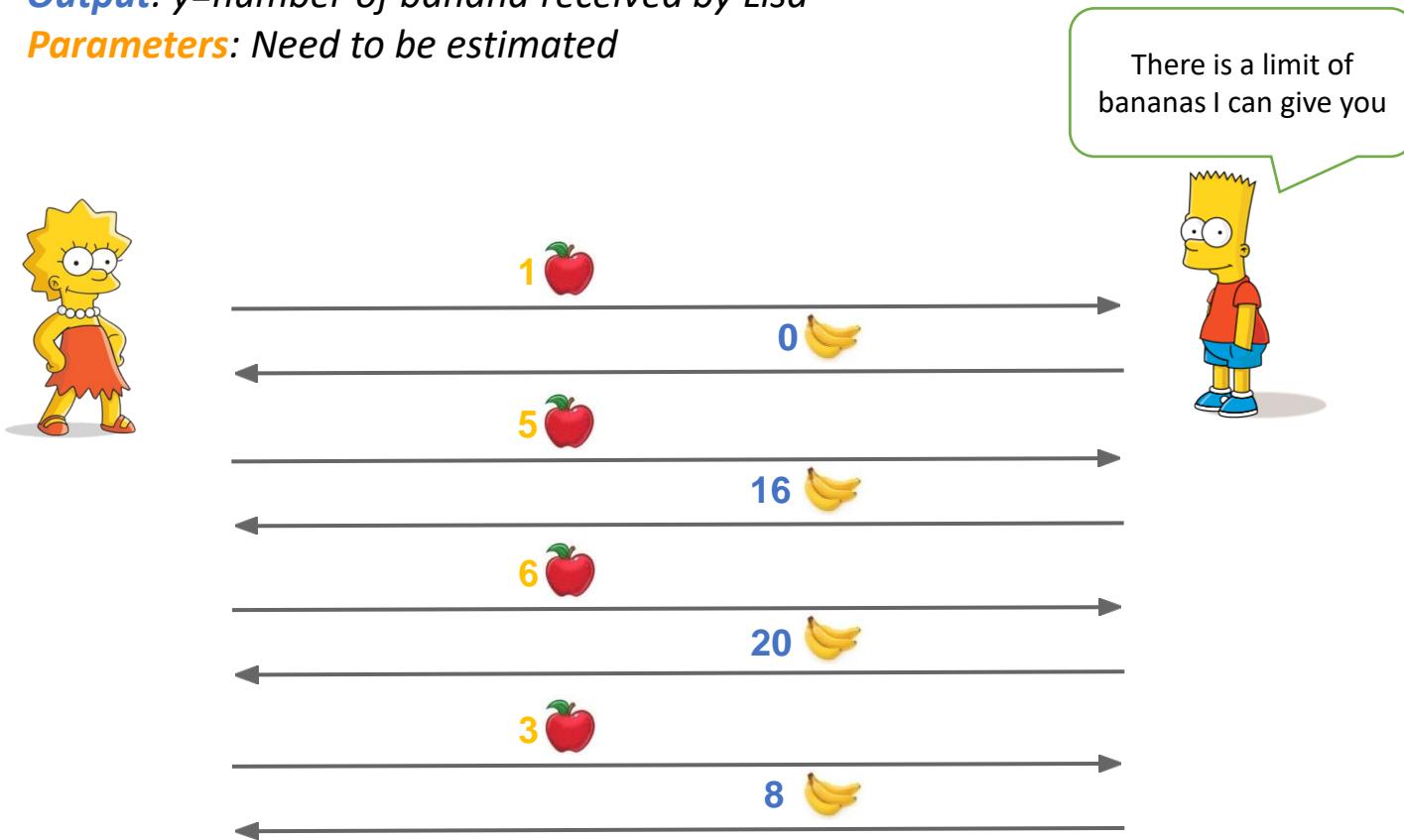
3 Deep Learning for NLP

Deep Learning with Neural Network

Input: x =number of apple given by Lisa

Output: y =number of banana received by Lisa

Parameters: Need to be estimated



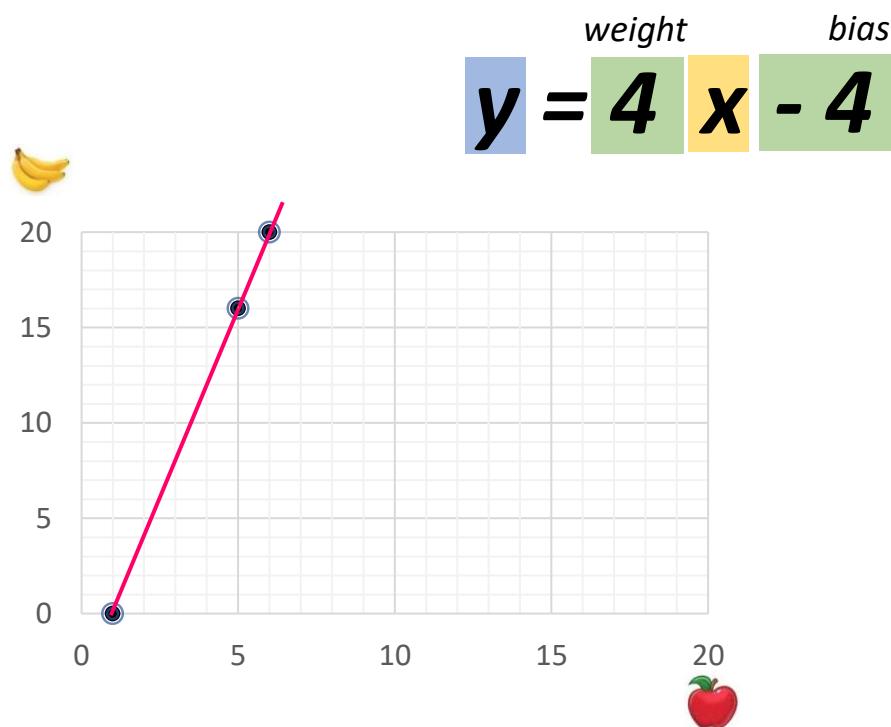
3 Deep Learning for NLP

Deep Learning with Neural Network

Nonlinear Neural Network

Data

x 🍎	y 🍌
1	0
5	16
6	20



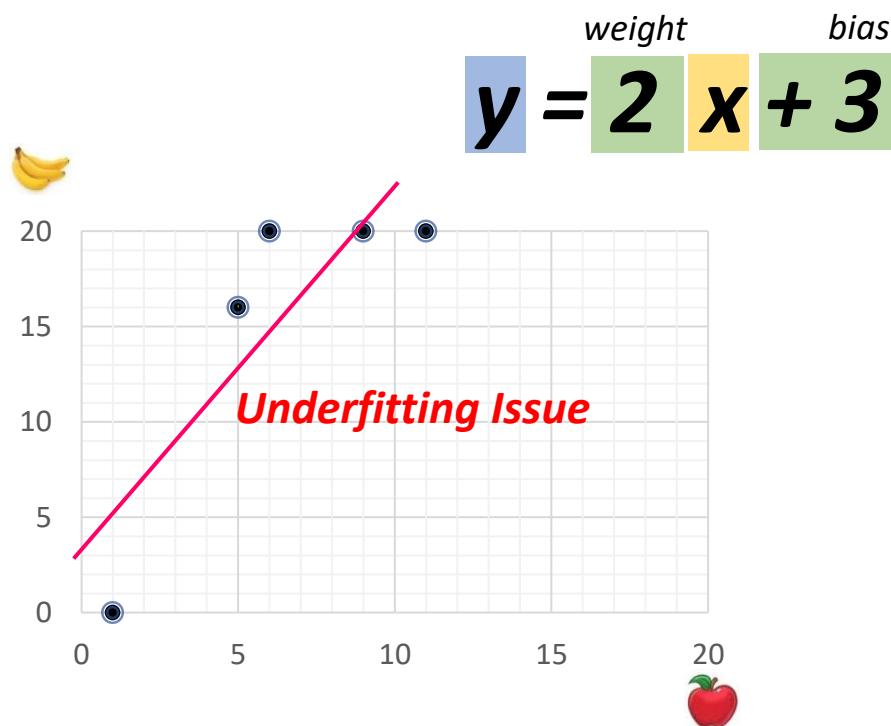
3 Deep Learning for NLP

Deep Learning with Neural Network

Nonlinear Neural Network

Data

x 🍎	y 🍌
1	0
5	16
6	20
9	20
11	20



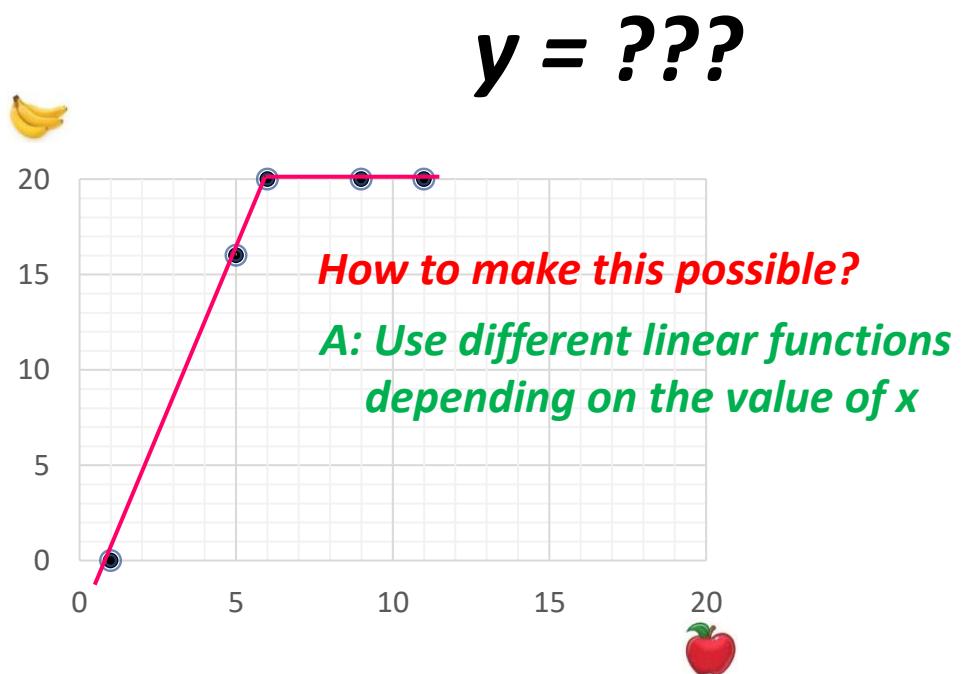
3 Deep Learning for NLP

Deep Learning with Neural Network

Nonlinear Neural Network

Data

x 🍎	y 🍌
1	0
5	16
6	20
9	20
11	20



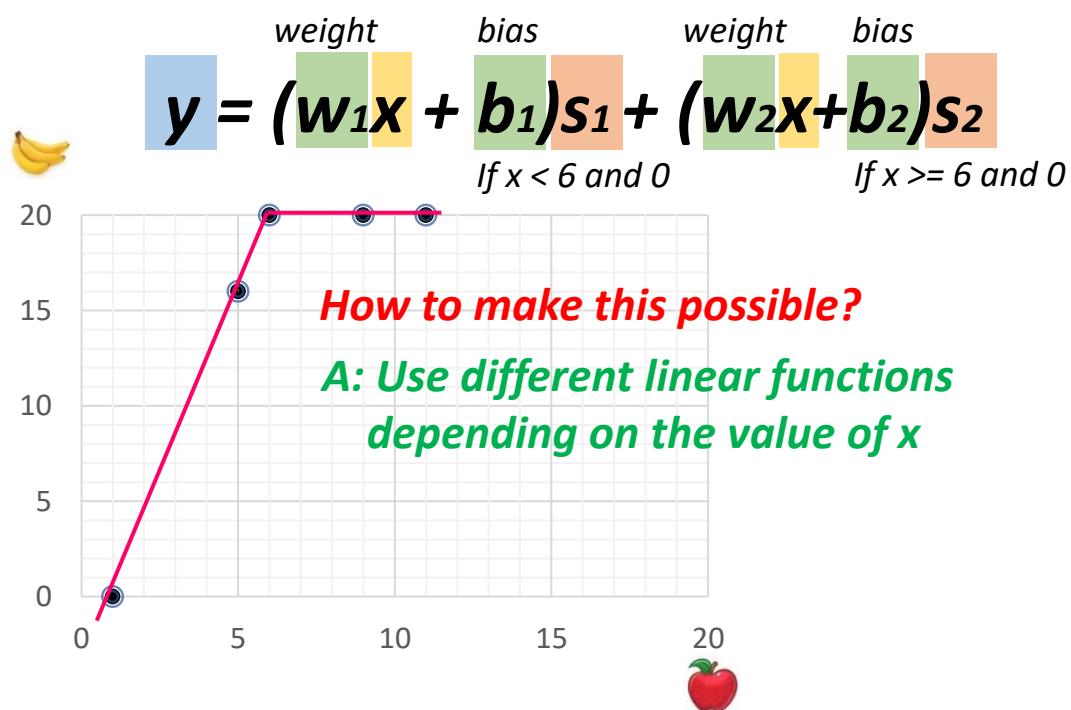
3 Deep Learning for NLP

Deep Learning with Neural Network

Nonlinear Neural Network

Data

x	y
1	0
5	16
6	20
9	20
11	20



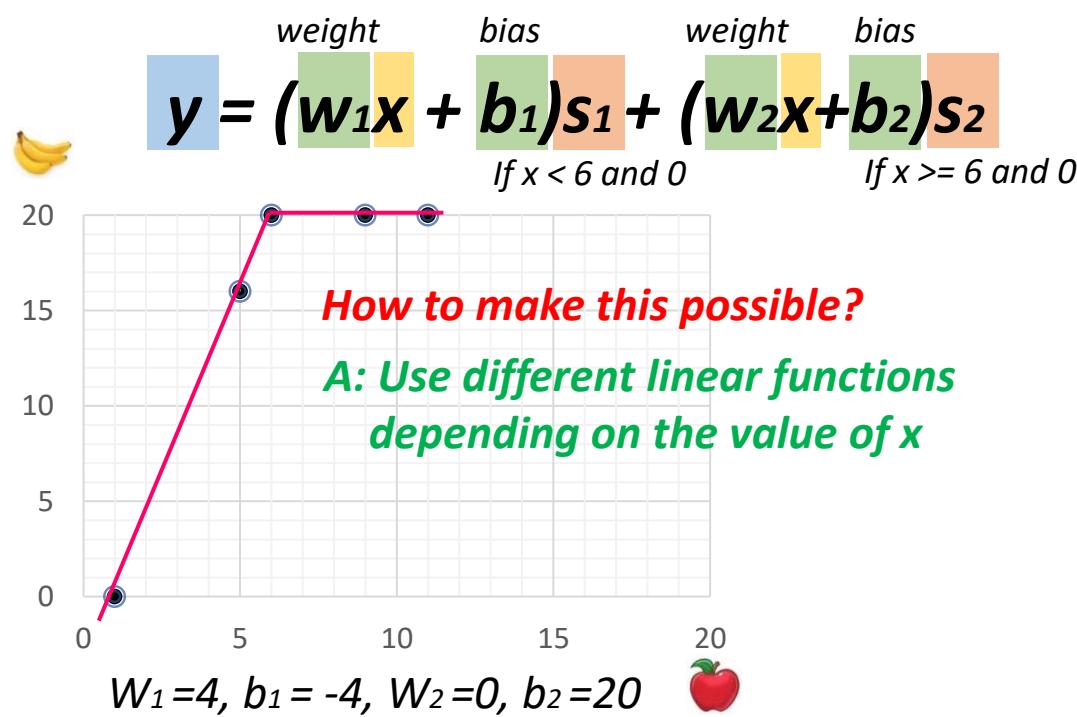
3 Deep Learning for NLP

Deep Learning with Neural Network

Nonlinear Neural Network

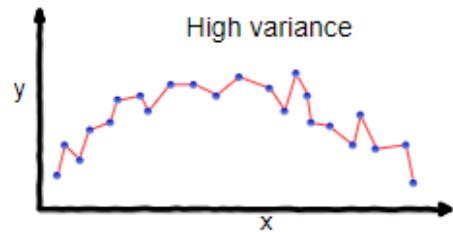
Data

x	y
1	0
5	16
6	20
9	20
11	20

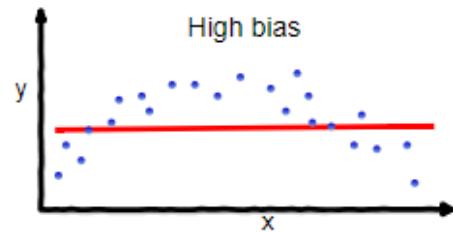


3 Deep Learning for NLP

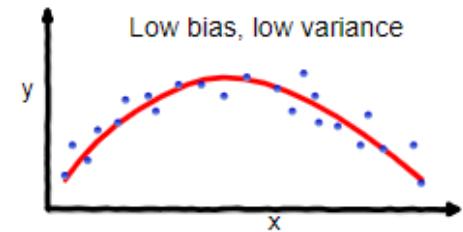
Deep Learning with Neural Network



overfitting

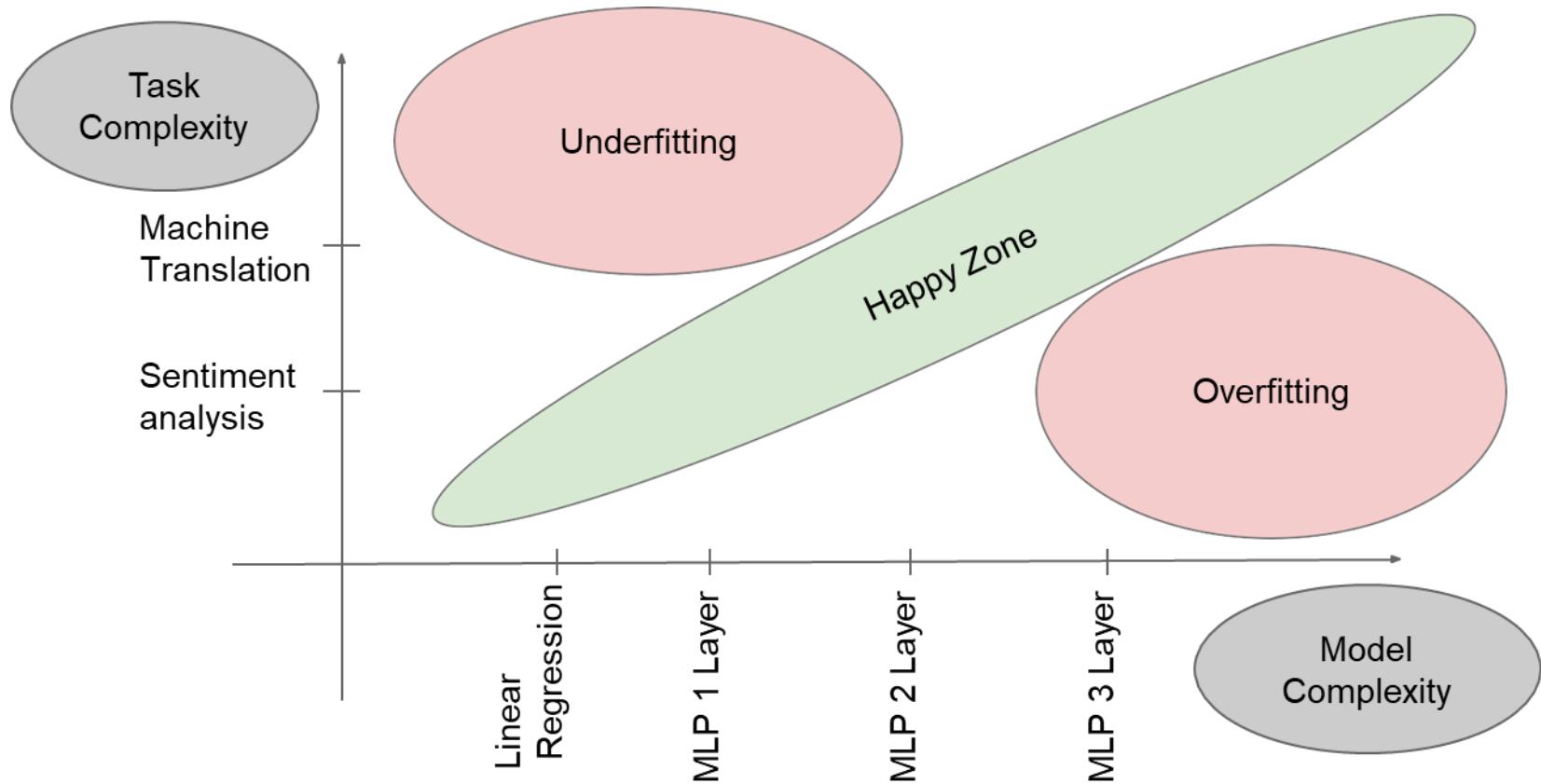


underfitting

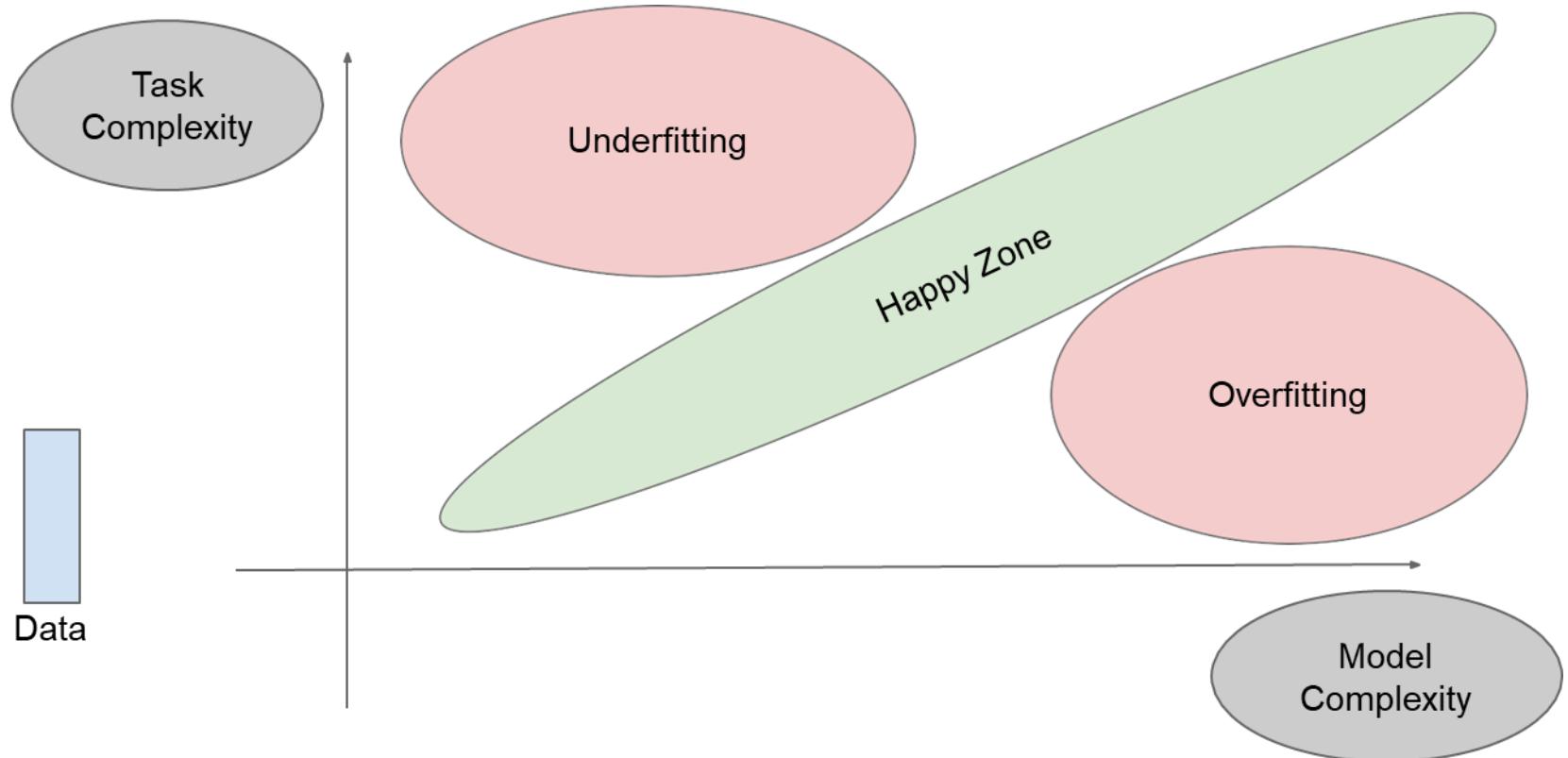


Good balance

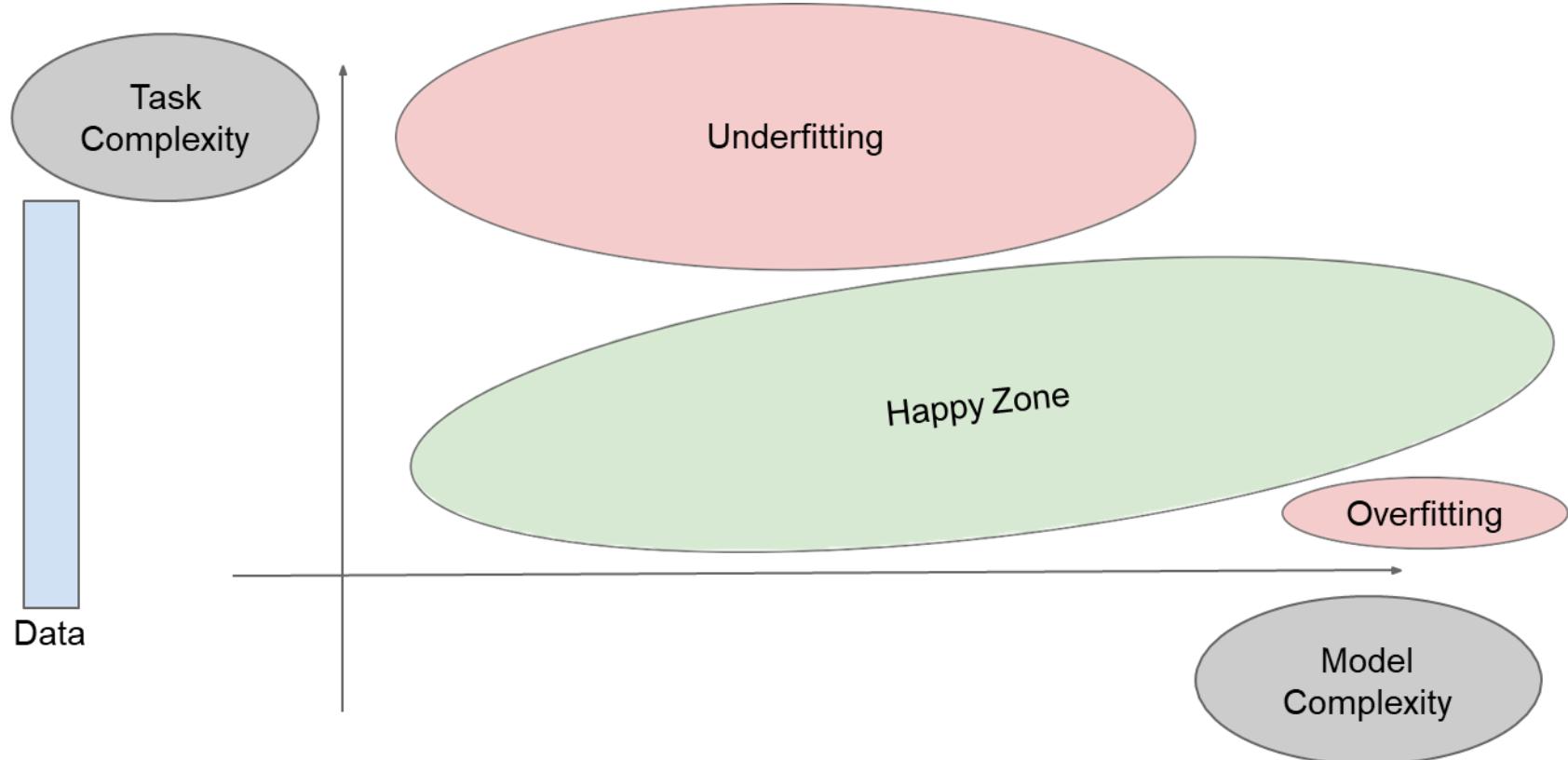
3 Deep Learning for NLP



3 Deep Learning for NLP

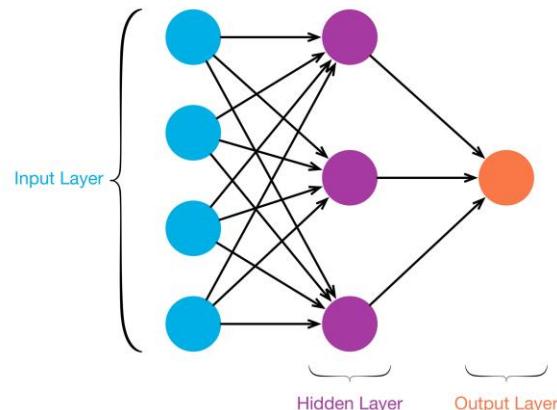
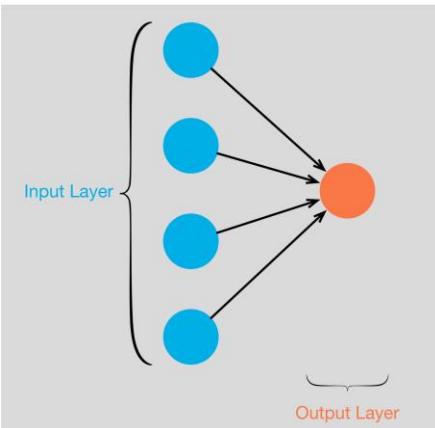
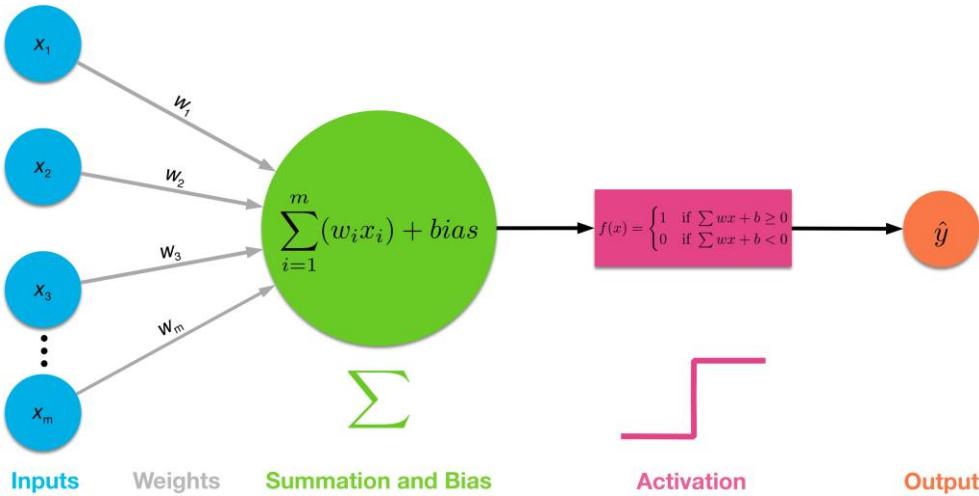


3 Deep Learning for NLP



3 Deep Learning for NLP

Single Neuron VS Multilayer

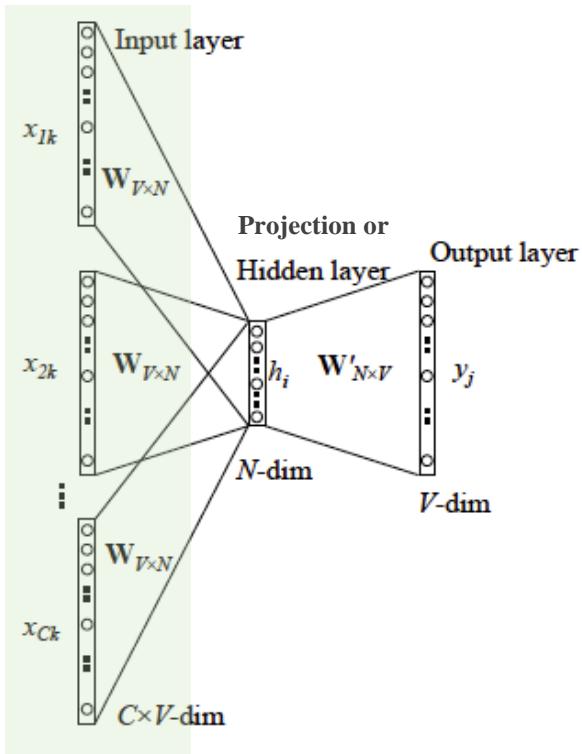


3 Deep Learning for NLP

CBOB – Neural Network Architecture (ReCAP with Optimizer)

Predict center word from (bag of) context words.

Summary of CBOB Training (Review your understanding with equations)



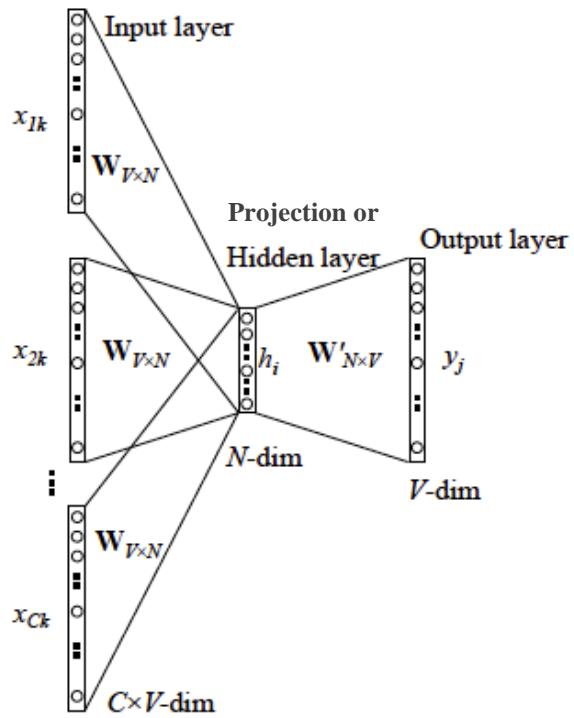
1. Initialise each word in a one-hot vector form.
 $x_k = [0, \dots, 0, 1, 0, \dots, 0]$
2. Use context words ($2m$, based on window size = m) as input of the Word2Vec-CBOW model.
 $(x^{c-m}, x^{c-m+1}, \dots, x^{c-1}, x^{c+1}, \dots, x^{c+m-1}, x^{c+m}) \in \mathbb{R}^{|V|}$
3. Has two Parameter Matrices:
 - 1) Parameter Matrix (from Input Layer to Hidden/Projection Layer)
 $\mathbf{W} \in \mathbb{R}^{V \times N}$
 - 2) Parameter Matrix (to Output Layer)
 $\mathbf{W}' \in \mathbb{R}^{N \times V}$

3 Deep Learning for NLP

CBOW – Neural Network Architecture (ReCAP with Optimizer)

Predict center word from (bag of) context words.

Summary of CBOW Training (Review your understanding with equations)



4. Initial words are represented in one hot vector so multiplying a **one hot vector** with $\mathbf{W}_{V \times N}$ will give you a $1 \times N$ (embedded word) vector.

$$(\mathbf{v}_{c-m} = \mathbf{W}\mathbf{x}^{c-m}, \dots, \mathbf{v}_{c+m} = \mathbf{W}\mathbf{x}^{c+m}) \in \mathbb{R}^n$$

5. Average those $2m$ embedded vectors to calculate the value of the Hidden Layer.

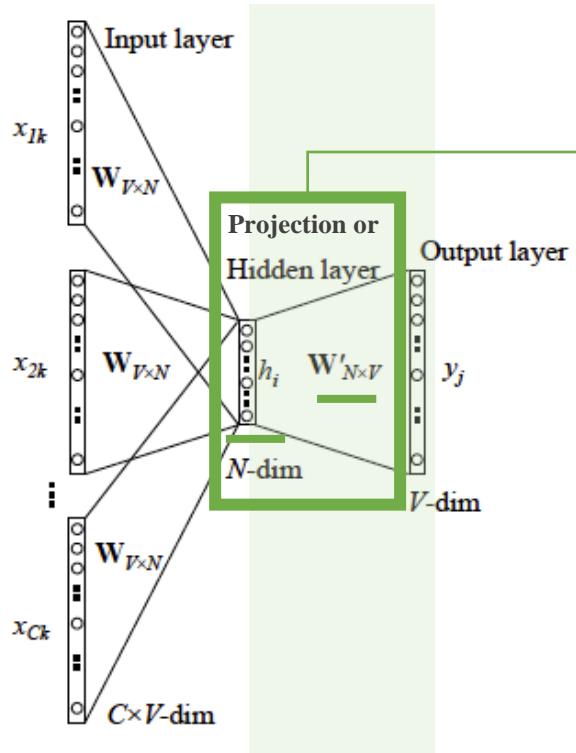
$$\hat{v} = \frac{\mathbf{v}_{c-m} + \mathbf{v}_{c-m+1} + \dots + \mathbf{v}_{c+m}}{2m}$$

3 Deep Learning for NLP

CBOV – Neural Network Architecture (ReCAP with Optimizer)

Predict center word from (bag of) context words.

Summary of CBOV Training (Review your understanding with equations)



6. Calculate the score value for the output layer. The higher score is produced when words are closer.
 $\mathbf{z} = \hat{\mathbf{v}}\mathbf{W}' \in \mathbb{R}^{|V|}$

7. Calculate the probability using softmax
 $\hat{y} = \text{softmax}(\mathbf{z}) \in \mathbb{R}^{|V|}$

8. Train the parameter matrix using objective function.

$$H(\hat{y}, y) = - \sum_{j=1}^{|V|} y_j \log(\hat{y}_j)$$

* Focus on minimising the value

We use an one-hot vector (one 1, the rest 0) so it will be calculated in only one.

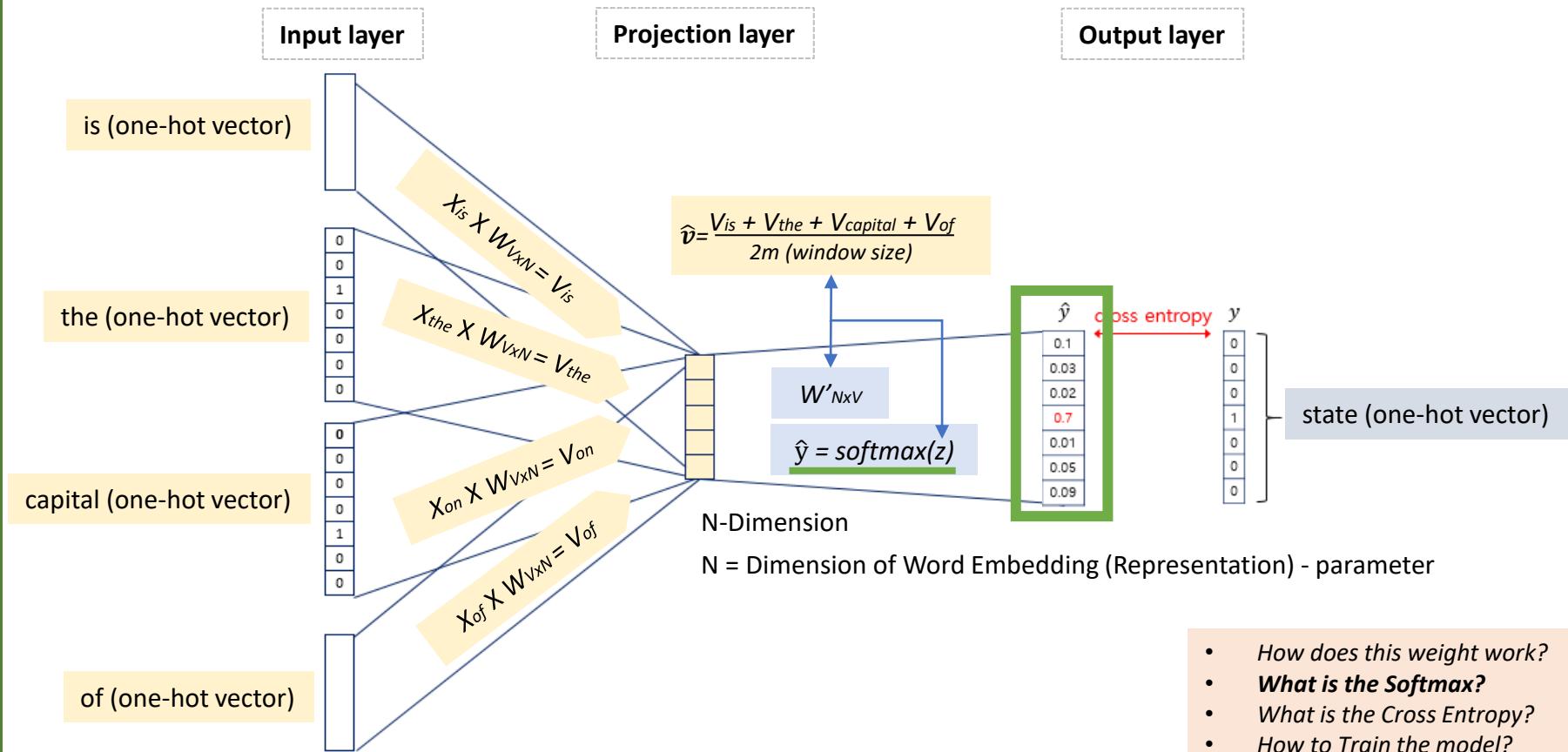
$$H(\hat{y}, y) = -y_j \log(\hat{y}_j)$$

3 Deep Learning for NLP

CBOW – Neural Network Architecture (ReCAP with Optimizer)

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”

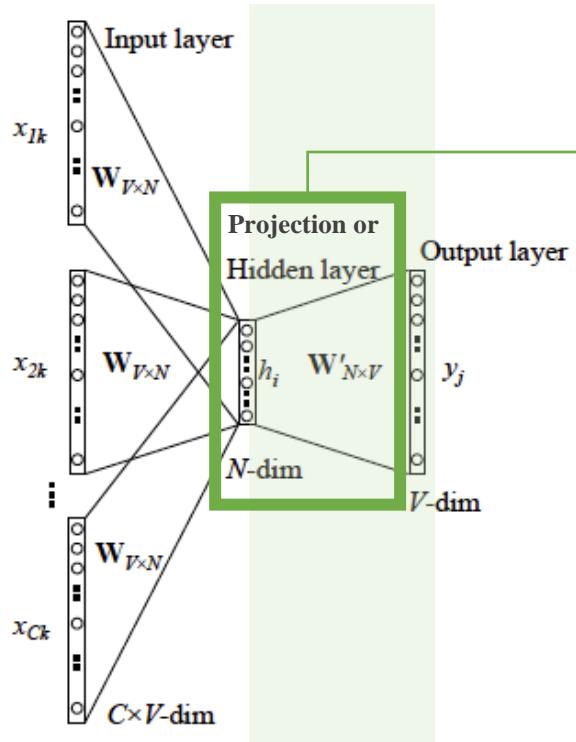


3 Deep Learning for NLP

CBOV – Neural Network Architecture (ReCAP with Optimizer)

Predict center word from (bag of) context words.

Summary of CBOV Training (Review your understanding with equations)



6. Calculate the score value for the output layer. The higher score is produced when words are closer.
 $\mathbf{z} = \hat{\mathbf{v}} \mathbf{W}' \in \mathbb{R}^{|V|}$

7. Calculate the probability using softmax
 $\hat{y} = \text{softmax}(\mathbf{z}) \in \mathbb{R}^{|V|}$

The softmax is an operator that will be used frequently. It transforms a vector into a vector whose i -th component is:

$$\frac{e^{\hat{y}_i}}{\sum_{j=1}^{|V|} e^{\hat{y}_j}}$$

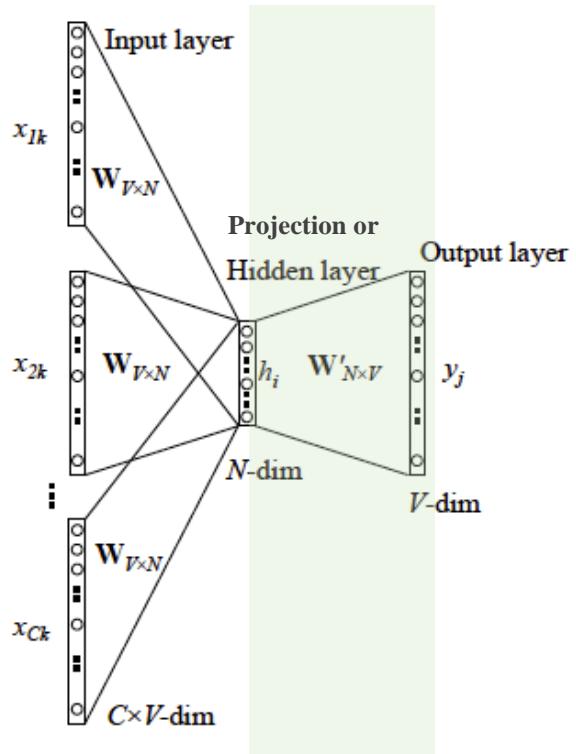
- Exponentiate to make positive
- Dividing by $\sum_{j=1}^{|V|} e^{\hat{y}_j}$ normalizes the vector ($\sum_{j=1}^n \hat{y}_j = 1$) to give probability

3 Deep Learning for NLP

CBOV – Neural Network Architecture (ReCAP with Optimizer)

Predict center word from (bag of) context words.

Summary of CBOV Training (Review your understanding with equations)



6. Calculate the score value for the output layer. The higher score is produced when words are closer.

$$\mathbf{z} = \hat{\mathbf{v}} \mathbf{W}' \in \mathbb{R}^{|V|}$$

7. Calculate the probability using softmax

$$\hat{y} = \text{softmax}(\mathbf{z}) \in \mathbb{R}^{|V|}$$

8. Train the parameter matrix using objective function.

$$H(\hat{y}, y) = - \sum_{j=1}^{|V|} y_j \log(\hat{y}_j) \quad \boxed{\text{Cross Entropy}}$$

* Focus on minimising the value

We use an one-hot vector (one 1, the rest 0) so it will be calculated in only one.

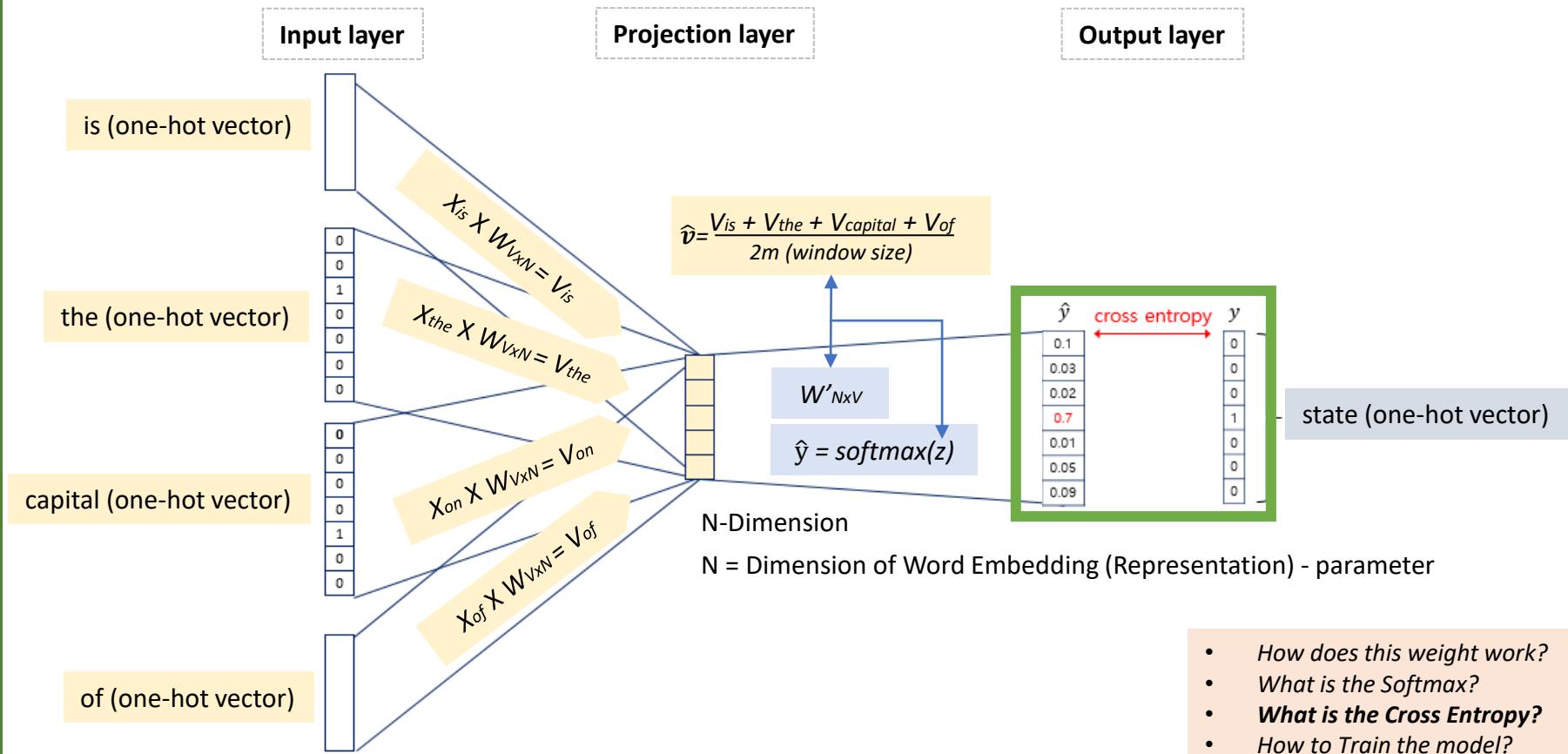
$$H(\hat{y}, y) = -y_j \log(\hat{y}_j)$$

3 Deep Learning for NLP

CBOV – Neural Network Architecture (ReCAP with Optimizer)

Predict center word from (bag of) context words

Sentence: “Sydney is the state capital of NSW”

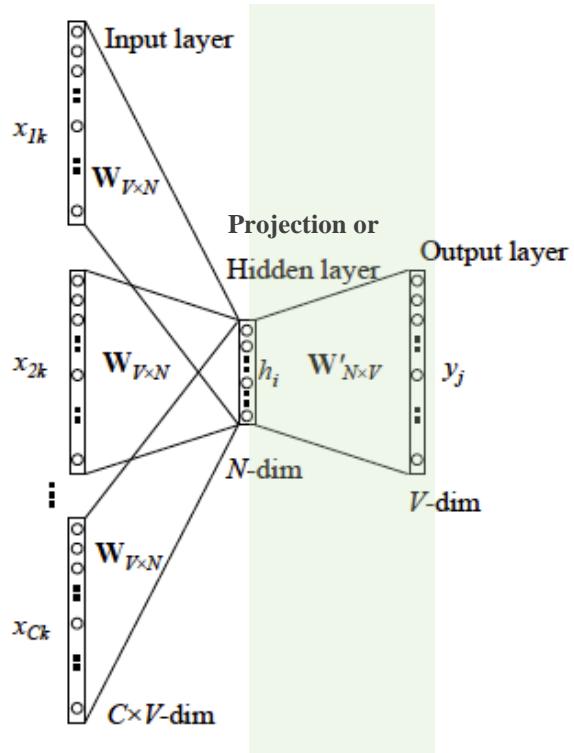


3 Deep Learning for NLP

CBOW – Neural Network Architecture (ReCAP with Optimizer)

Predict center word from (bag of) context words.

Summary of CBOW Training (Review your understanding with equations)



8-1. Optimization Objective Function can be presented:

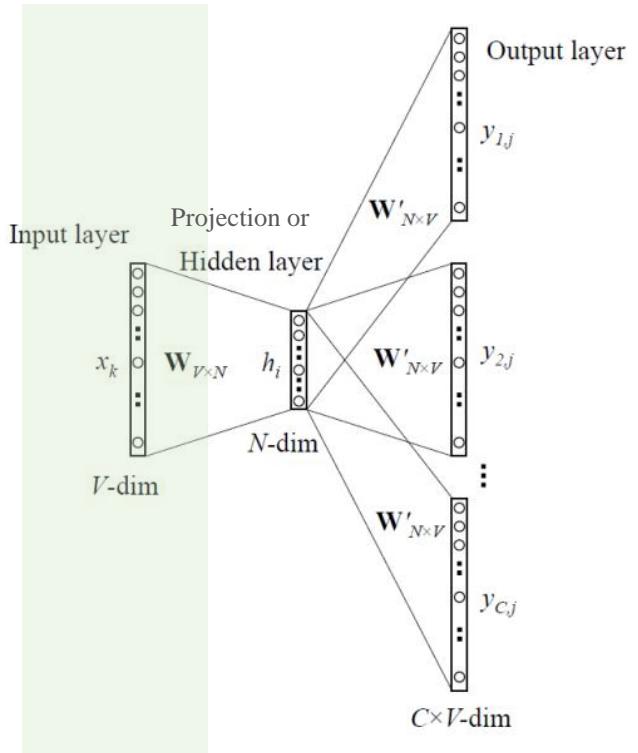
$$\begin{aligned}
 \text{minimize } J &= -\log P(w_c | w_{c-m}, \dots, w_{c-1}, w_{c+1}, \dots, w_{c+m}) \\
 &= -\log P(u_c | \hat{v}) \\
 &= -\log \frac{\exp(u_c^T \hat{v})}{\sum_{j=1}^{|V|} \exp(u_j^T \hat{v})} \\
 &= -u_c^T \hat{v} + \log \sum_{j=1}^{|V|} \exp(u_j^T \hat{v})
 \end{aligned}$$

u_i =the output vector representation of word w_i

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given center word

Summary of Skip Gram Training (Review your understanding with equations)



1. Initialise the centre word in a one-hot vector form.

$$\mathbf{x}_k = [0, \dots, 0, 1, 0, \dots, 0]$$

$$\mathbf{x} \in \mathbb{R}^{|V|}$$

2. Has two Parameter Matrices:

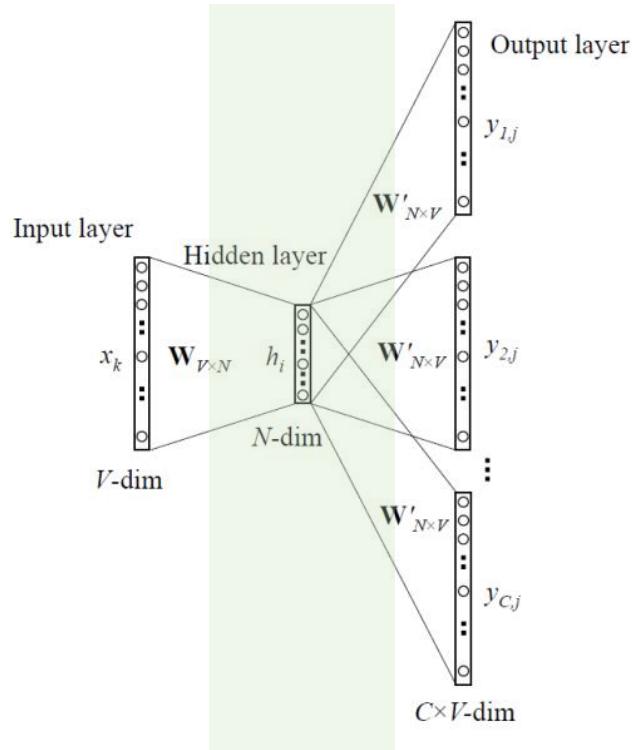
1) Parameter Matrix (from Input Layer to Hidden/Projection Layer)
 $\mathbf{W} \in \mathbb{R}^{V \times N}$

2) Parameter Matrix (to Output Layer)
 $\mathbf{W}' \in \mathbb{R}^{N \times V}$

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given center word

Summary of Skip Gram Training (Review your understanding with equations)



3. Initial words are represented in one hot vector so multiplying a **one hot vector** with $\mathbf{W}_{V \times N}$ will give you a $1 \times N$ (embedded word) vector.

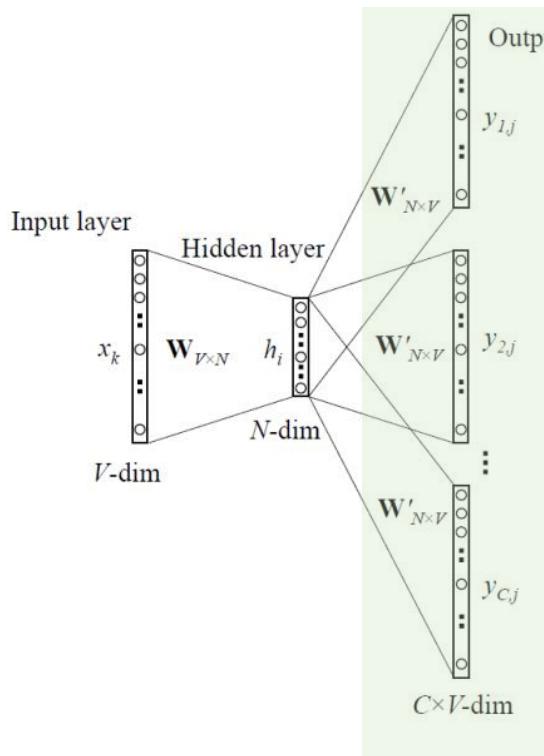
$$\mathbf{v}_c = \mathbf{W}_x \in \mathbb{R}^n \text{ (as there is only one input)}$$

4. Calculate the score value for the output layer by multiplying the parameter matrix \mathbf{W}'
 $\mathbf{z} = \mathbf{W}' \mathbf{v}_c$

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given center word

Summary of Skip Gram Training (Review your understanding with equations)



5. Calculate the probability using softmax
 $\hat{y} = \text{softmax}(\mathbf{z})$

6. Calculate $2m$ probabilities as we need to predict $2m$ context words.

$$\hat{y}_{c-m}, \dots, \hat{y}_{c-1}, \hat{y}_{c+1}, \dots, \hat{y}_{c+m}$$

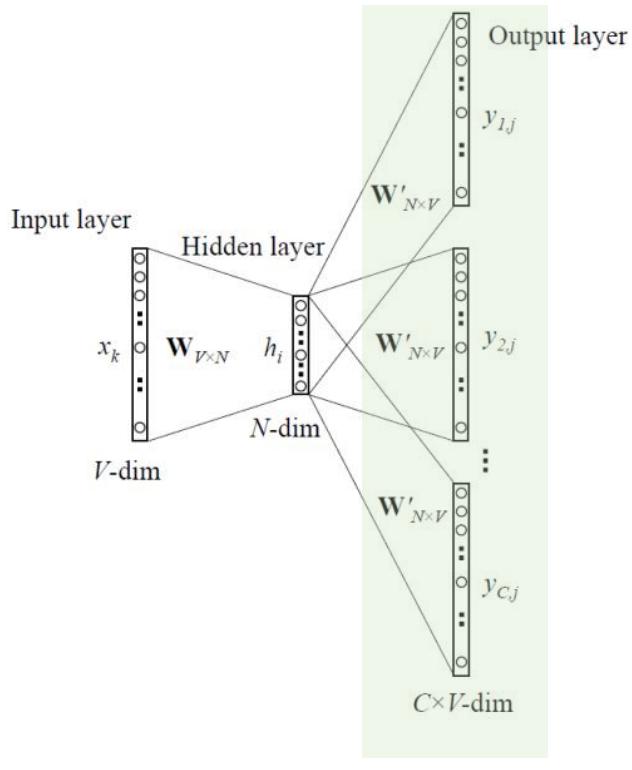
and compare with the ground truth (one-hot vector)
 $y^{(c-m)}, \dots, y^{(c-1)}, y^{(c+1)}, \dots, y^{(c+m)}$

Prediction based Word representation

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given center word

Summary of Skip Gram Training (Review your understanding with equations)



8. As in CBOW, use an objective function for us to evaluate the model. A key difference here is that we invoke a Naïve Bayes assumption to break out the probabilities. It is a strong naïve conditional independence assumption. Given the centre word, all output words are completely independent.

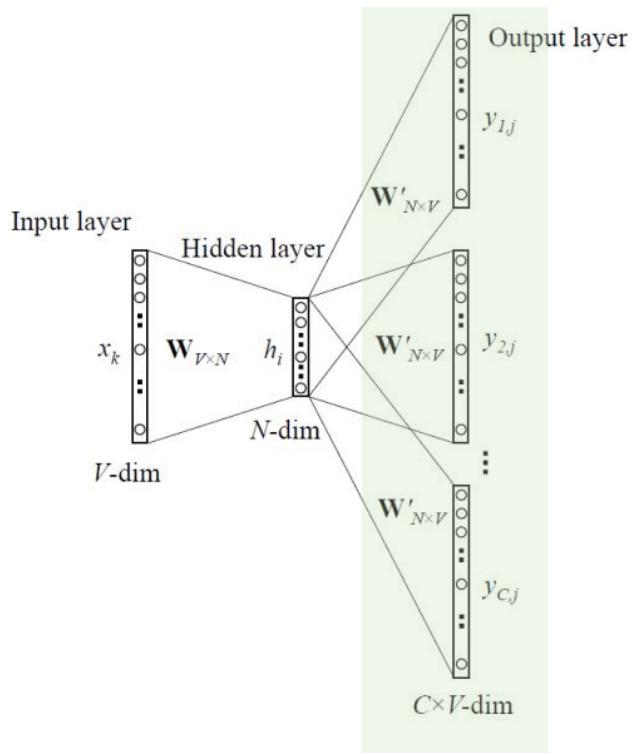
$$\begin{aligned}
 \text{minimize } J &= -\log P(w_{c-m}, \dots, w_{c-1}, w_{c+1}, \dots, w_{c+m} | w_c) \\
 &= -\log \prod_{j=0, j \neq m}^{2m} P(w_{c-m+j} | w_c) \\
 &= -\log \prod_{j=0, j \neq m}^{2m} \frac{\exp(u_{c-m+j}^\top v_c)}{\sum_{k=1}^{|V|} \exp(u_k^\top v_c)} \\
 &= - \sum_{j=0, j \neq m}^{2m} u_{c-m+j}^\top v_c + 2m \log \sum_{k=1}^{|V|} \exp(u_k^\top v_c)
 \end{aligned}$$

u_i =the output vector representation of word w_i

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given center word

Summary of Skip Gram Training (Review your understanding with equations)

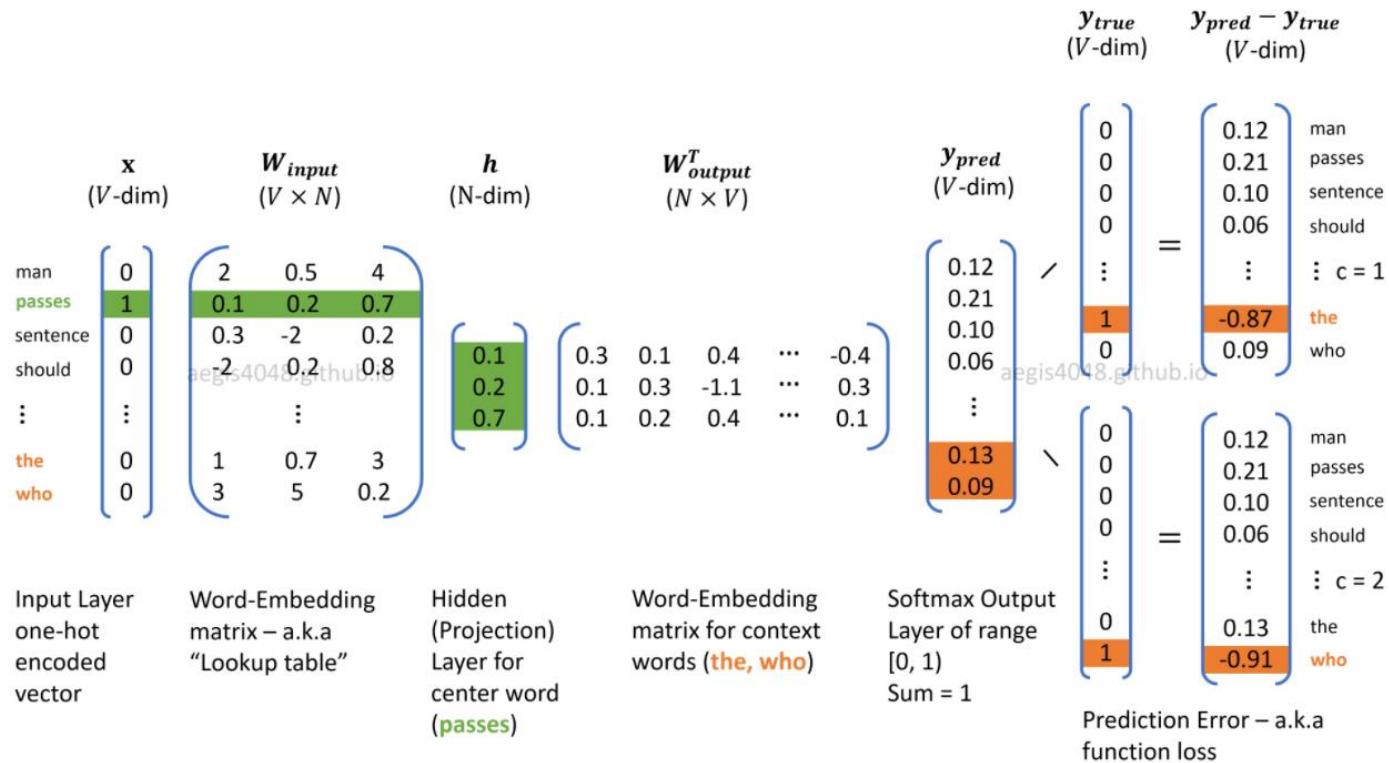


8-1. With this objective function, we can compute the gradients with respect to the unknown parameters and at each iteration update them via Stochastic Gradient Descent

$$\begin{aligned}
 J &= - \sum_{j=0, j \neq m}^{2m} \log P(u_{c-m+j} | v_c) \\
 &= \sum_{j=0, j \neq m}^{2m} H(\hat{y}, y_{c-m+j})
 \end{aligned}$$

Word2Vec-SkipGram Overview

With a simple diagram



Key Parameter (2) for Training methods: Negative Samples

The number of negative samples is another factor of the training process.

Negative samples to our dataset – samples of words that are not neighbors

Negative sample: 2

<i>Input word</i>	<i>Output word</i>	<i>Target</i>
eat	mango	1
eat	exam	0
eat	tobacco	0

*1=Appeared, 0=Not Appeared

Negative sample: 5

<i>Input word</i>	<i>Output word</i>	<i>Target</i>
eat	mango	1
eat	exam	0
eat	tobacco	0
eat	pool	0
eat	supervisor	0
eat	building	0

The original paper prescribes **5-20** as being a good number of negative samples. It also states that **2-5** seems to be enough when you have a large enough dataset.

4 Deep Learning for NLP

Word2Vec-SkipGram Overview – negative sampling

With a simple diagram

**Vanilla
Skip-Gram**

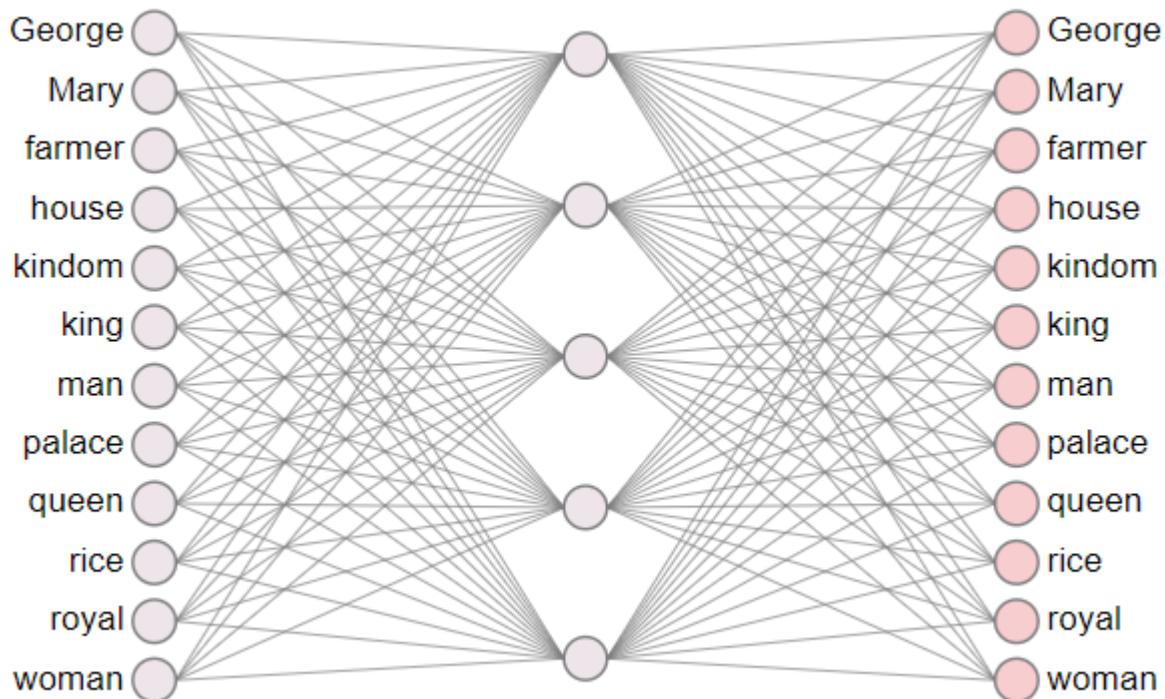
$$\begin{array}{c}
 \text{W_output (old)} \\
 \begin{array}{ccc} -0.560 & 0.340 & 0.160 \\ -0.910 & -0.440 & 1.560 \\ -1.210 & -0.130 & -1.320 \\ 1.670 & -0.150 & -1.030 \\ 1.720 & -1.460 & 0.730 \\ 0.000 & 1.390 & -0.120 \\ -0.060 & 1.520 & -0.790 \\ 0.800 & 1.850 & -1.670 \\ -1.370 & 1.320 & -0.480 \\ 0.670 & 1.990 & -1.850 \\ -1.520 & -1.740 & -1.860 \end{array} \\
 (11 \times 3)
 \end{array}
 \quad - \quad \boxed{0.05} \quad \times \quad
 \begin{array}{c}
 \text{grad_W_output} \\
 \begin{array}{ccc} 0.064 & 0.071 & -0.014 \\ 0.098 & 0.015 & 0.063 \\ 0.069 & 0.089 & 0.045 \\ 0.014 & 0.085 & 0.079 \\ -0.021 & 0.067 & 0.071 \\ -0.098 & -0.088 & 0.091 \\ -0.072 & -0.078 & -0.089 \\ 0.046 & -0.079 & -0.053 \\ -0.049 & -0.087 & 0.025 \\ -0.060 & 0.092 & 0.042 \\ 0.074 & 0.050 & 0.070 \end{array} \\
 (11 \times 3)
 \end{array}
 \quad = \quad
 \begin{array}{c}
 \text{W_output (new)} \\
 \begin{array}{ccc} -0.563 & 0.336 & 0.161 \\ -0.915 & -0.441 & 1.557 \\ -1.213 & -0.134 & -1.322 \\ 1.669 & -0.154 & -1.034 \\ 1.721 & -1.463 & 0.726 \\ 0.005 & 1.394 & -0.125 \\ -0.056 & 1.524 & -0.786 \\ 0.798 & 1.854 & -1.667 \\ -1.368 & 1.324 & -0.481 \\ 0.673 & 1.985 & -1.852 \\ -1.524 & -1.743 & -1.864 \end{array} \\
 (11 \times 3)
 \end{array}$$

**Negative
Sampling**

$$\begin{array}{c}
 \text{W_output (old)} \\
 \begin{array}{ccc} -0.560 & 0.340 & 0.160 \\ -0.910 & -0.440 & 1.560 \\ -1.210 & -0.130 & -1.320 \\ 1.670 & -0.150 & -1.030 \\ 1.720 & -1.460 & 0.730 \\ 0.000 & 1.390 & -0.120 \\ -0.060 & 1.520 & -0.790 \\ 0.800 & 1.850 & -1.670 \\ -1.370 & 1.320 & -0.480 \\ 0.670 & 1.990 & -1.850 \\ -1.520 & -1.740 & -1.860 \end{array} \\
 (11 \times 3)
 \end{array}
 \quad - \quad \boxed{0.05} \quad \times \quad
 \begin{array}{c}
 \text{grad_W_output} \\
 \begin{array}{c} \text{Not computed!} \end{array}
 \end{array}
 \quad = \quad
 \begin{array}{c}
 \text{W_output (new)} \\
 \begin{array}{ccc} -0.560 & 0.340 & 0.160 \\ -0.910 & -0.440 & 1.560 \\ -1.210 & -0.130 & -1.320 \\ 1.670 & -0.150 & -1.030 \\ 1.720 & -1.460 & 0.730 \\ 0.000 & 1.390 & -0.120 \\ -0.060 & 1.520 & -0.790 \\ \text{Positive sample, w_o} & \text{0.031} & \text{0.030} & \text{0.041} \\ \text{Negative sample, k=1} & -0.090 & 0.031 & -0.065 \\ \text{Negative sample, k=2} & 0.056 & 0.098 & -0.061 \\ \text{Negative sample, k=3} & 0.069 & 0.084 & -0.044 \end{array} \\
 (11 \times 3)
 \end{array}
 \quad (11 \times 3)$$

Application

Application #1: Embedding Pretraining



0 LECTURE PLAN

Lecture 3: Word Classification and Machine Learning

1. Previous Lecture: Word Embedding Review
2. Word Embedding Evaluation
3. Deep Neural Network for Natural Language Processing
 1. Perceptron and Neural Network (NN)
 2. Multilayer Perceptron
 3. Applications

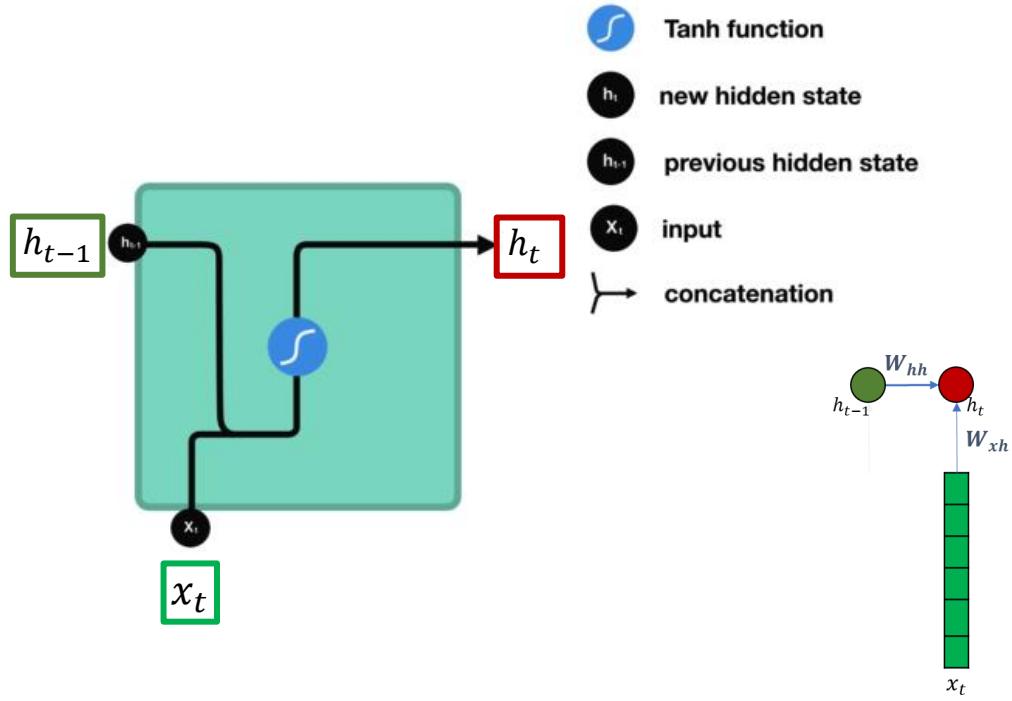
4. Next Week Preview

See how the Deep Learning can be used for NLP

- Text Classification, etc.

Hidden Layer

Input Layer



$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$

New hidden state A function
 Previous state with parameters W
 input

/ Reference

Reference for this lecture

- Deng, L., & Liu, Y. (Eds.). (2018). Deep Learning in Natural Language Processing. Springer.
- Rao, D., & McMahan, B. (2019). Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. " O'Reilly Media, Inc.".
- Manning, C. D., Manning, C. D., & Schütze, H. (1999). Foundations of statistical natural language processing. MIT press.
- Blunsom, P 2017, Deep Natural Language Processing, lecture notes, Oxford University
- Manning, C 2017, Natural Language Processing with Deep Learning, lecture notes, Stanford University

The lecture will be started at 5:05PM sharply!

COMP5046

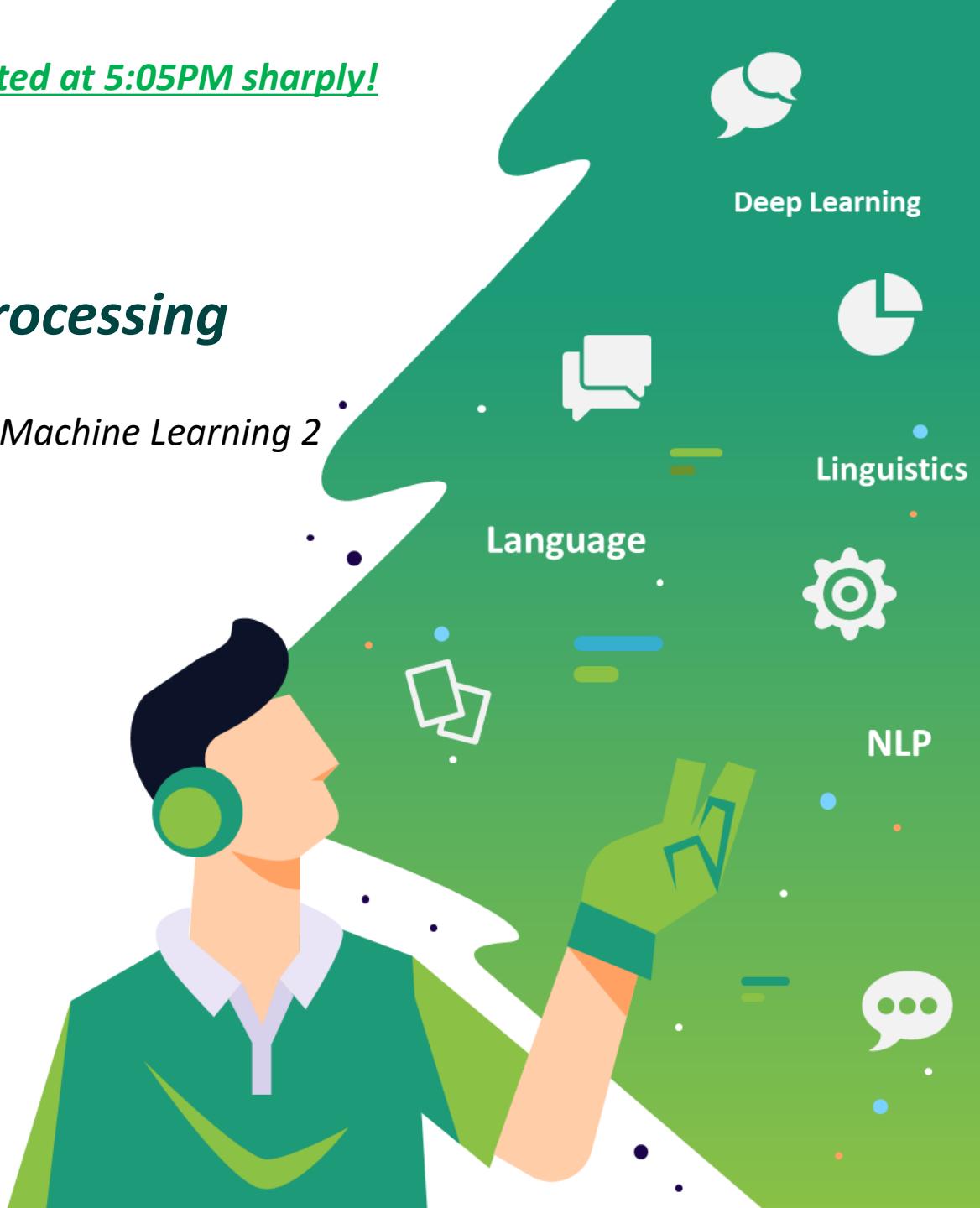
Natural Language Processing

Lecture 4: Word Classification and Machine Learning 2

Dr. Caren Han

Semester 1, 2022

*School of Computer Science,
University of Sydney*

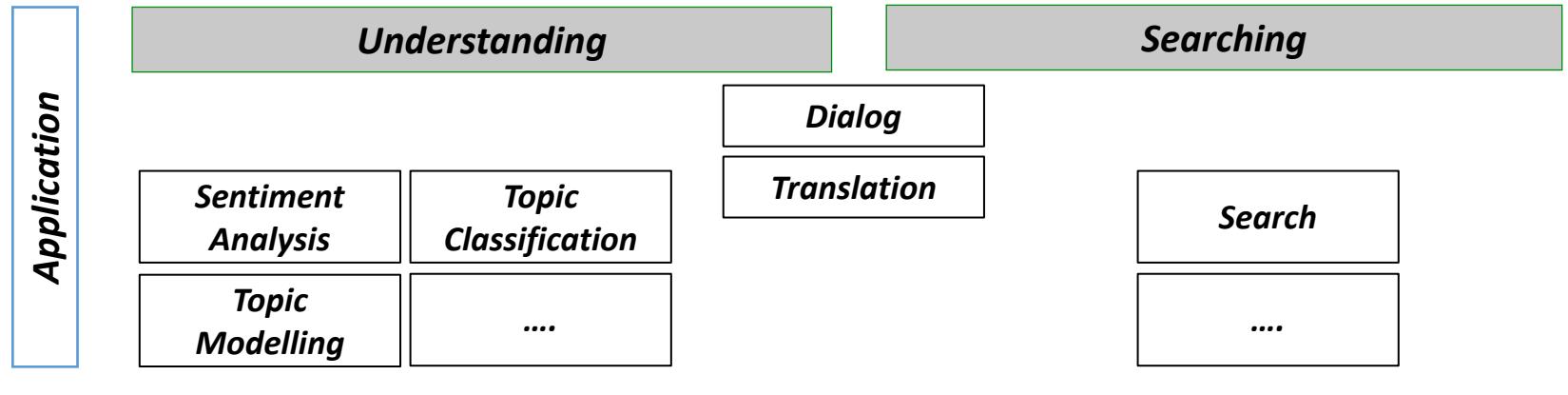


0 LECTURE PLAN

Lecture 4: Word Classification and Machine Learning 2

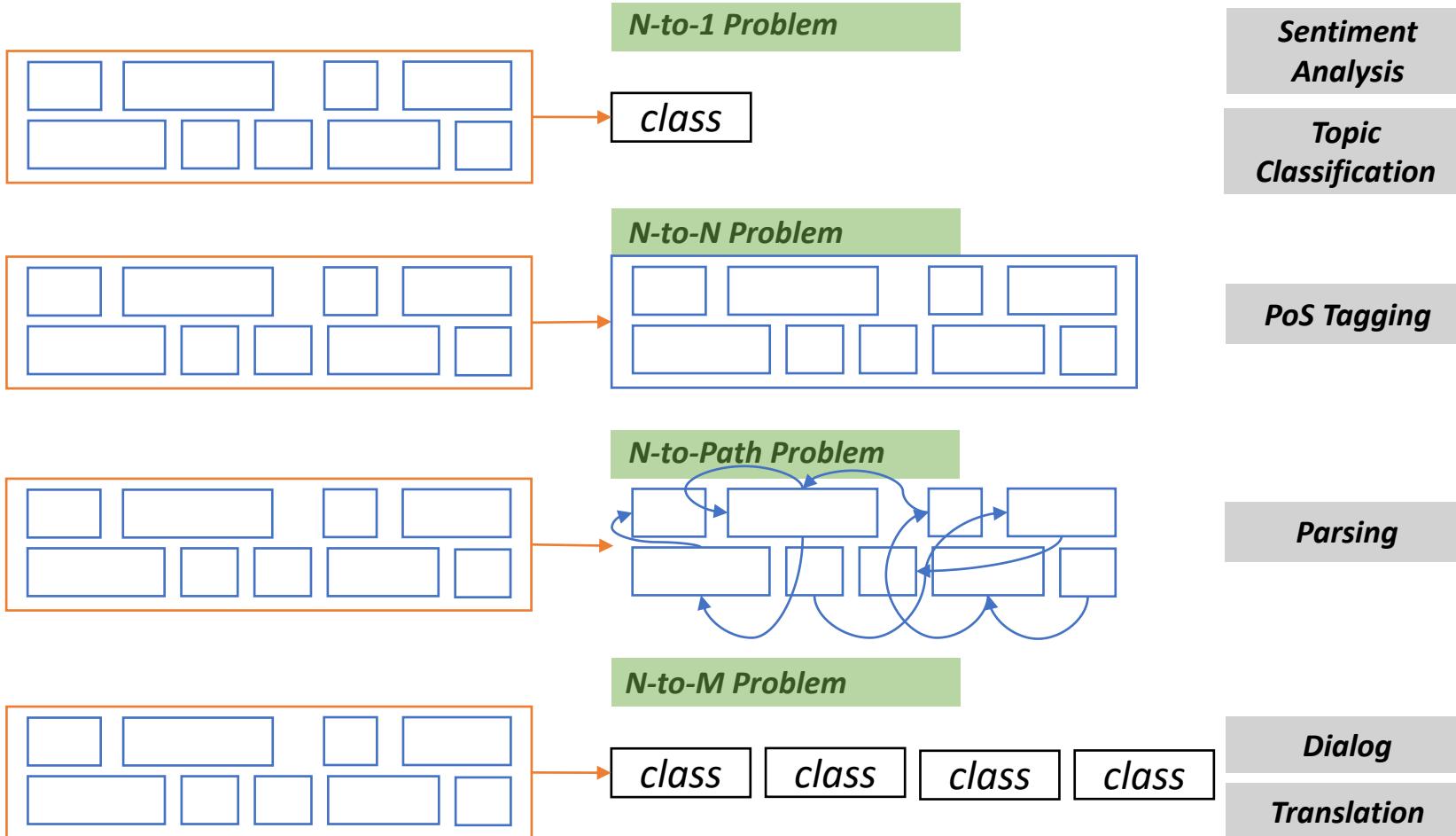
1. Machine Learning and NLP: Finish
2. Seq2Seq Learning
3. Seq2Seq Deep Learning
 1. RNN (Recurrent Neural Network)
 2. LSTM (Long Short-Term Memory)
 3. GRU (Gated Recurrent Unit)
4. Data Transformation for Deep Learning NLP
5. Next Week Preview
 - Natural Language Processing Stack

The purpose of Natural Language Processing: Overview

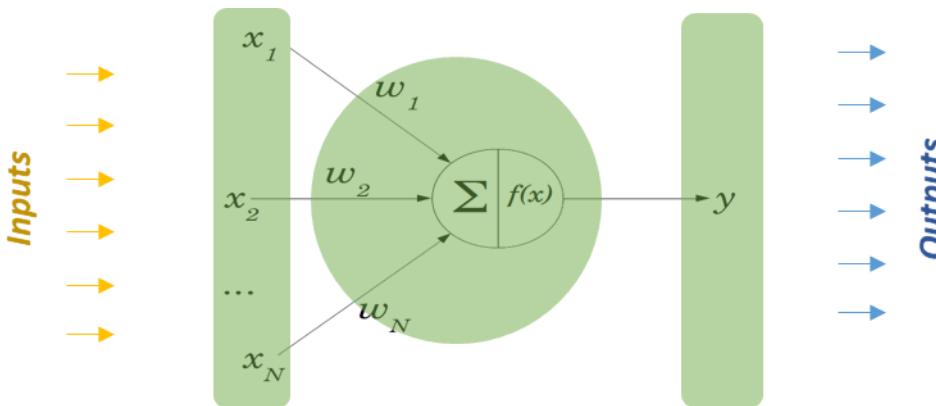


NLP Stack	Entity Extraction	When Sebastian Thrun ...	When Sebastian Thrun PERSON started at Google ORG in 2007 DATE
	Parsing	Claudia sat on a stool	<pre> graph TD S --- NP1[NP] S --- VP NP1 --- N1[Claudia] VP --- V1[sat] VP --- PP PP --- P1[on] PP --- AT1[a] PP --- NP2[NP] NP2 --- N2[stool] </pre>
	PoS Tagging	She sells seashells	[she/PRP] [sells/VBZ] [seashells/NNS]
	Stemming	Drinking, Drank, Drunk	Drink
	Tokenisation	How is the weather today	[How] [is] [the] [weather] [today]

Problem Abstraction

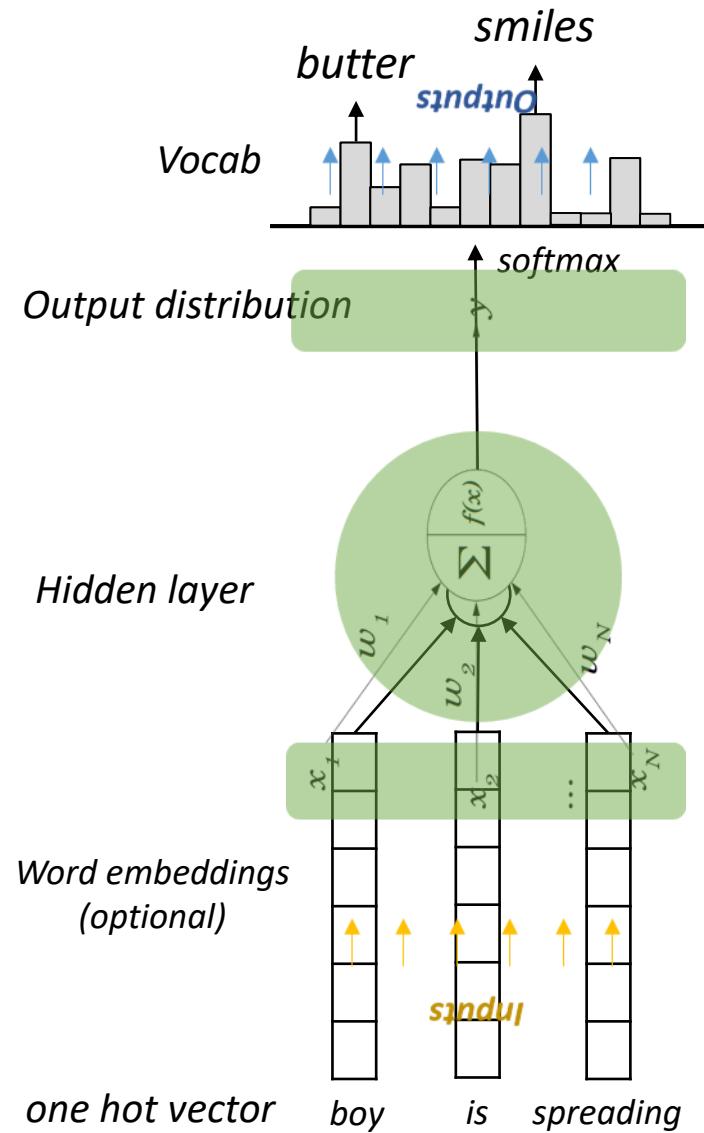


Prediction

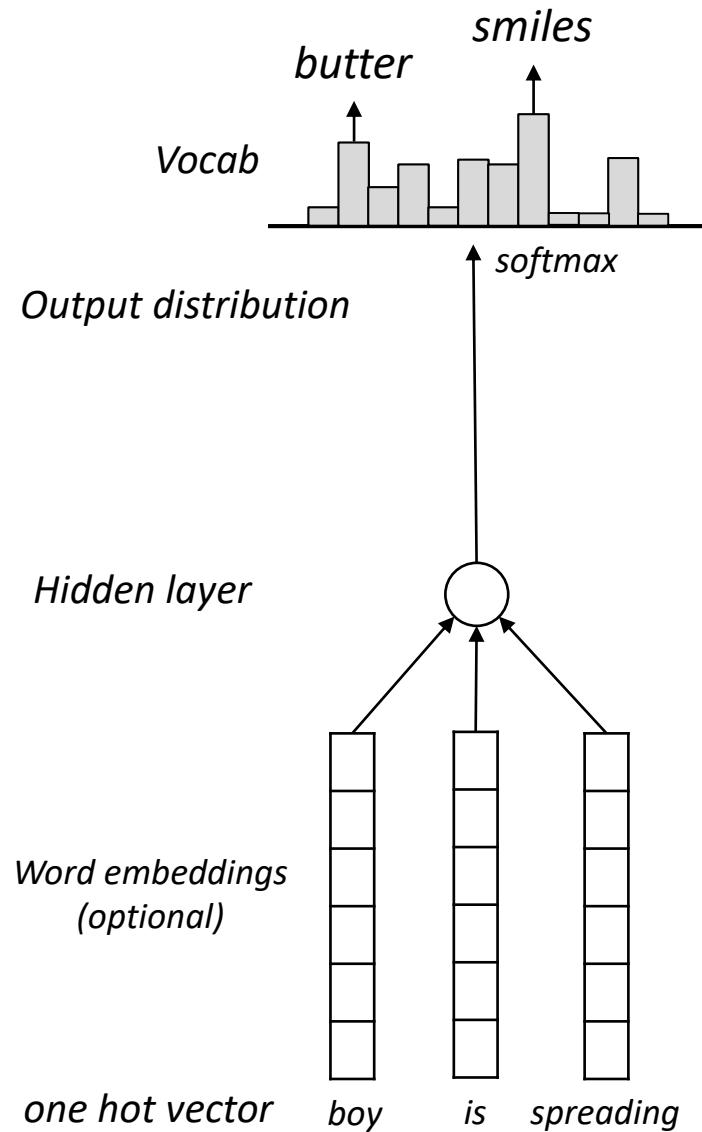


x_i	Inputs	Features words (indices or vectors!), context windows, sentences, documents, etc.
y_i	Outputs (labels)	What we try to predict/classify <ul style="list-style-type: none"> E.g. word meaning, sentiment, name entity

Prediction



Prediction



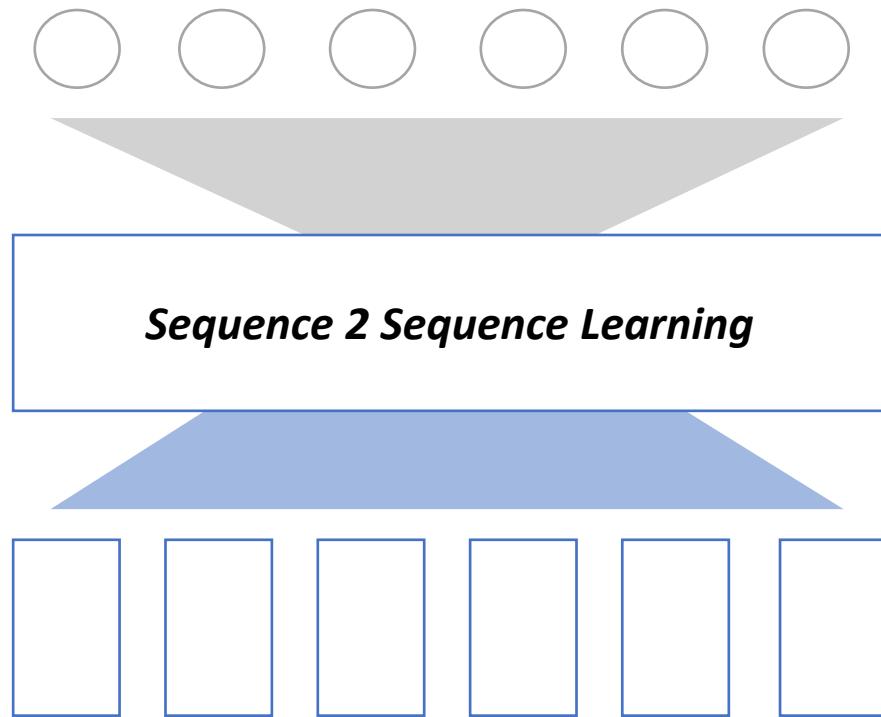
What if we consider this as
a sequential input?
 Let's add the concept 'time'

0 LECTURE PLAN

Lecture 4: Word Classification and Machine Learning 2

1. Machine Learning and NLP: Finish
2. **Seq2Seq Learning**
3. Seq2Seq Deep Learning
 1. RNN (Recurrent Neural Network)
 2. LSTM (Long Short-Term Memory)
 3. GRU (Gated Recurrent Unit)
4. Data Transformation for Deep Learning NLP
5. Next Week Preview
 - Natural Language Processing Stack

Illustration



Sequence 2 Sequence Learning

Running time

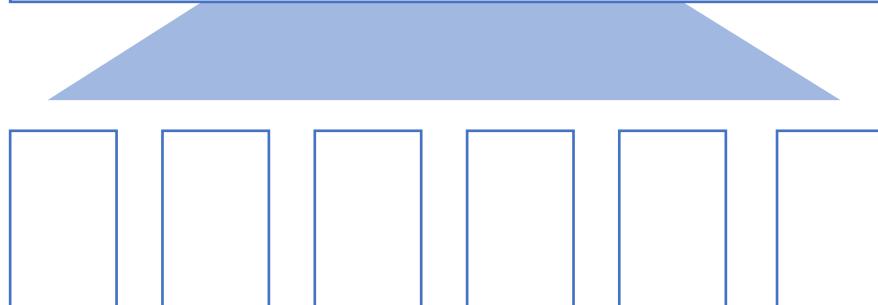
$M = \# \text{ of } \textcolor{green}{\circ}$



Sequence Generation



Sequence 2 Sequence Learning

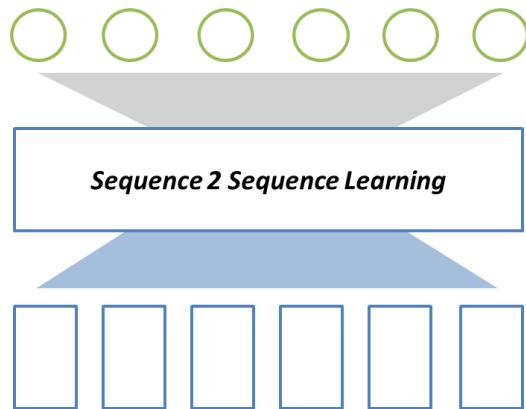


Sequence Feeding

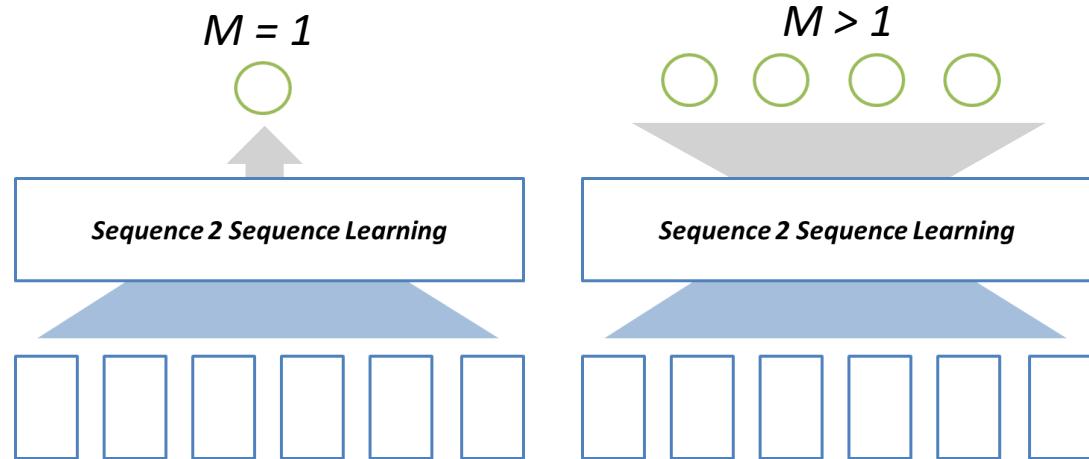
$N = \# \text{ of } \textcolor{blue}{\square}$

Sequence 2 Sequence Learning

$N = M$



$N \neq M$

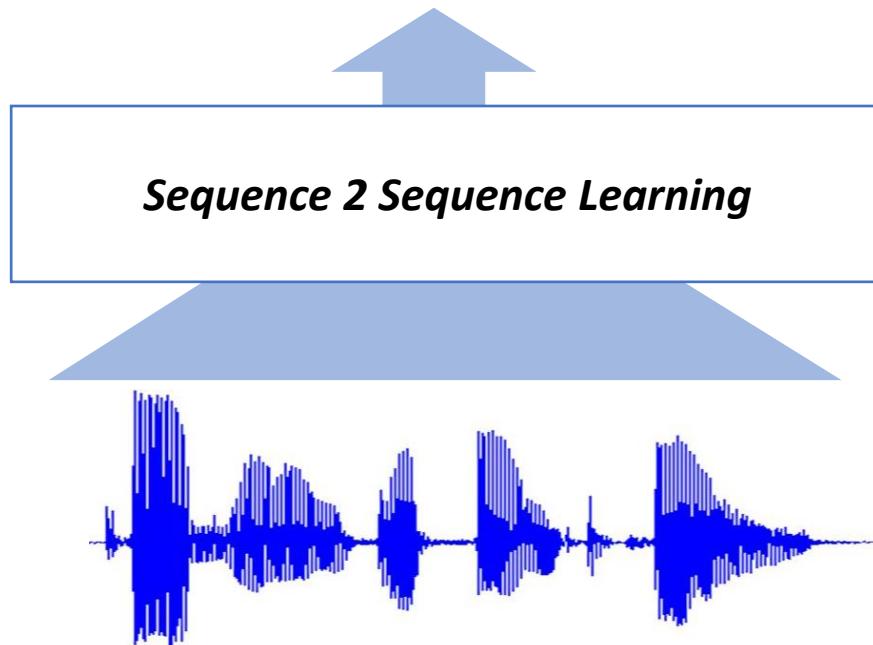


Sequence 2 Sequence Learning

Seq2Seq – Speech Recognition

How is the weather today

Output: Text



Input: Speech Signal

Sequence 2 Sequence Learning

Seq2Seq – Movie Frame Labelling

Swing Swing Hit Bat_Broken



Sequence 2 Sequence Learning



Output: Scene Labels



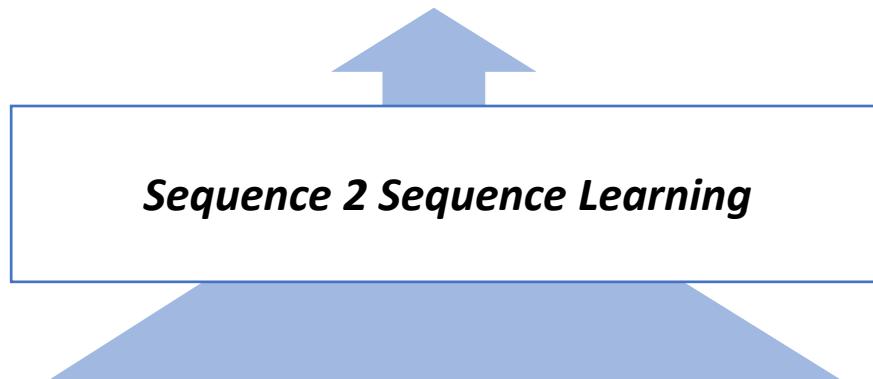
Input: Video Frame

Sequence 2 Sequence Learning

Seq2Seq – PoS Tagging

ADV VERB DET NOUN NOUN

Output: Part of Speech



How is the weather today

Input: Text

2

Sequence 2 Sequence Learning

Seq2Seq – Arithmetic Calculation

4. A farmer has 7 ducks.
He has 5 times as many chickens as ducks.
How many more chickens than ducks does he have?

ducks



chickens



Find the number of chickens first.

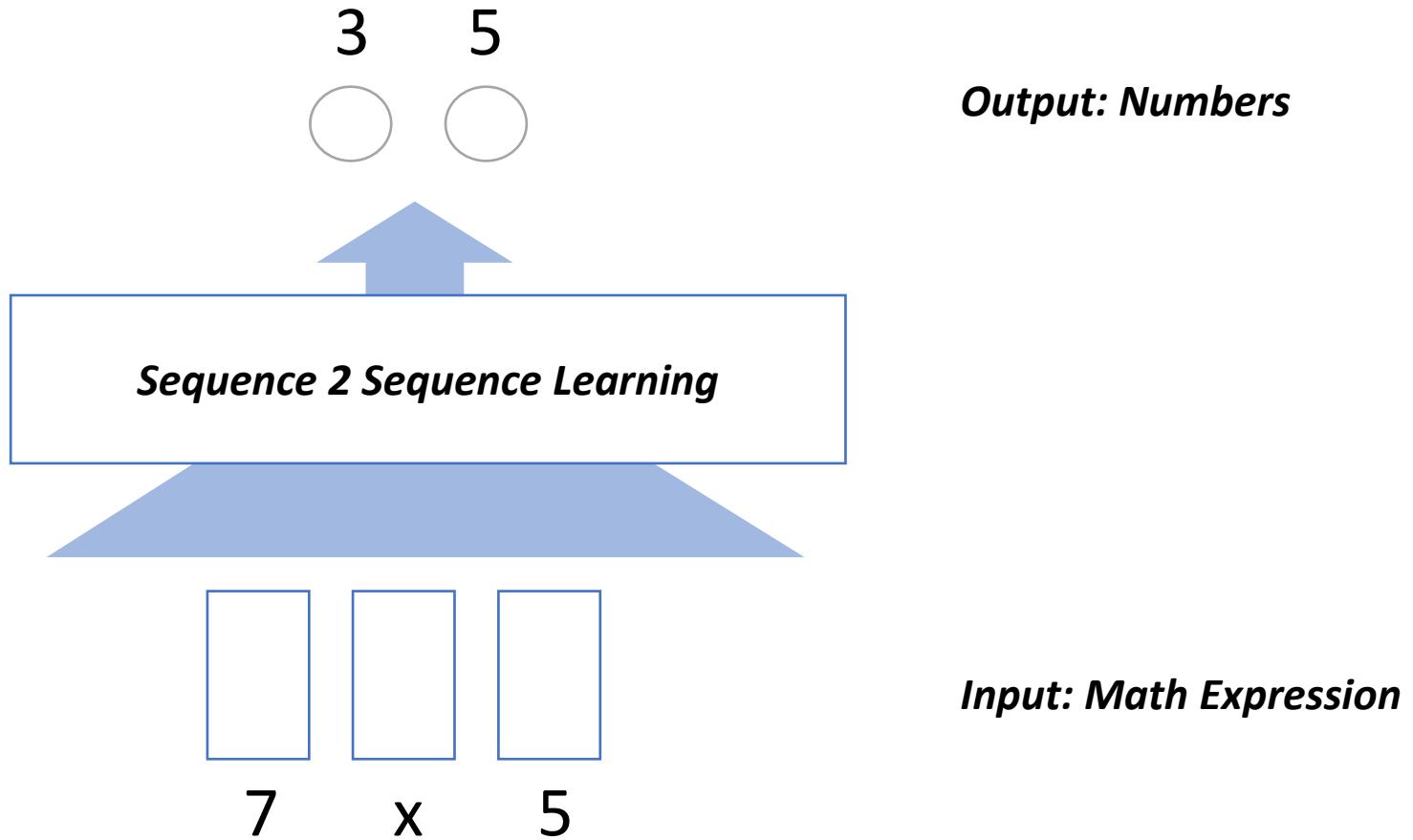


$$\boxed{7} \times \boxed{5} = \boxed{35}$$

X Y

Sequence 2 Sequence Learning

Seq2Seq – Arithmetic Calculation



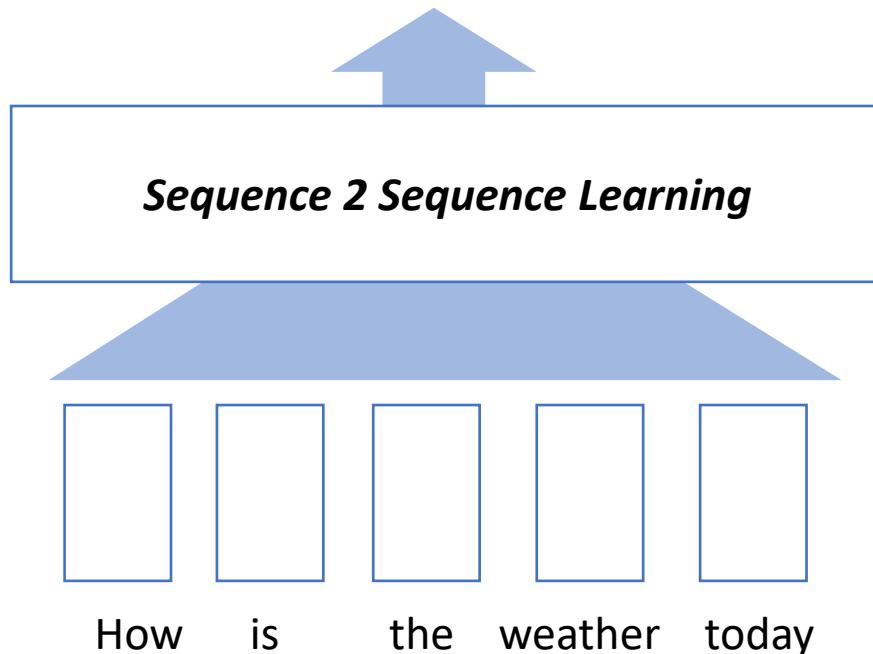
2

Sequence 2 Sequence Learning

Seq2Seq – Machine Translation

今天 天气 怎么 样?

Output: Chinese Text



Input: English Text

Sequence 2 Sequence Learning

Seq2Seq – Sentence Completion

How is the weather today?

How long does it take?

Let's go to the opera house

It is quite hot inside

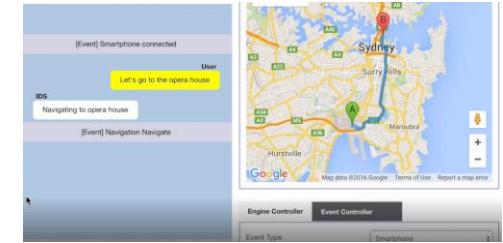
I may need to stop by Darling Harbour

When is the dinner appointment

Change the schedule

Text him that I cannot meet at 6:30pm

I like learning Natural Language Processing



Sequence 2 Sequence Learning

Seq2Seq – Sentence Completion

How is the weather today?

How long does it take?

Let's go to the opera house

It is quite hot inside

I may need to stop by Darling Harbour

When is the dinner appointment

Change the schedule

Text him that I cannot meet at 6:30pm

X

I like learning Natural Language Processing

Y

Sequence 2 Sequence Learning

I like learning Natural Language Processing

Seq2Seq – Sentence Completion

Natural Language Processing



Output: Partial Sentence



Sequence 2 Sequence Learning



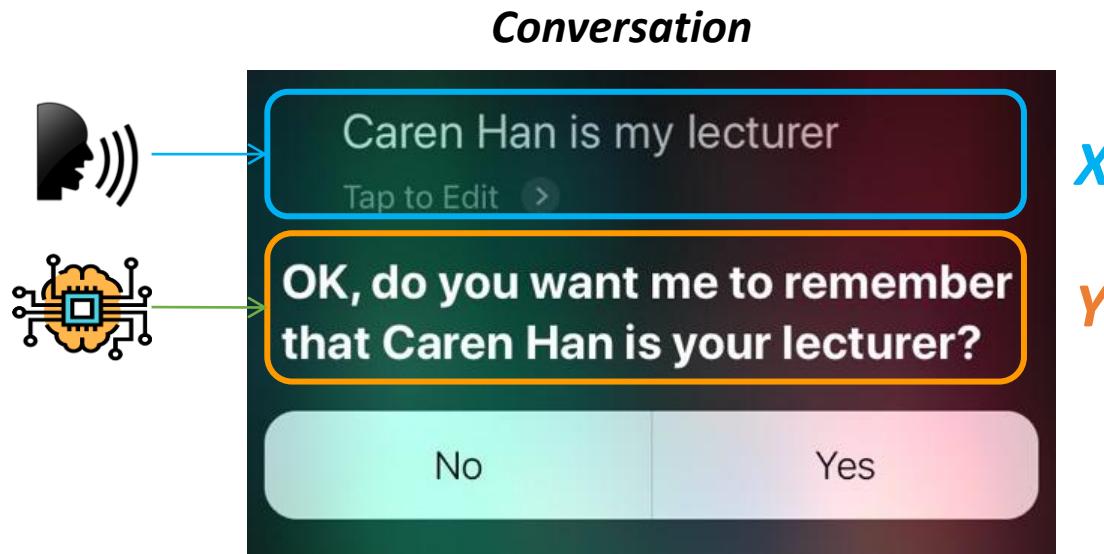
Input: Partial Sentence

I

like

learning

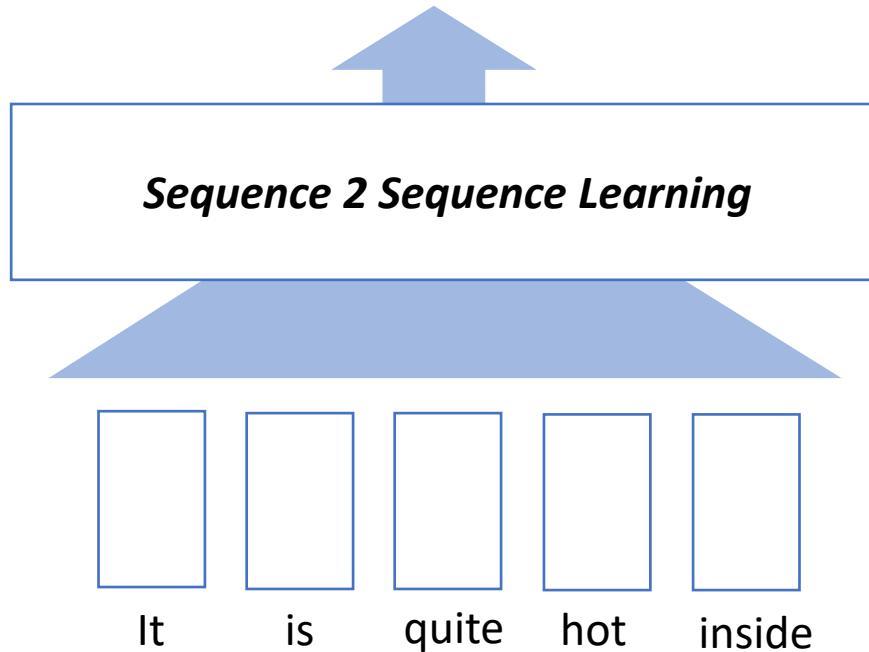
Seq2Seq – Conversation Modelling



Seq2Seq – Conversation Modelling

Okay. I will open windows for you

Output: Utterance



Input: Utterance

0 LECTURE PLAN

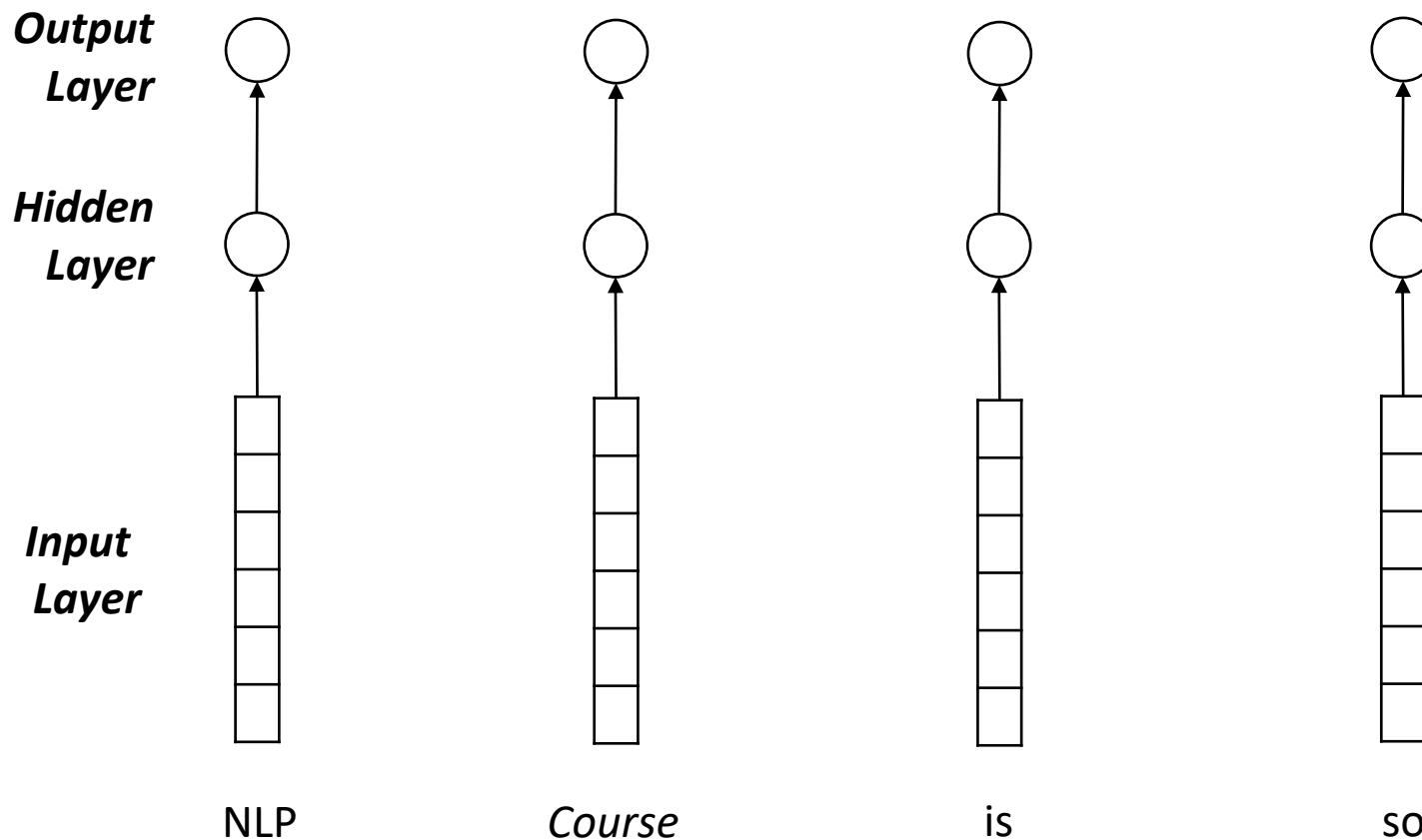
Lecture 4: Word Classification and Machine Learning 2

1. Machine Learning and NLP: Finish
2. Seq2Seq Learning
3. **Seq2Seq Deep Learning**
 1. RNN (Recurrent Neural Network)
 2. LSTM (Long Short-Term Memory)
 3. GRU (Gated Recurrent Unit)
4. Data Transformation for Deep Learning NLP
5. Next Week Preview
 - Natural Language Processing Stack

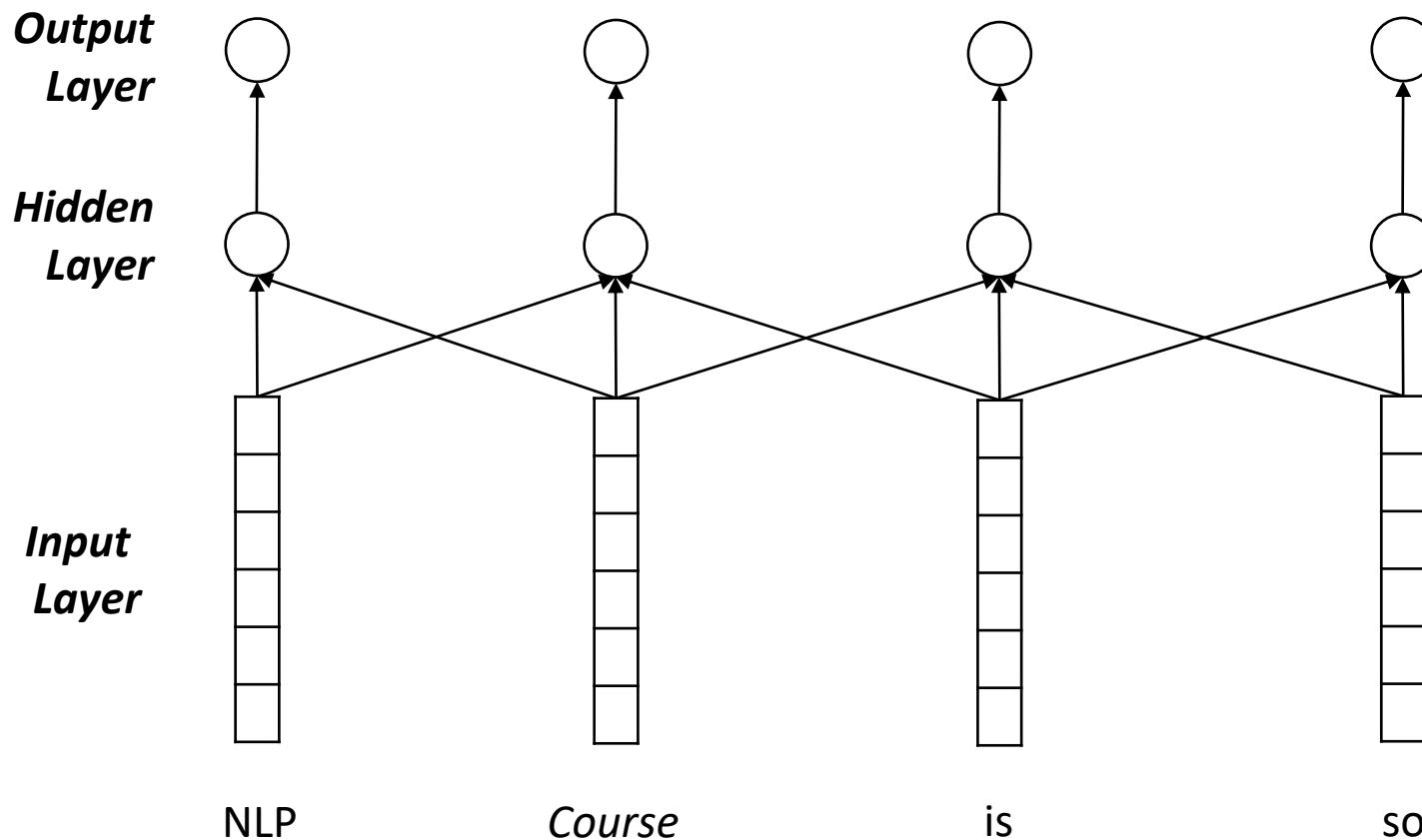
3

Seq2Seq with Deep Learning

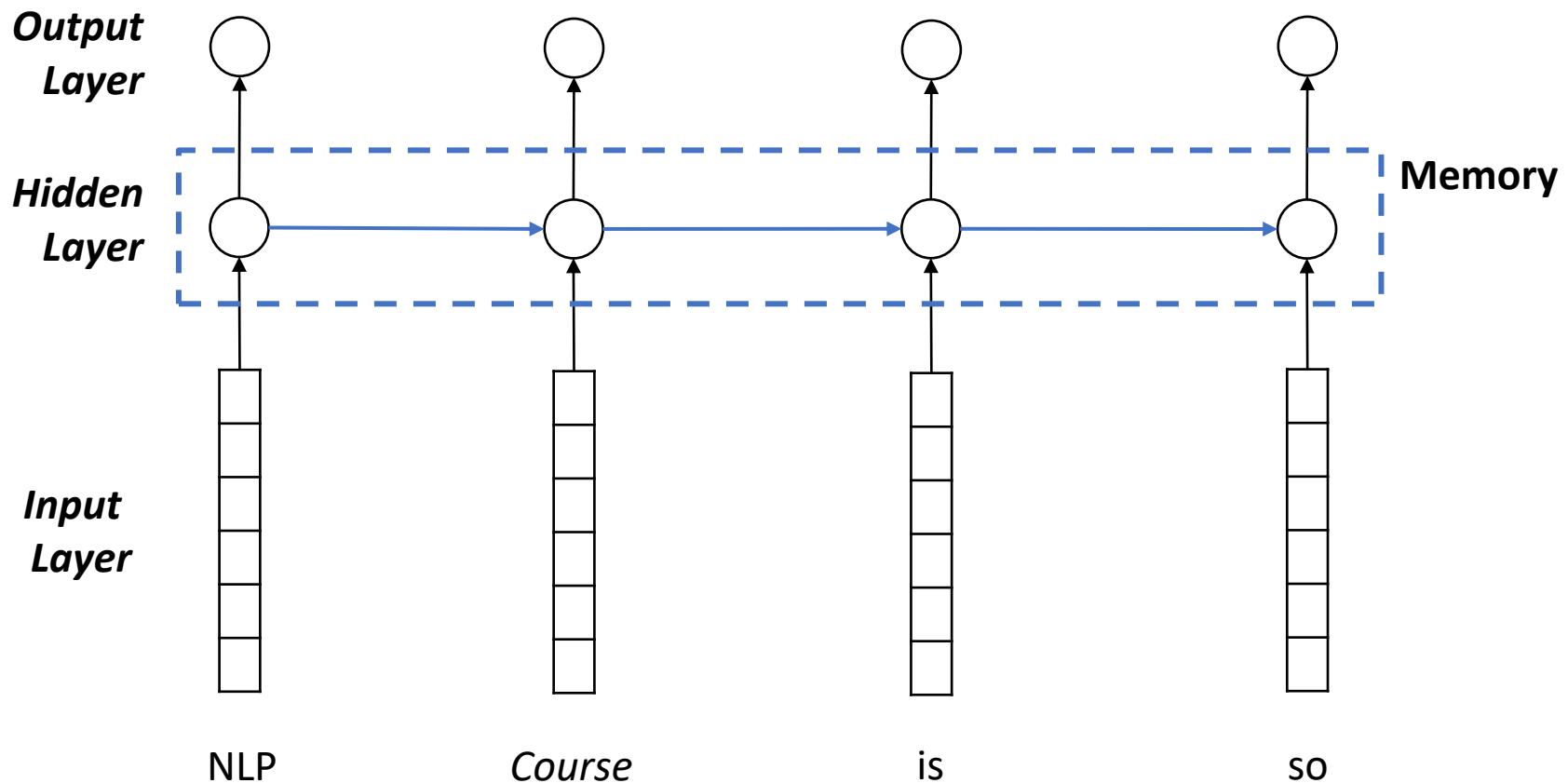
Prediction



Prediction + Convolution Idea



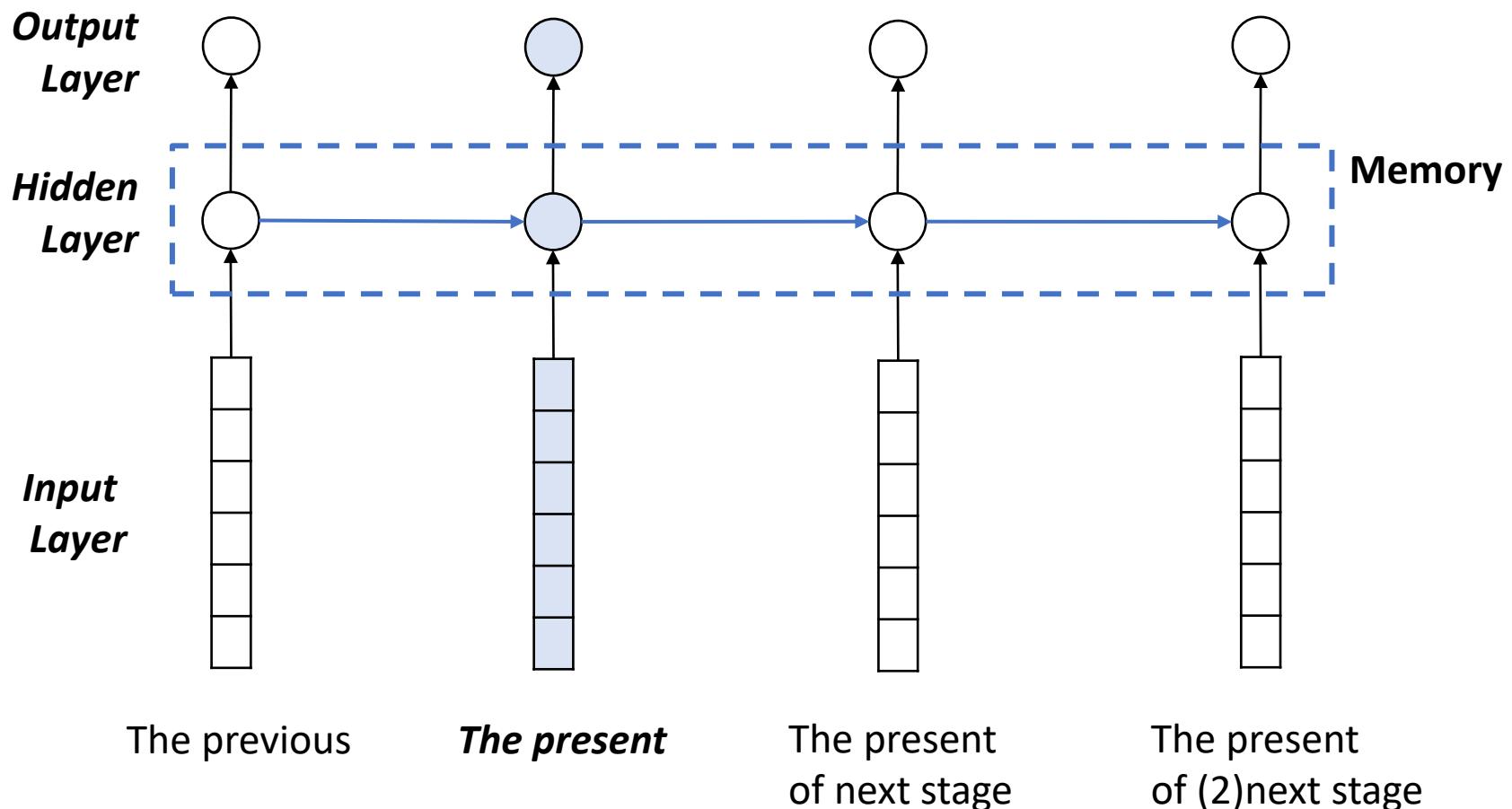
Prediction + Memory = Sequence Modelling



3

Seq2Seq with Deep Learning

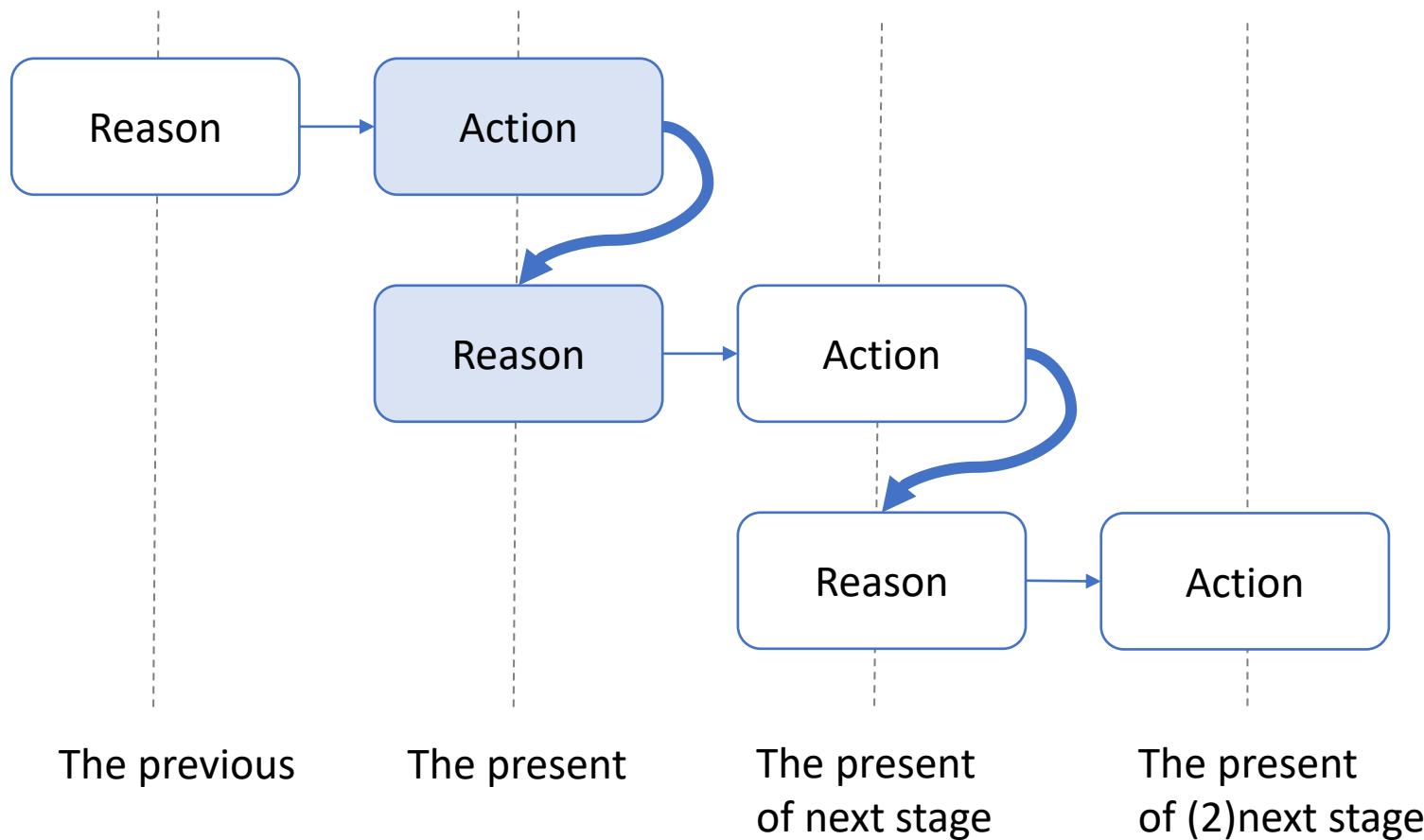
Prediction + Memory = Sequence Modelling



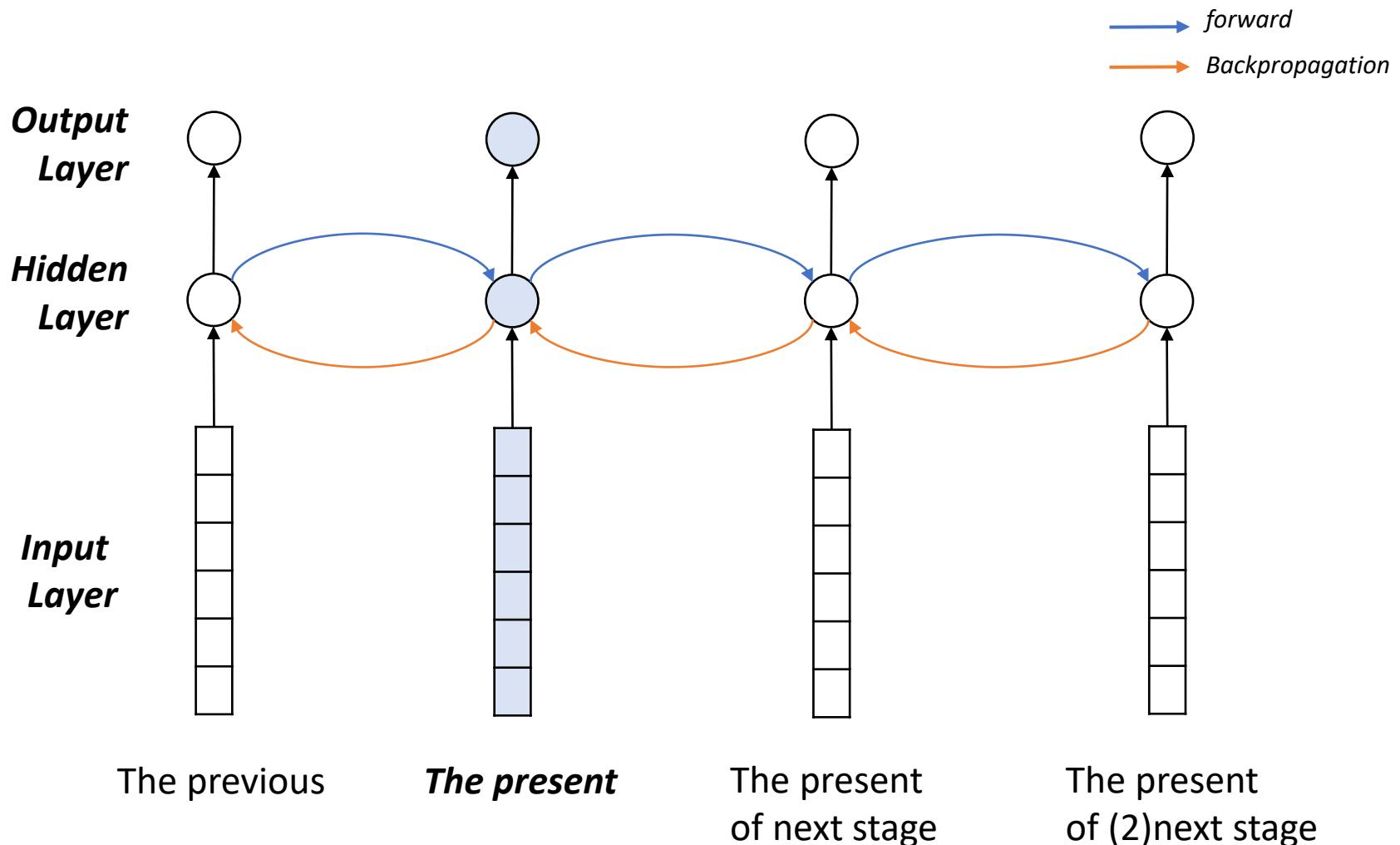
Seq2Seq with Deep Learning

Neural Network + Memory

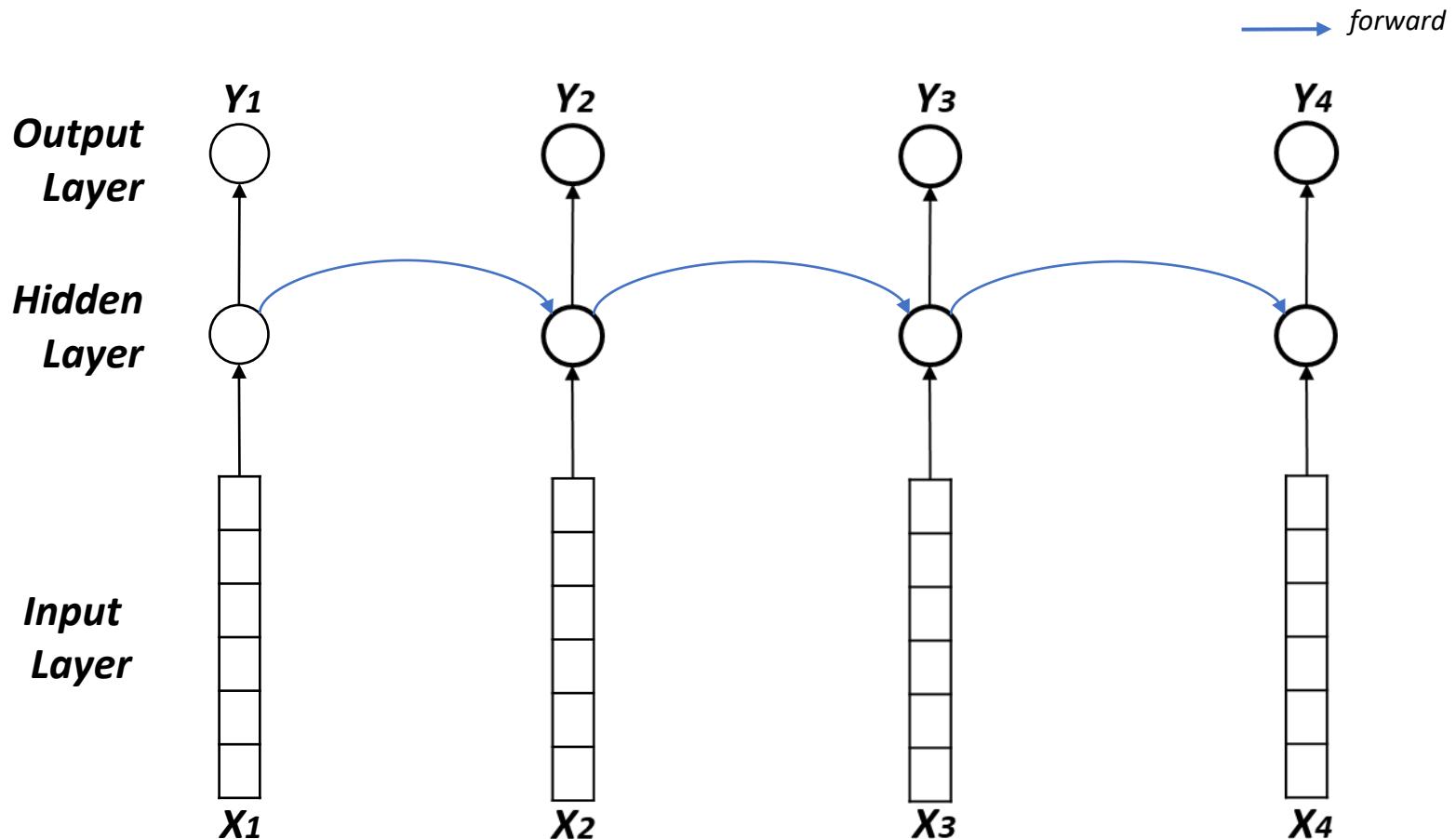
Memory is vital to experiences; it is the retention of information over time for the purpose of influencing future action



Neural Network + Memory

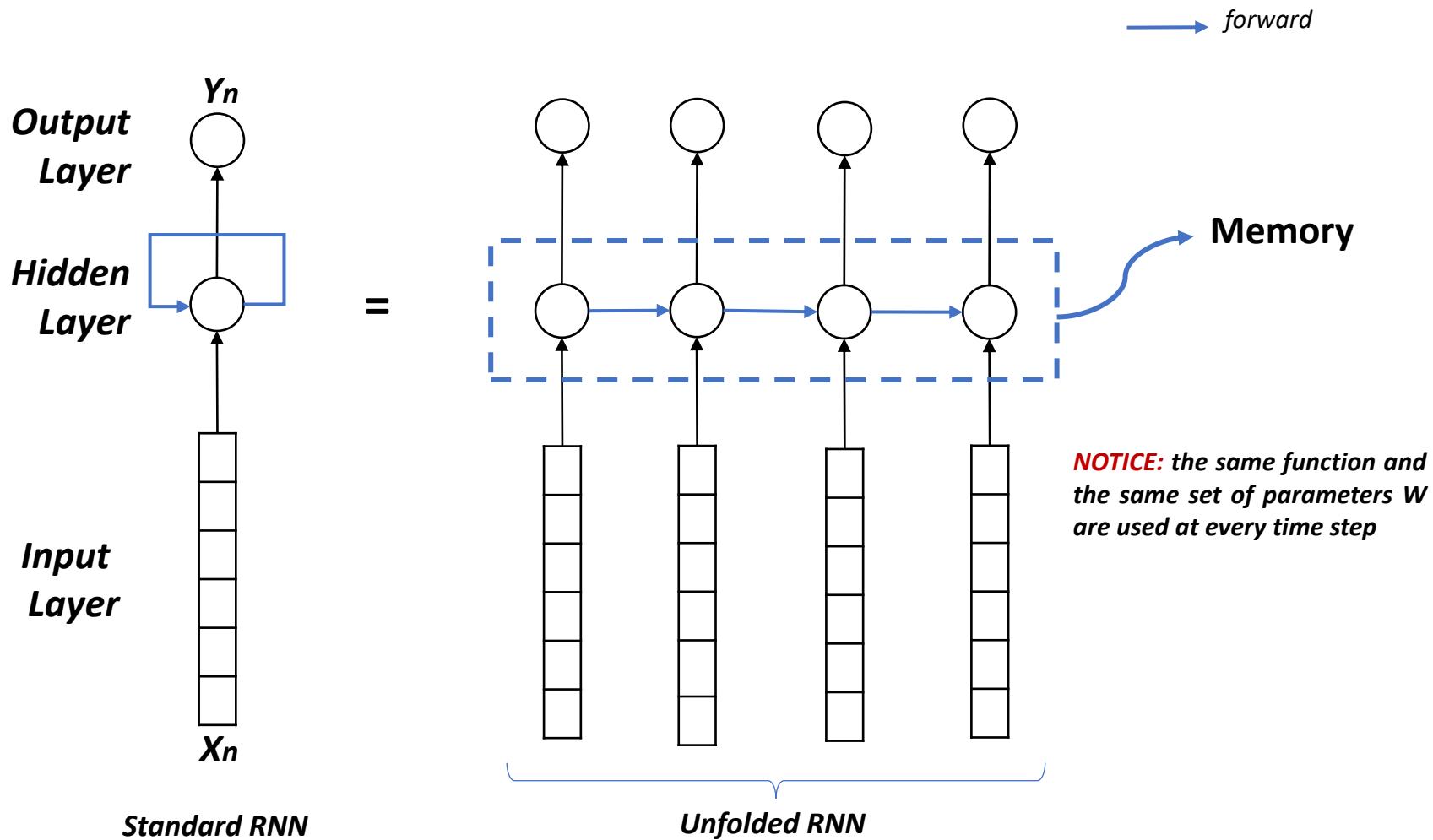


Neural Network + Memory = Recurrent Neural Network



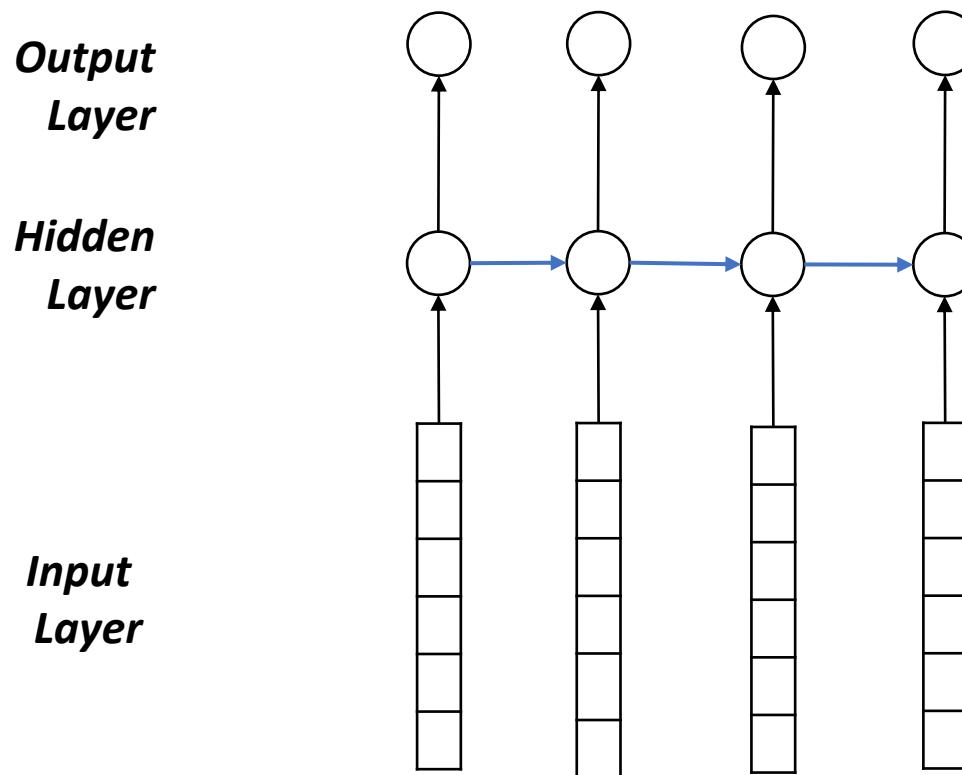
Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network (RNN)



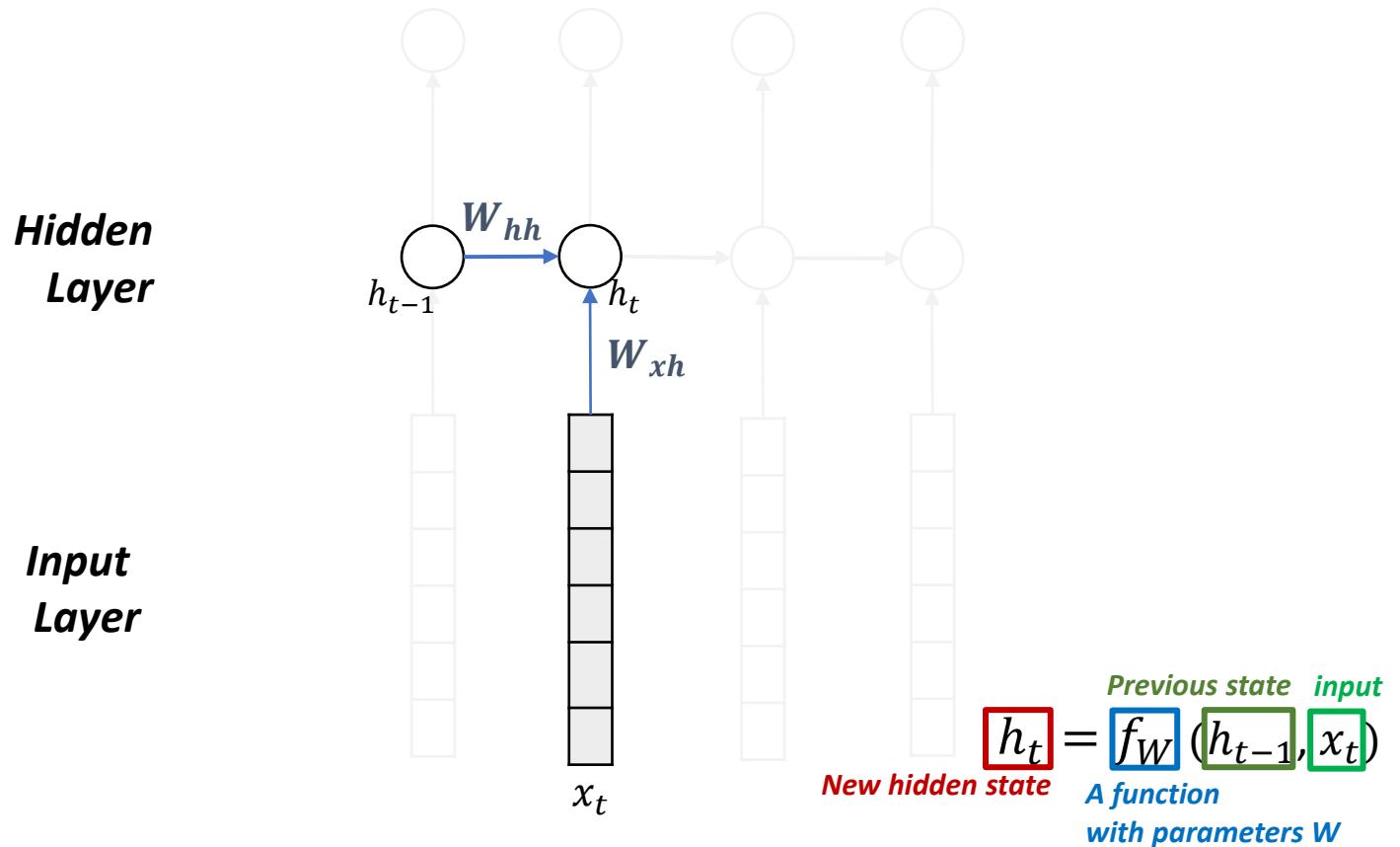
Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network (RNN)



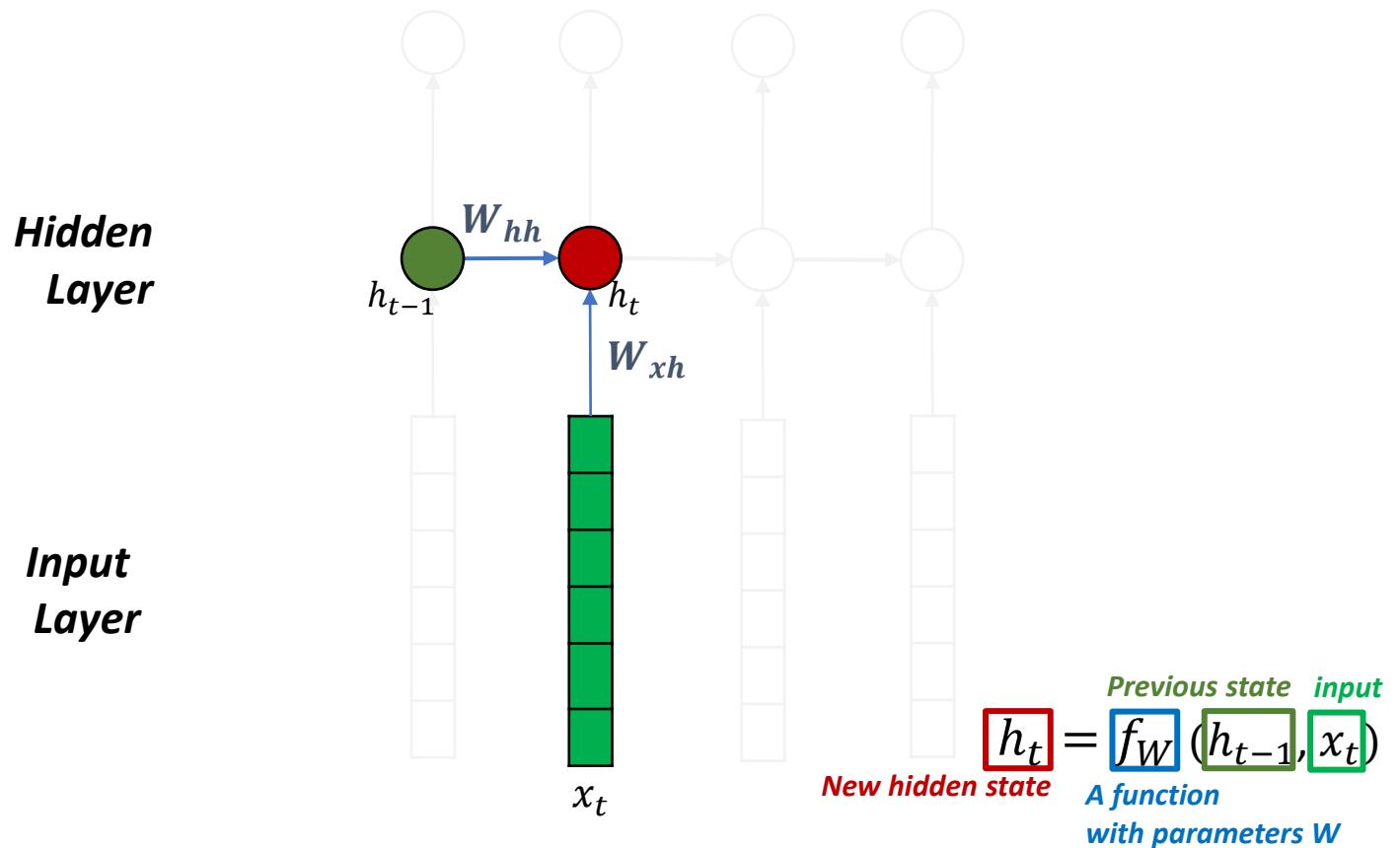
Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network



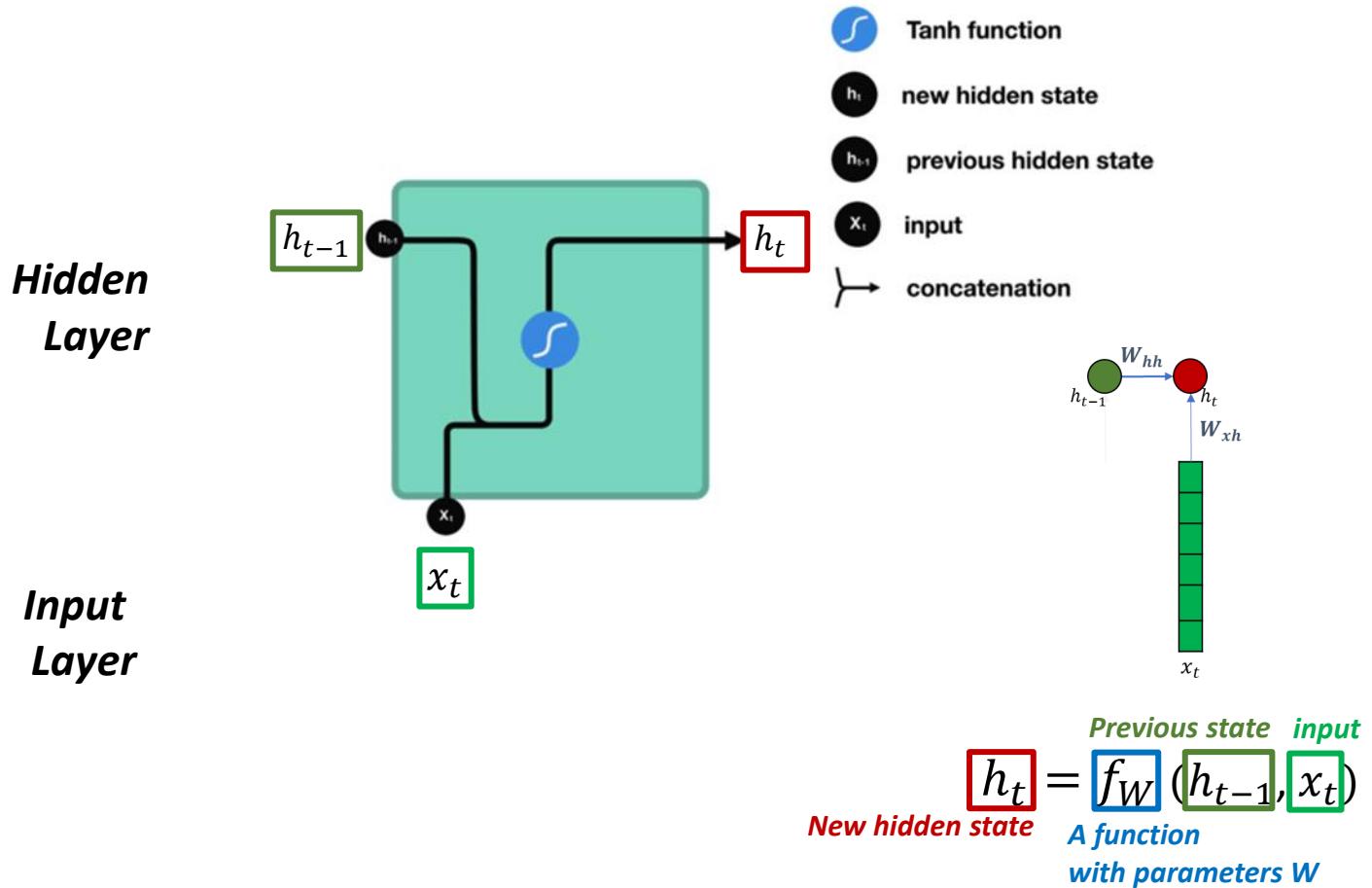
Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network



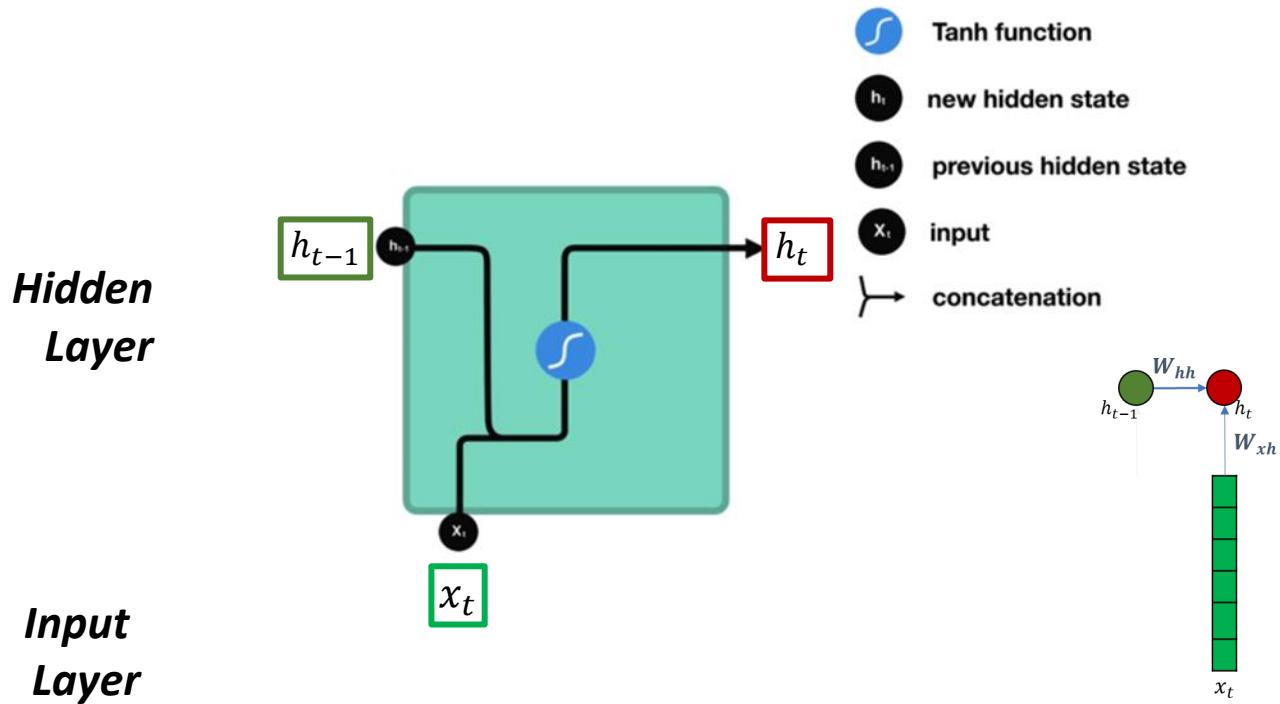
Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network



Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network

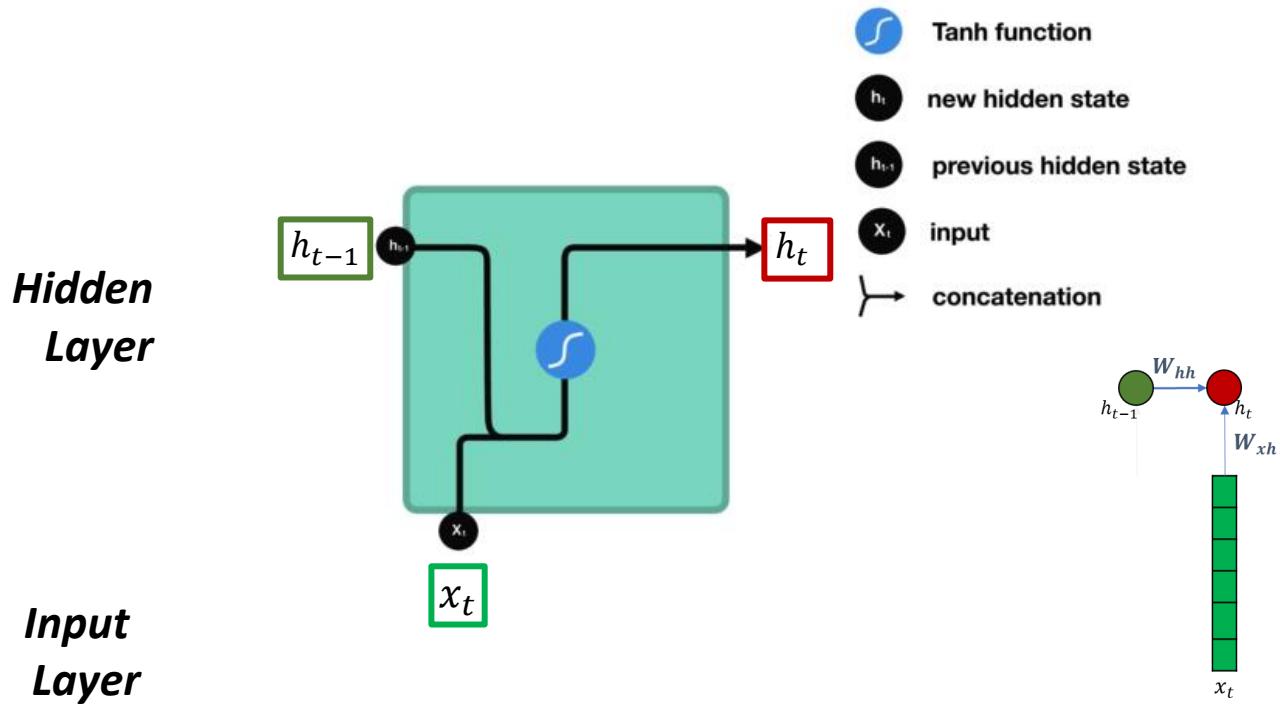


$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$

New hidden state Previous state input
 A function with parameters W

Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network



$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$

New hidden state Previous state input
 A function with parameters W

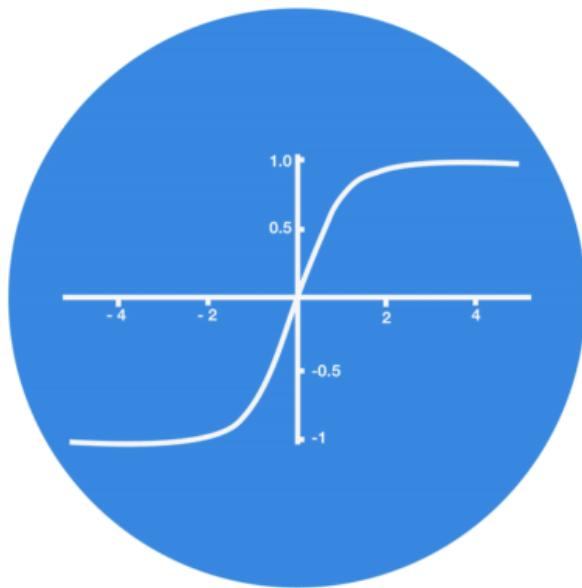
3

Seq2Seq with Deep Learning

Tanh activation

The tanh activation is used to help regulate the values flowing through the network. The tanh function squishes values to always be between -1 and 1.

5
0.1
-0.5

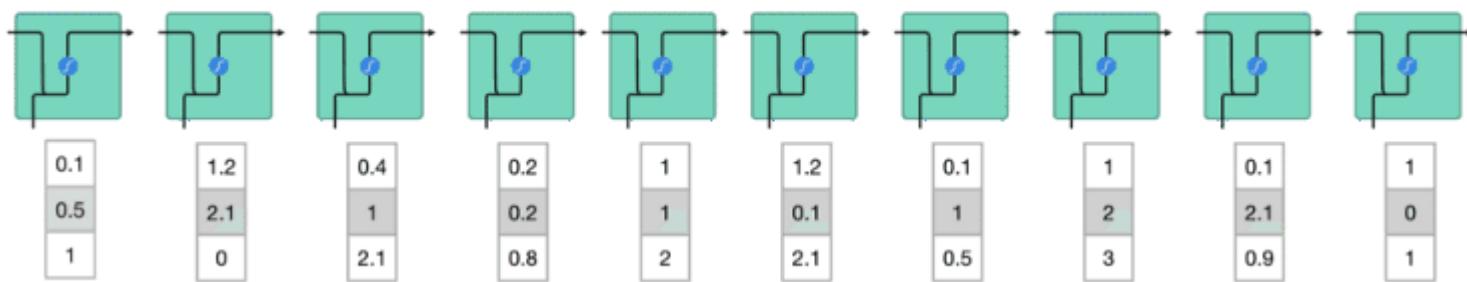


3

Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network

With Sequence Input



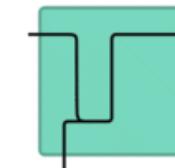
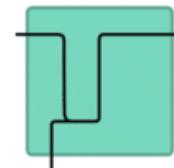
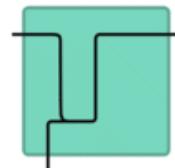
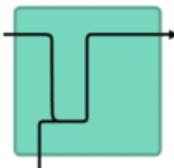
3

Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network

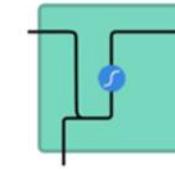
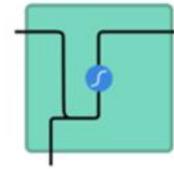
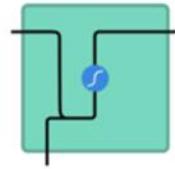
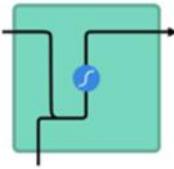
Q: *Why do we need tanh function?*

5
0.01
-0.5



Vector Transformations without tanh

5
0.01
-0.5



Vector Transformations *with tanh*

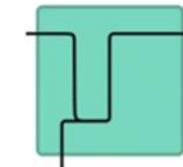
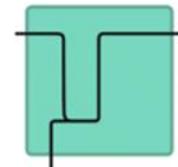
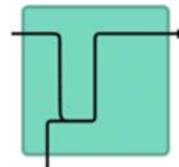
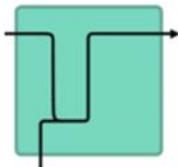
3

Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network

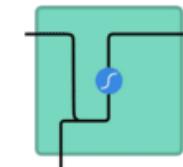
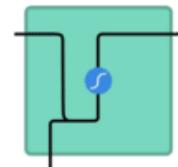
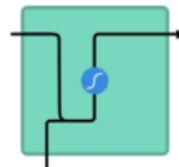
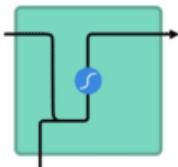
Q: *Why do we need tanh function?*

5
0.01
-0.5



Vector Transformations without tanh

5
0.01
-0.5

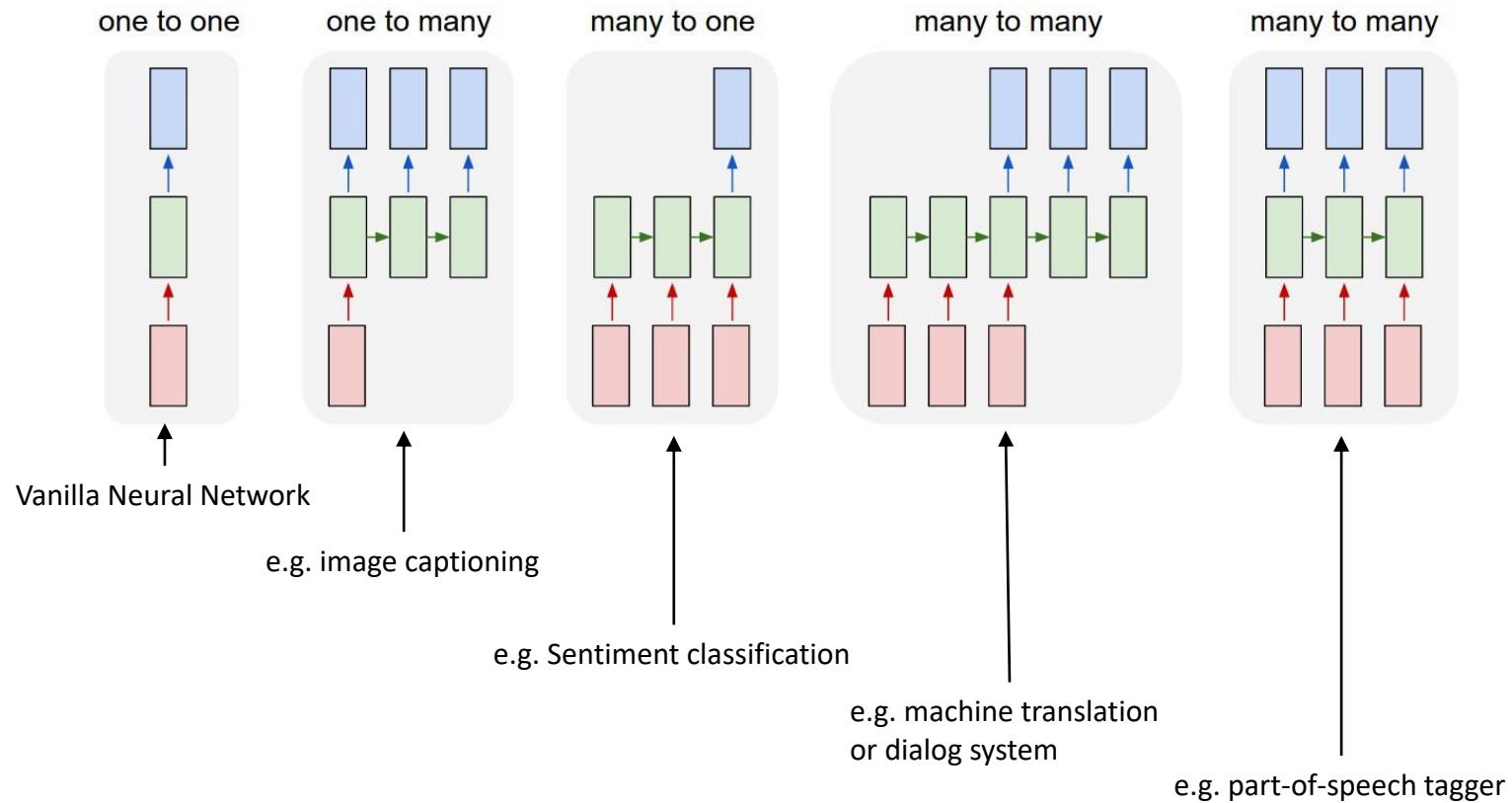


Vector Transformations with tanh

Seq2Seq with Deep Learning

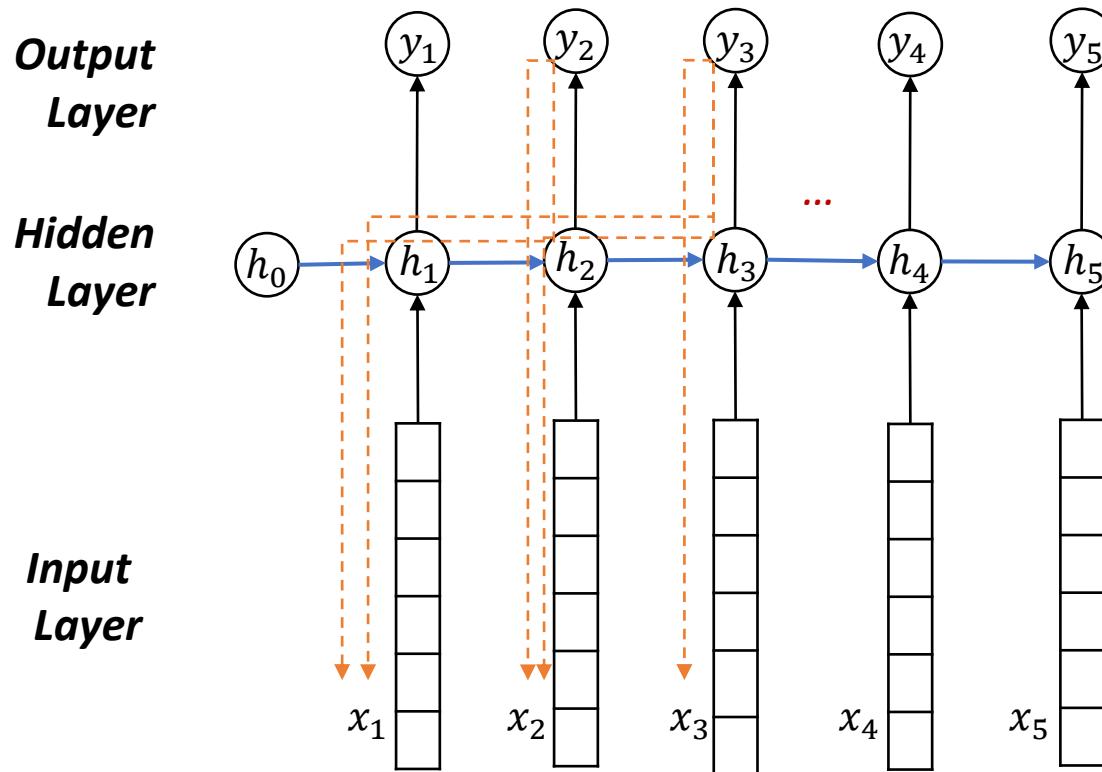
Neural Network + Memory = Recurrent Neural Network

Several Variants of RNN



Neural Network + Memory = Recurrent Neural Network

Backpropagation through time

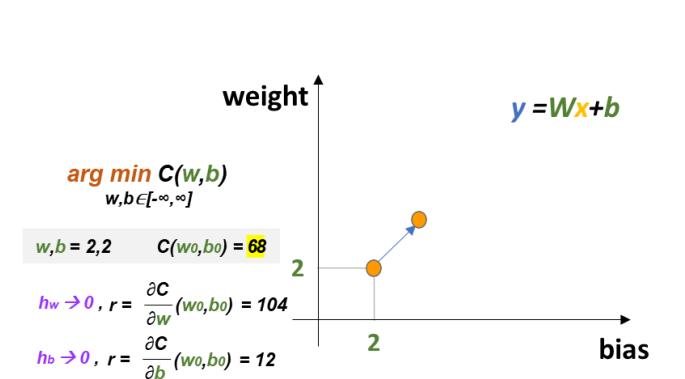
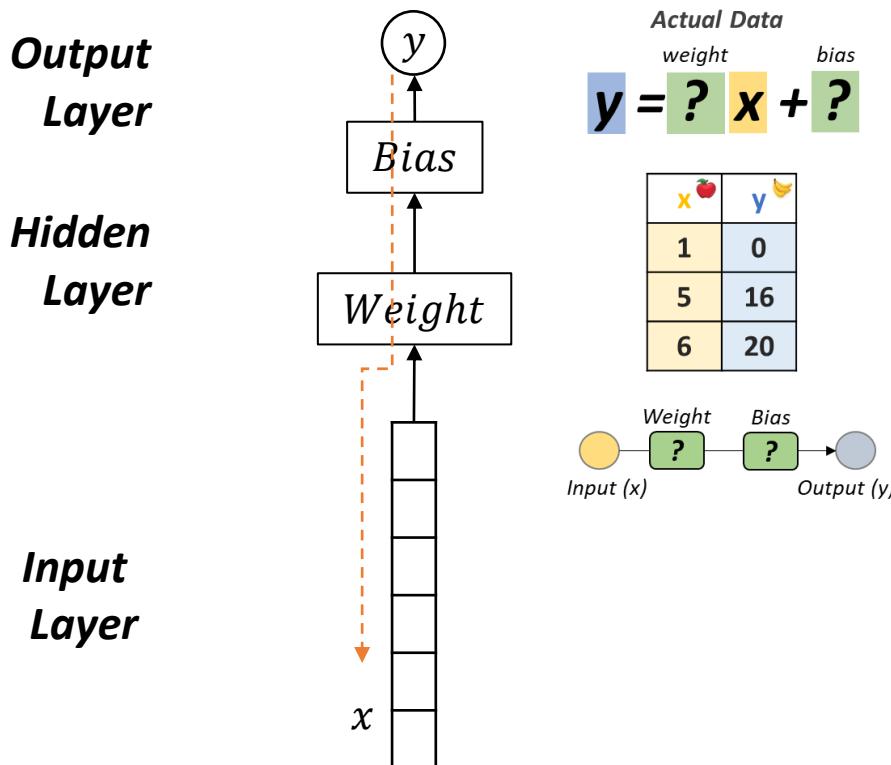


- Similar as **standard backpropagation** on unrolled network
- Similar as **training very deep networks** with tied parameters

Seq2Seq with Deep Learning

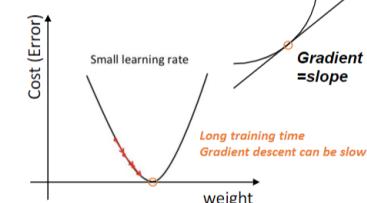
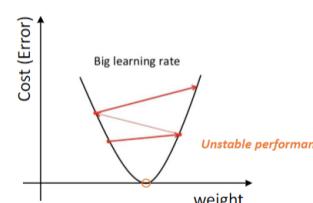
Neural Network + Memory = Recurrent Neural Network

Remember? Backpropagation in the basic neural networks



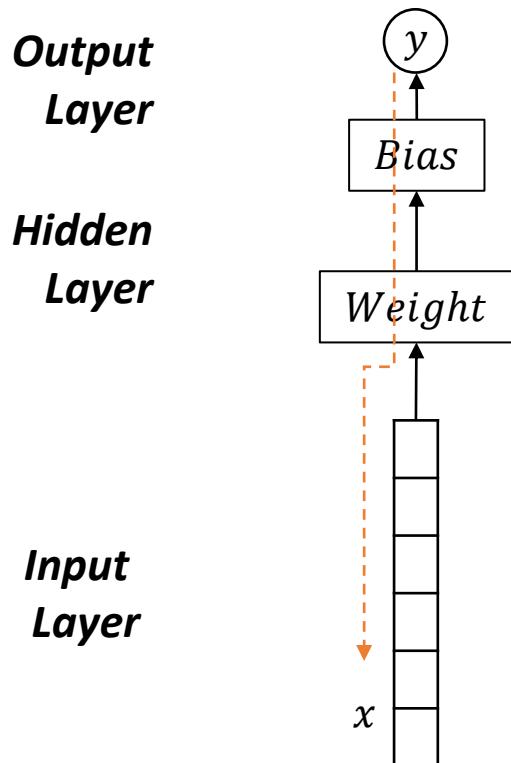
a = learning rate or step size

$$\arg \min c(w, b)$$



Neural Network + Memory = Recurrent Neural Network

Remember? Backpropagation in the basic neural networks

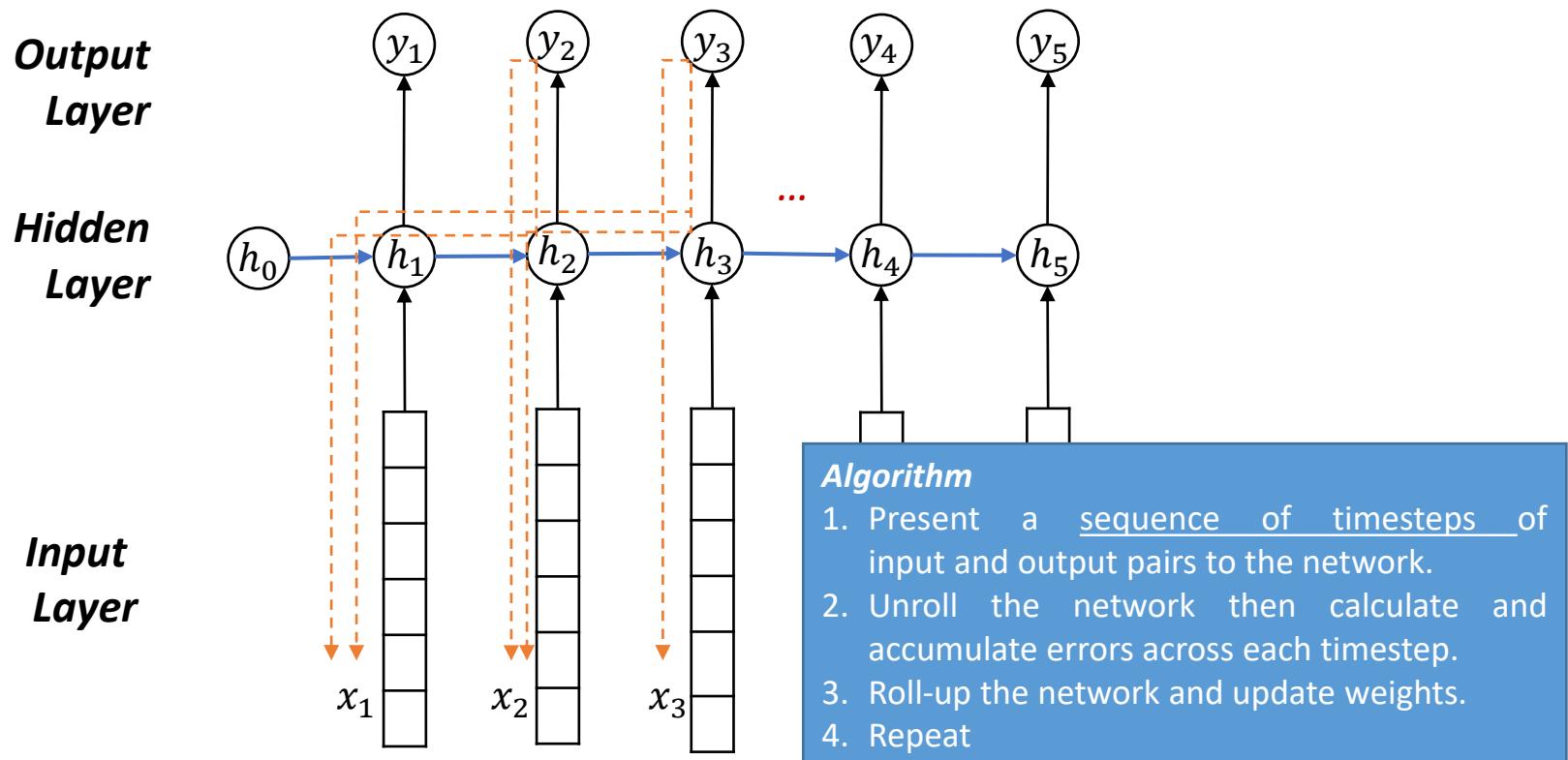


Algorithm

1. Present a training input pattern and propagate it through the network to get an output.
2. Compare the predicted outputs to the expected outputs and calculate the error.
3. Calculate the derivatives of the error with respect to the network weights.
4. Adjust the weights to minimize the error.
5. Repeat

Neural Network + Memory = Recurrent Neural Network

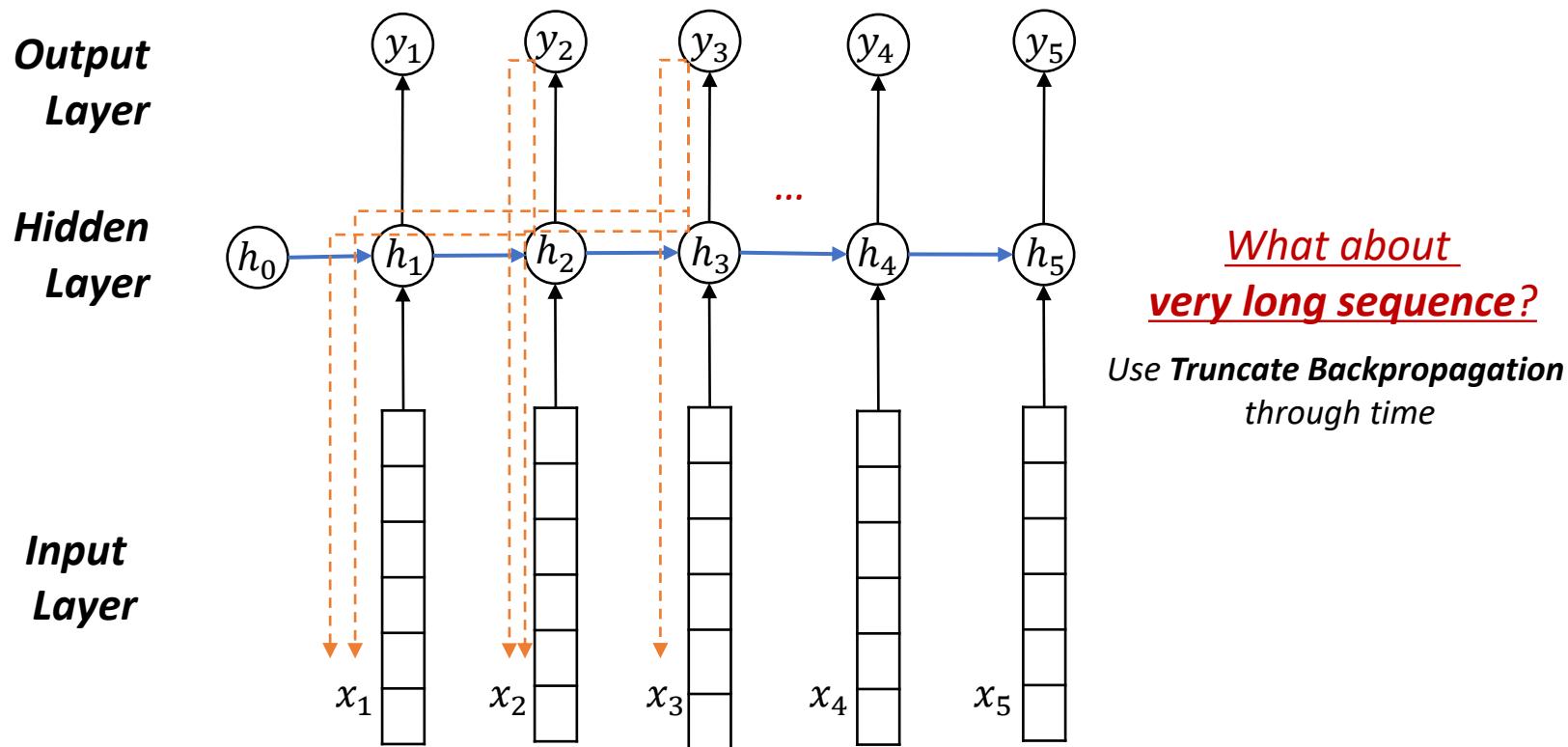
Backpropagation through time



- Similar as **standard backpropagation** on unrolled network
- Similar as **training very deep networks** with tied parameters

Neural Network + Memory = Recurrent Neural Network

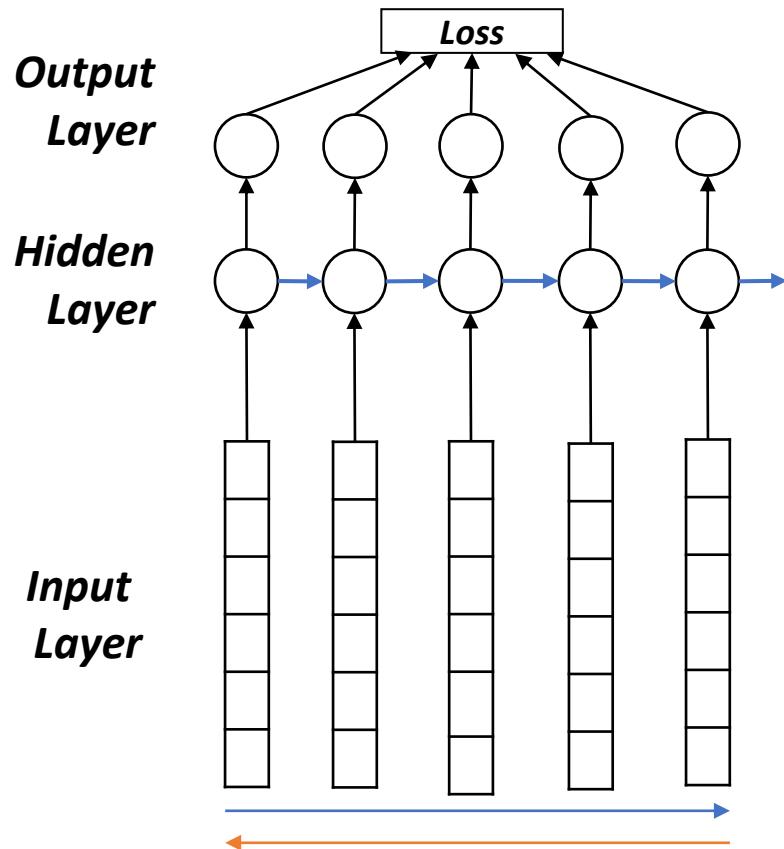
Backpropagation through time



- Similar as **standard backpropagation** on unrolled network
- Similar as **training very deep networks** with tied parameters

Neural Network + Memory = Recurrent Neural Network

Truncated Backpropagation through time

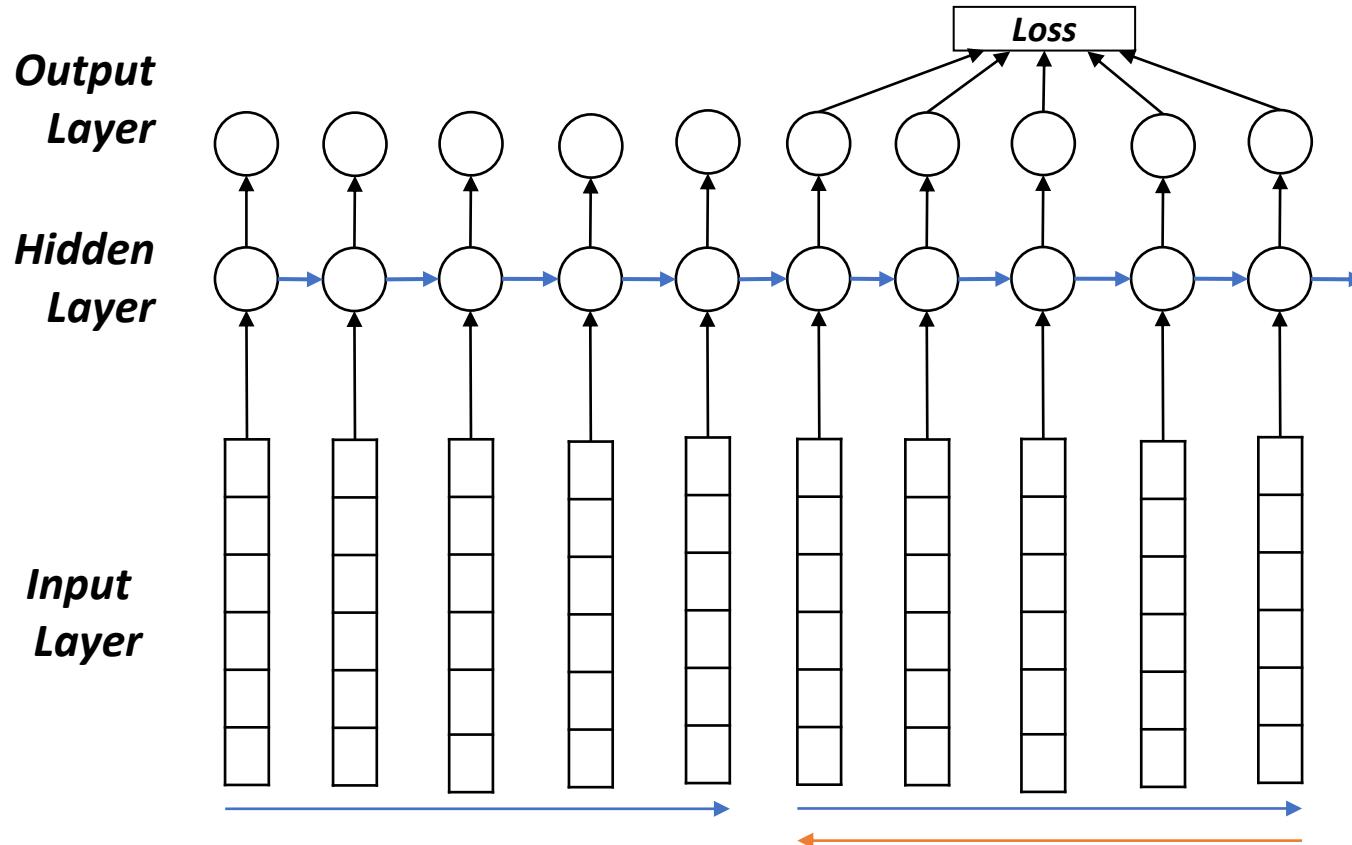


Run forward and backward through chunks of the sequence instead of whole sequence

Seq2Seq with Deep Learning

Neural Network + Memory = Recurrent Neural Network

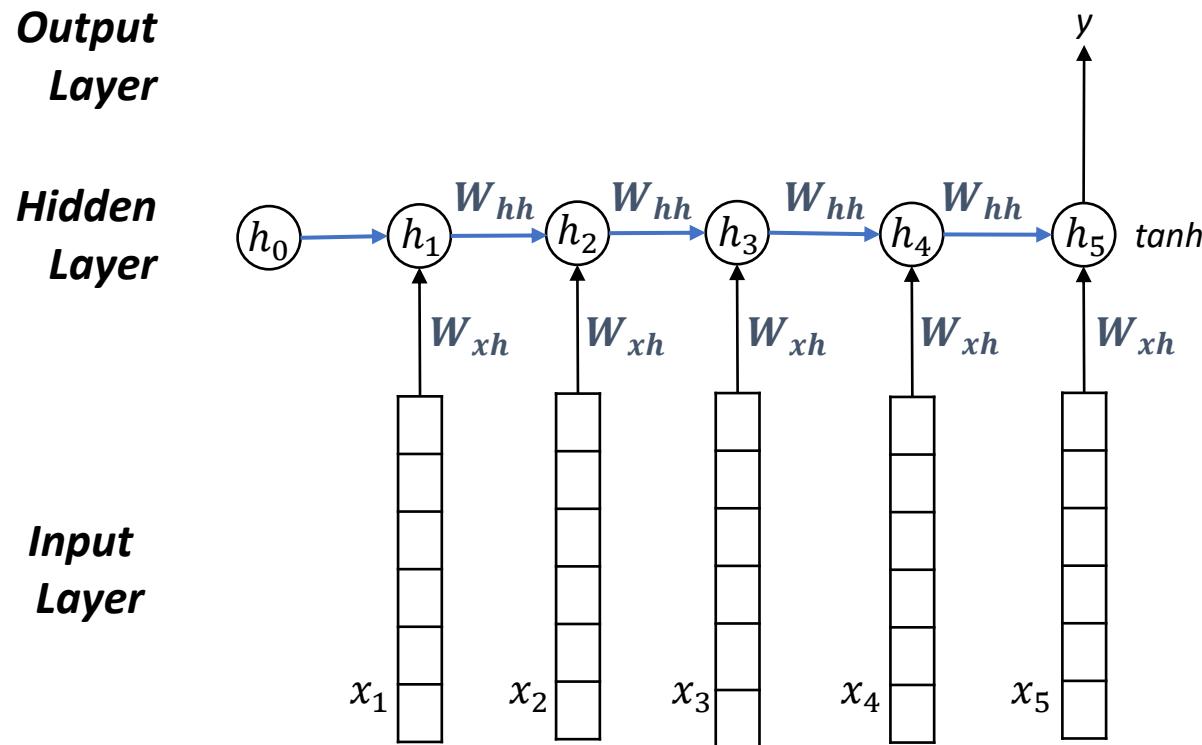
Truncated Backpropagation through time



Carry hidden states forward in time forever, but only backpropagate for some smaller number of steps

Neural Network + Memory = Recurrent Neural Network

Many to 1



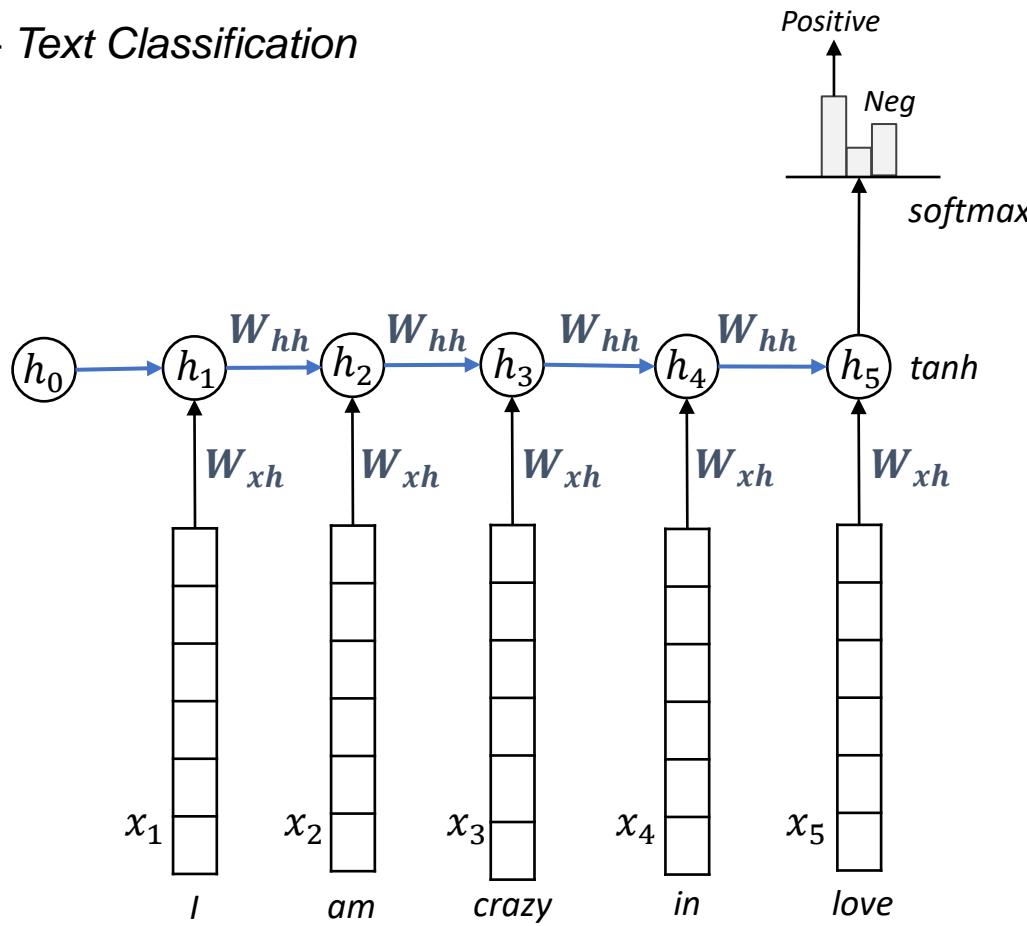
Neural Network + Memory = Recurrent Neural Network

Many to 1 – Text Classification

**Output
Layer**

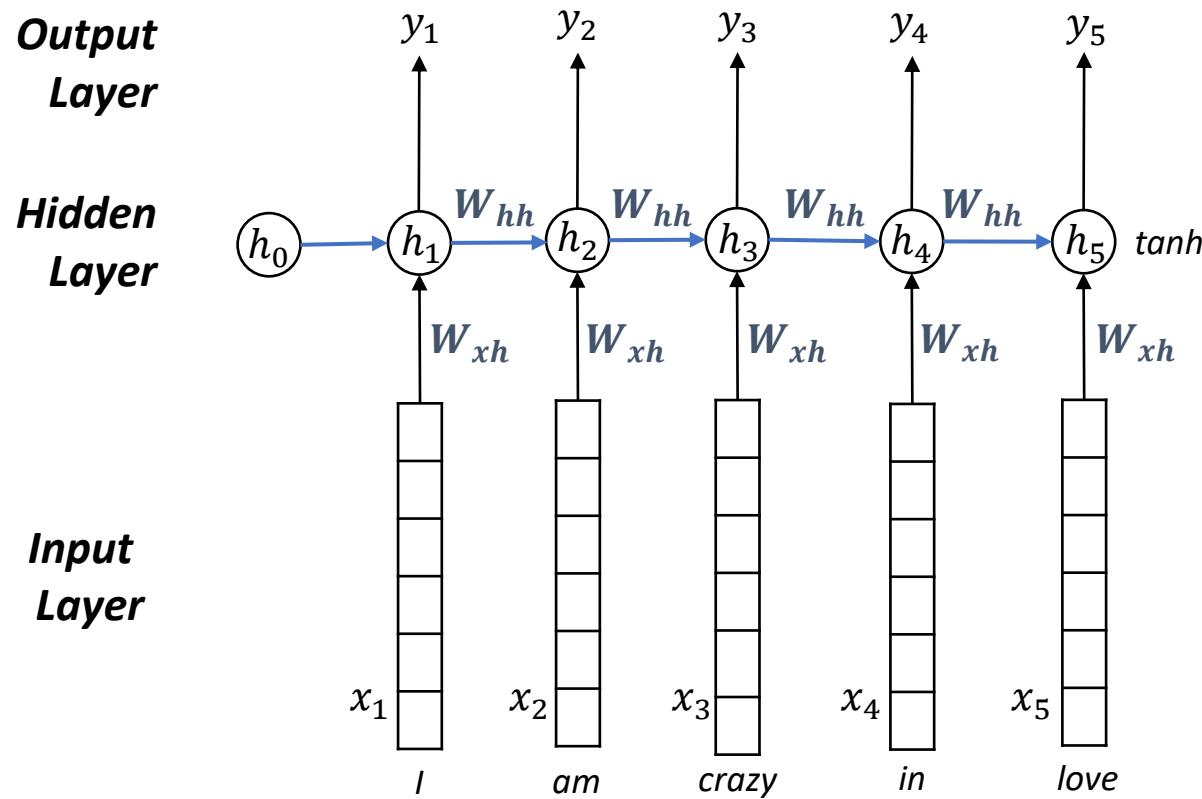
**Hidden
Layer**

**Input
Layer**



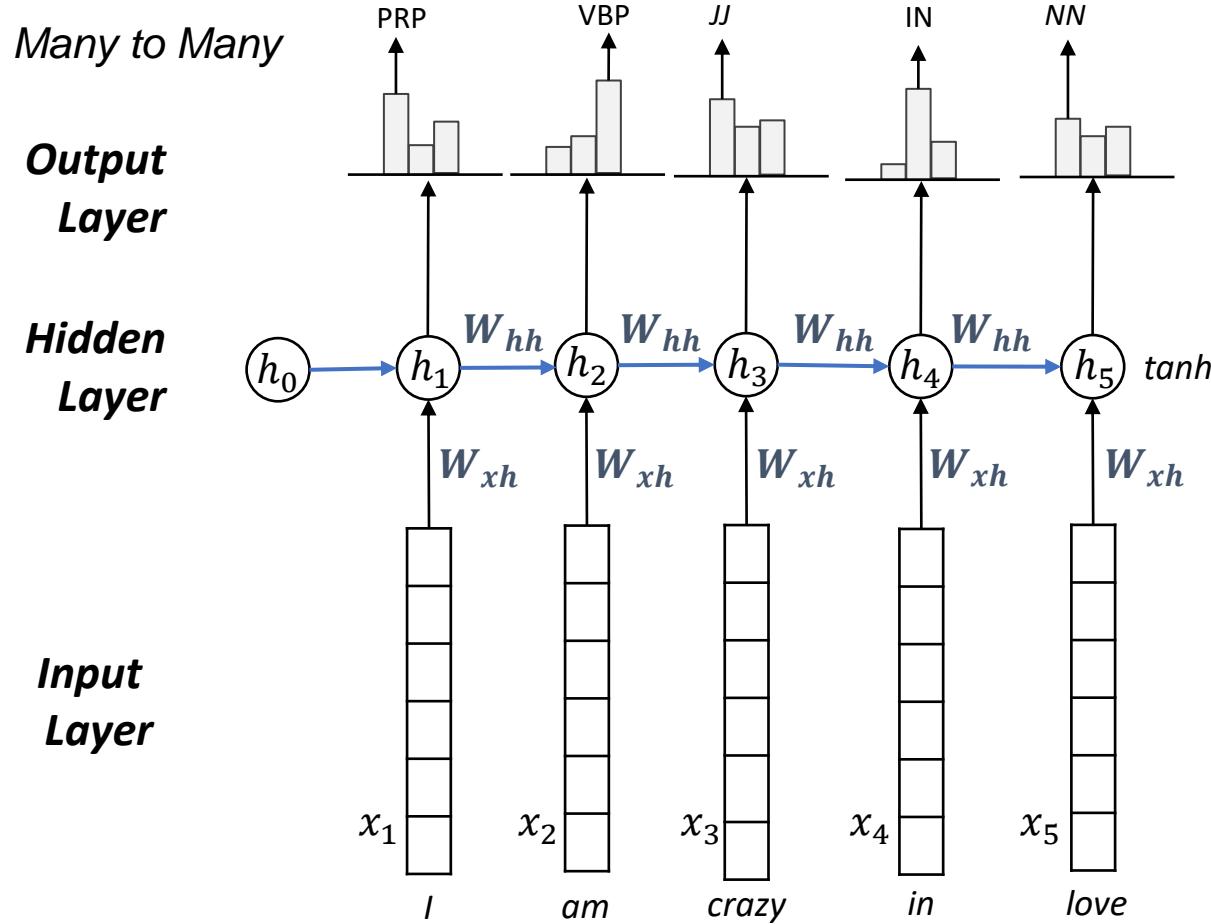
Neural Network + Memory = Recurrent Neural Network

Many to Many



Seq2Seq with Deep Learning

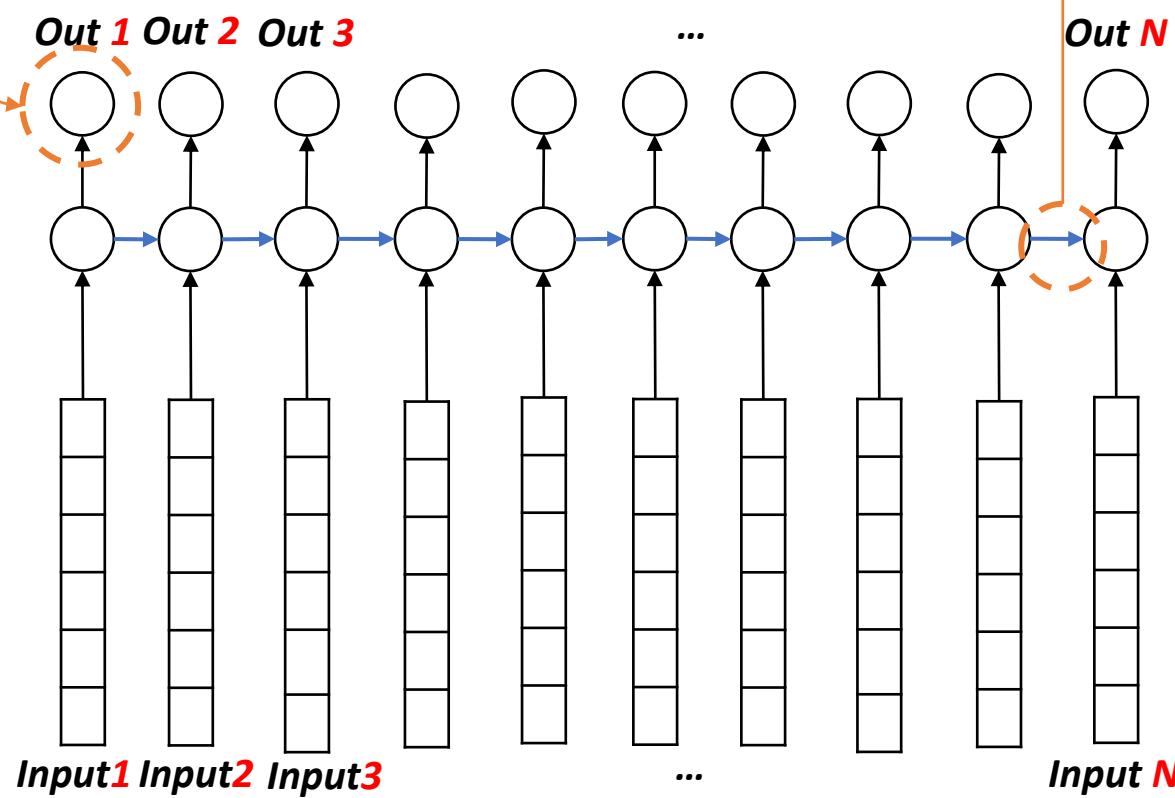
Neural Network + Memory = Recurrent Neural Network



Limitation of Vanilla RNN

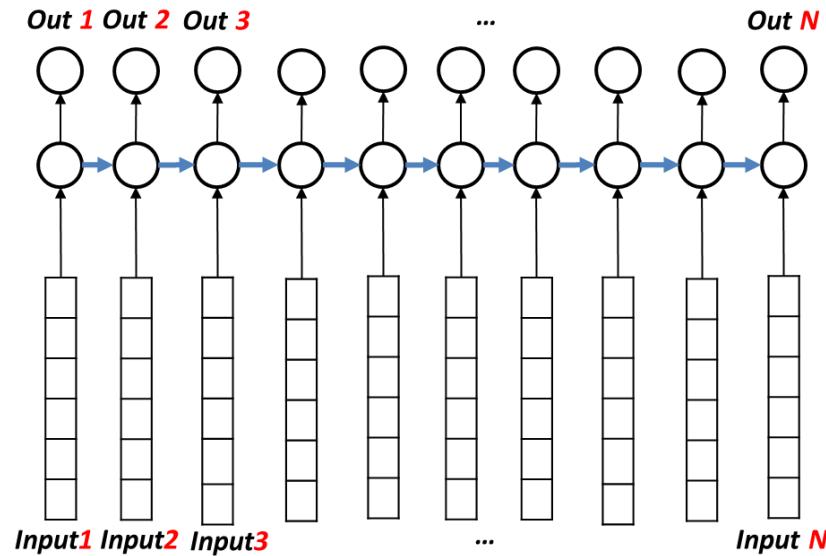
Out1 does not cover the input2 and input3

If Input1 is too far, it cannot cover the input1



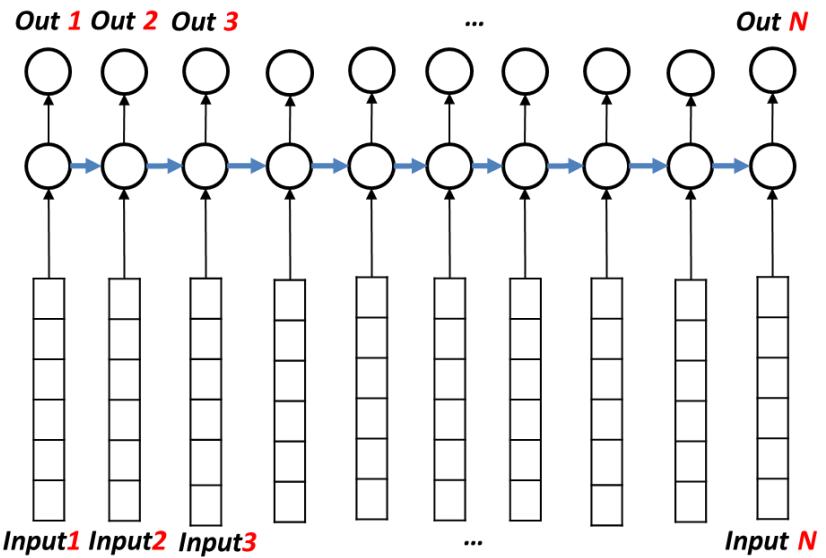
The Problem of Learning Long-Range Dependencies

Limitation of Vanilla RNN

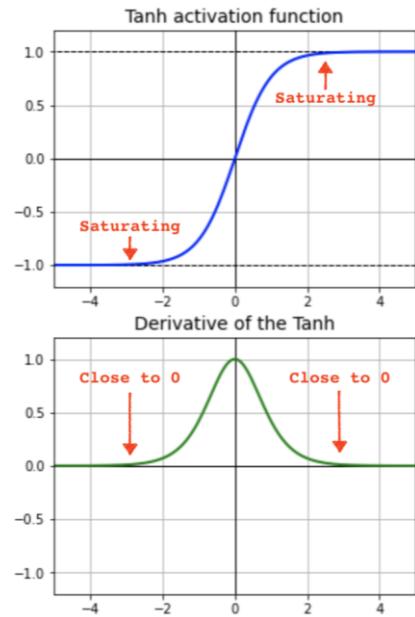
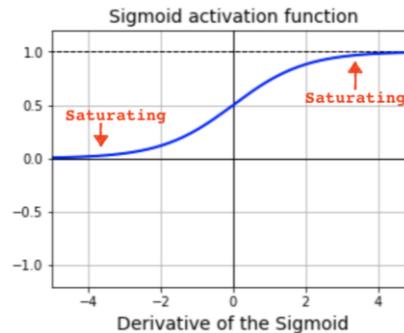


*"I grew up in Italy ... (5 more sentences)...
My grandma's house was very cosy and...
(5 more sentences)... I speak fluent ____"*

Limitation of Vanilla RNN

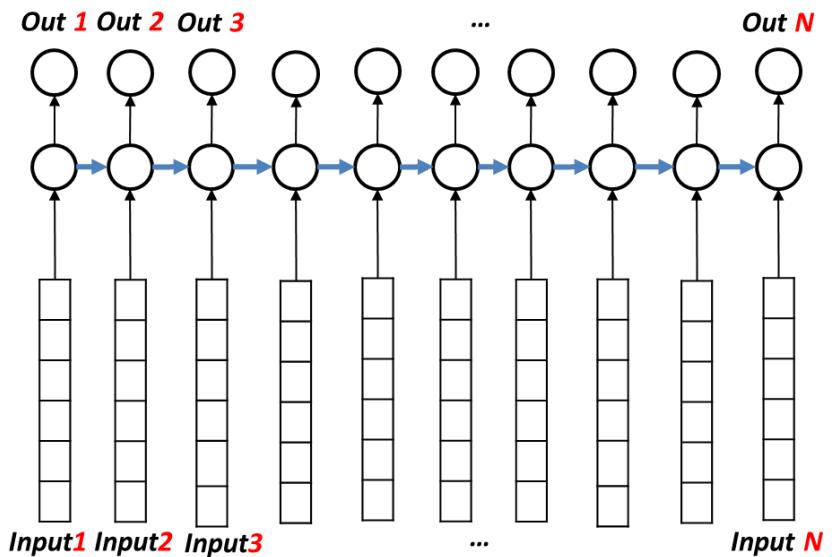


Assume that f is activation function



*"I grew up in Italy ... (5 more sentences)...
 My grandma's house was very cosy and...
 (5 more sentences)... I speak fluent _____"*

Limitation of Vanilla RNN



Assume that f is activation function

$$O_5 = f(f(f(f(f(x, W_{xh})W_{hh})W_{hh}) W_{hh}) W_{hh})$$

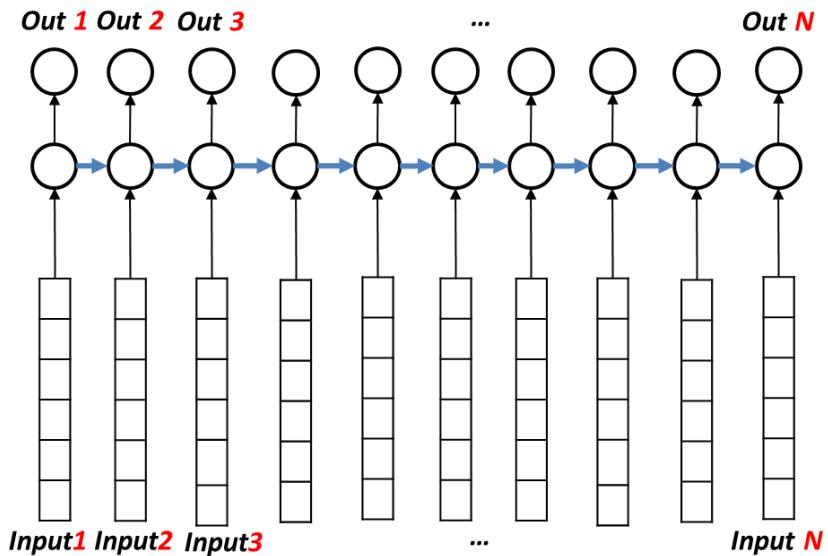
$$O_5 \approx x \cdot W_{xh} \cdot W_{hh}^4$$

Lower than 1
 Exponential decay

Higher than 1
 Exponential grow

*"I grew up in Italy ... (5 more sentences)...
 My grandma's house was very cosy and...
 (5 more sentences)... I speak fluent _____"*

Limitation of Vanilla RNN



Assume that f is activation function

$$O_5 = f(f(f(f(f(x, W_{xh})W_{hh})W_{hh}) W_{hh}) W_{hh})$$

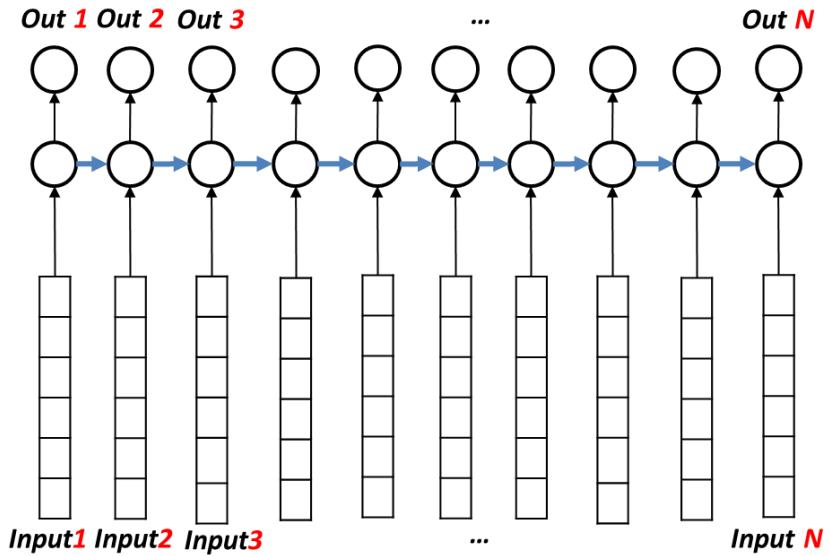
$$O_5 \approx x \cdot W_{xh} \cdot W_{hh}^4$$

Lower than 1
Exponential decay
 Higher than 1
Exponential grow

Limitation1: Vanishing Gradient Issue

During back-propagation and calculating gradients, it tends to get smaller and smaller as we keep on moving backward in the Network. This means that the neurons in the Earlier layers learn very slowly as compared to the neurons in the later layers in the Hierarchy.

Limitation of Vanilla RNN



Assume that f is activation function

$$O_5 = f(f(f(f(x, W_{xh})W_{hh})W_{hh}) W_{hh})$$

$$O_5 \approx x \cdot W_{xh} \cdot W_{hh}^4$$

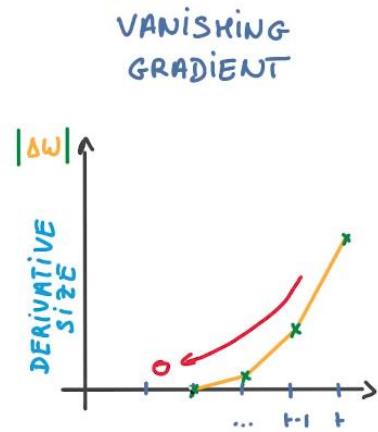
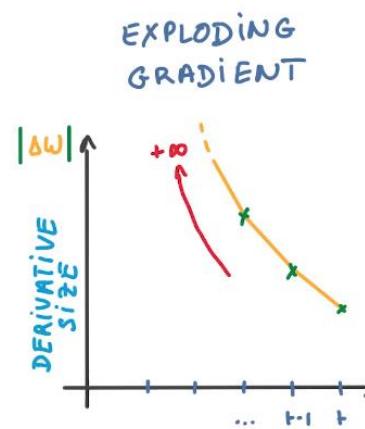
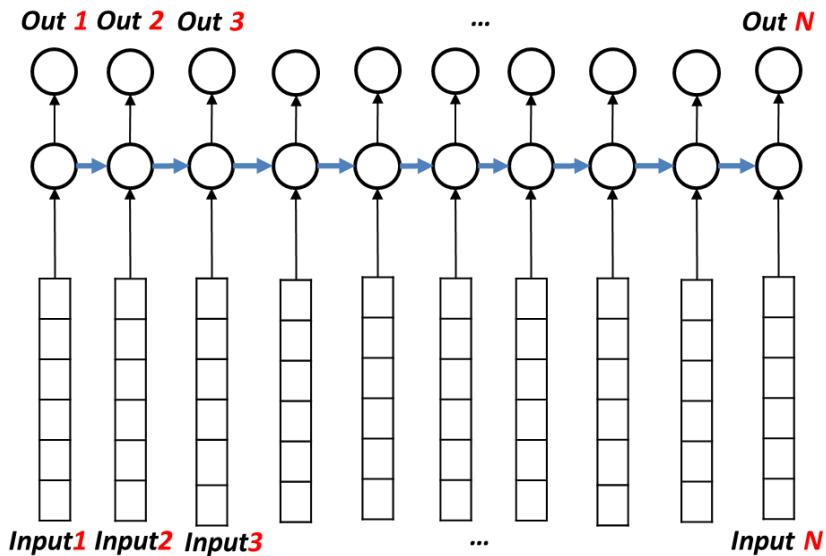
Lower than 1
Exponential decay

Higher than 1
Exponential grow

Limitation2: Exploding Gradient

In RNN, error gradients can accumulate during an update and result in very large gradients. These in turn result in large updates to the network weights, and an unstable network. At an extreme, the values of weights can become so large as to overflow and result in NaN weight values that can no longer be updated.

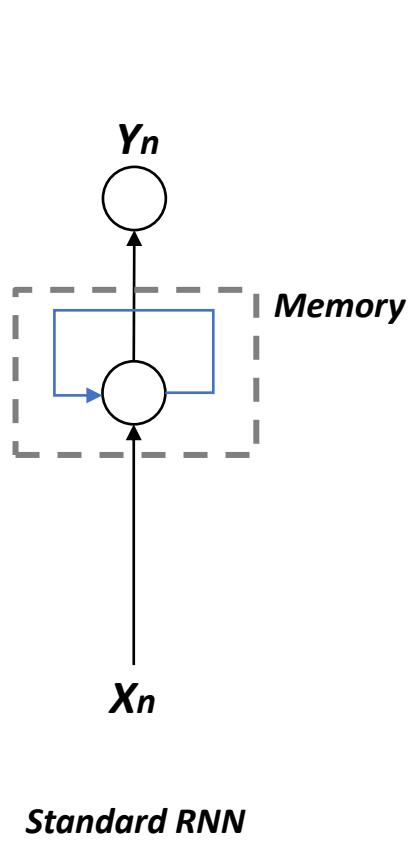
Limitation of Vanilla RNN



3

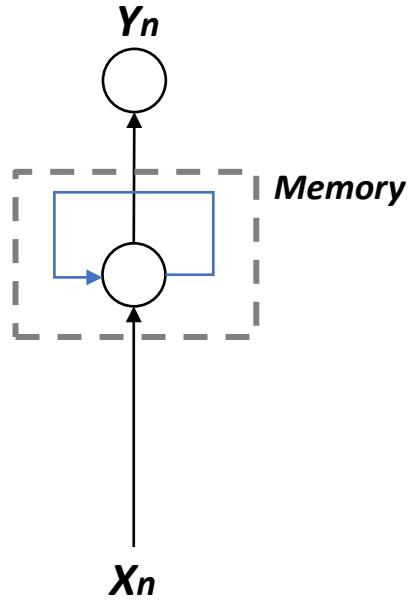
Seq2Seq with Deep Learning

LSTM (Long Short-Term Memory) - Idea

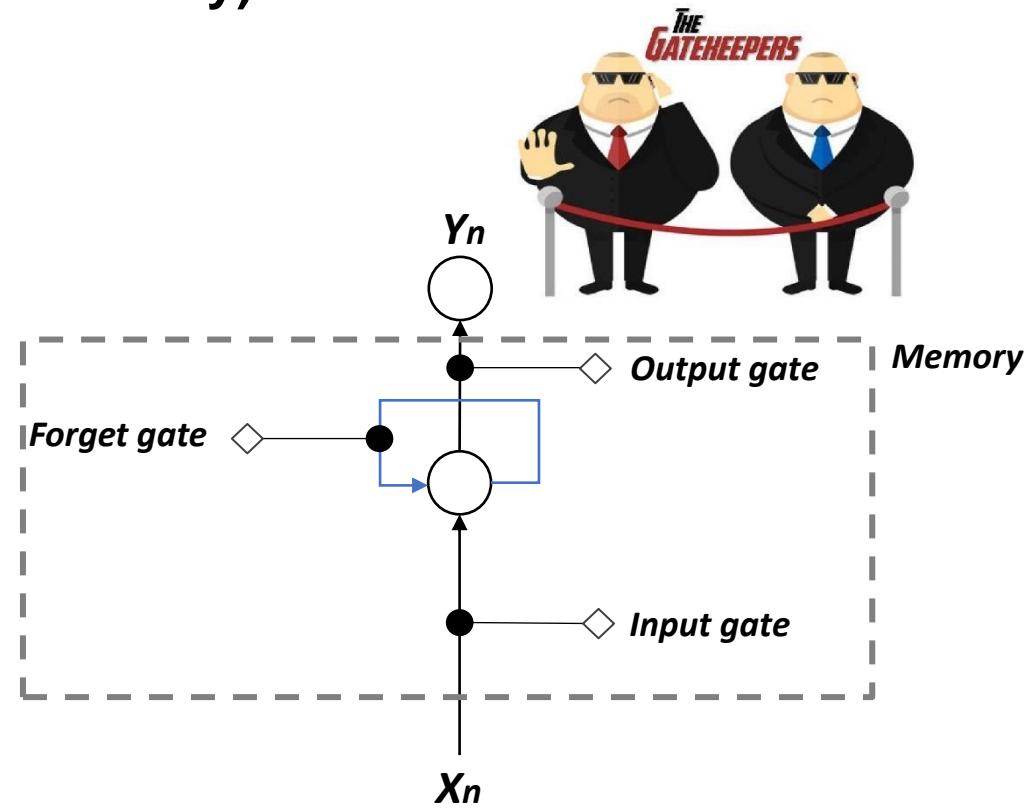


Seq2Seq with Deep Learning

LSTM (Long Short-Term Memory) - Idea



Standard RNN

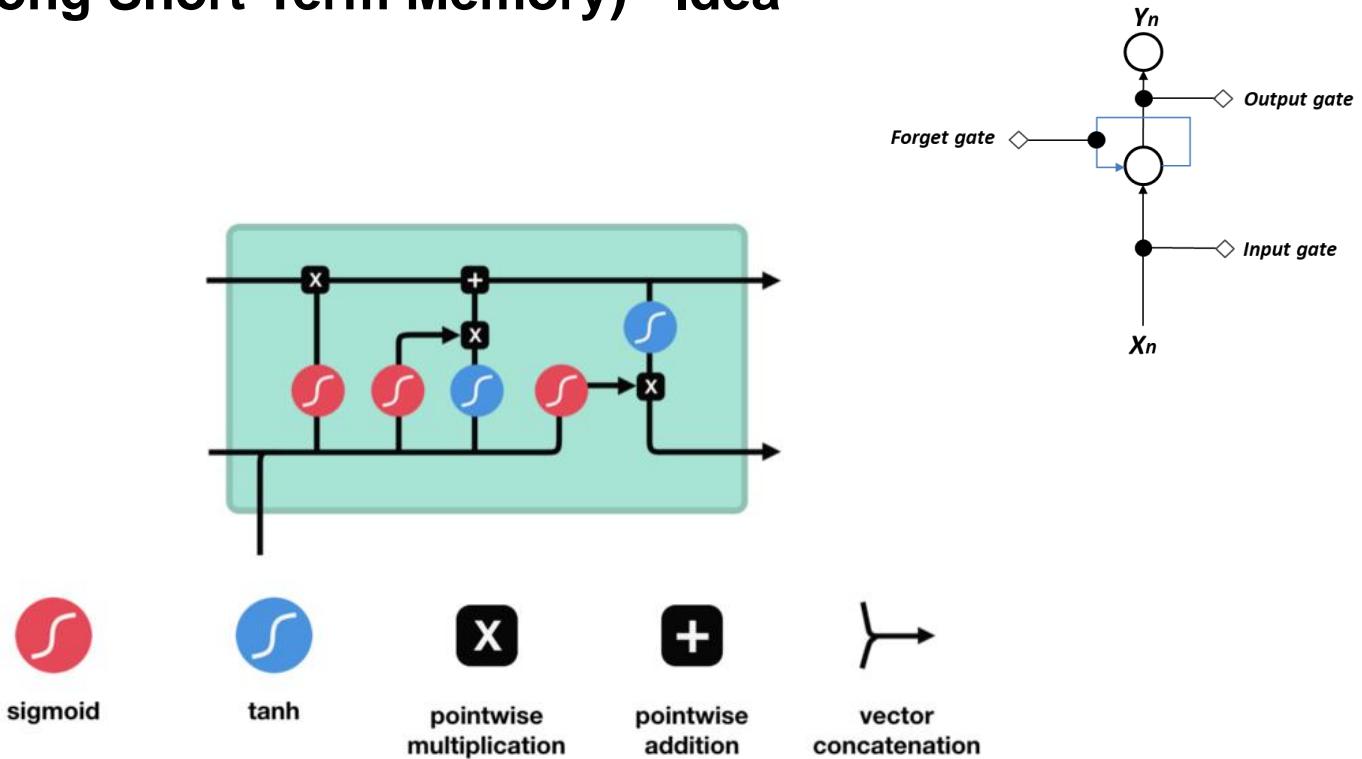


Long Short-Term Memory

3

Seq2Seq with Deep Learning

LSTM (Long Short-Term Memory) - Idea



- 4 times more parameters than RNN
- Mitigates **vanishing gradient** problem through **gating**
- Widely used and was SOTA in many sequence learning problems

State-Of-The-Art

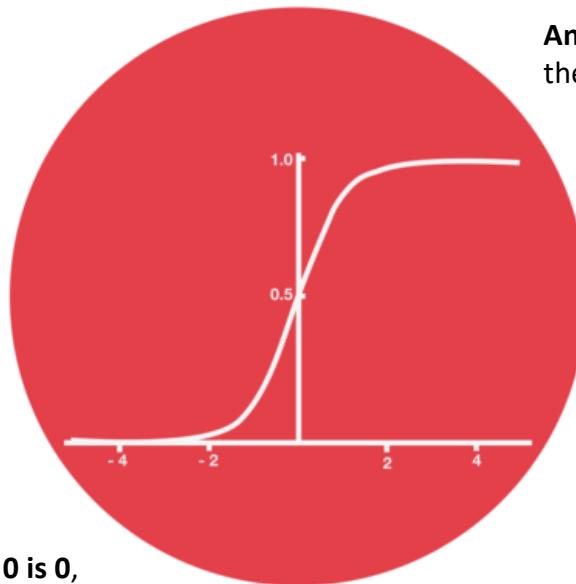
3

Seq2Seq with Deep Learning

Sigmoid activation

A sigmoid activation is similar to the tanh activation. Instead of squishing values between -1 and 1, it squishes values between 0 and 1.

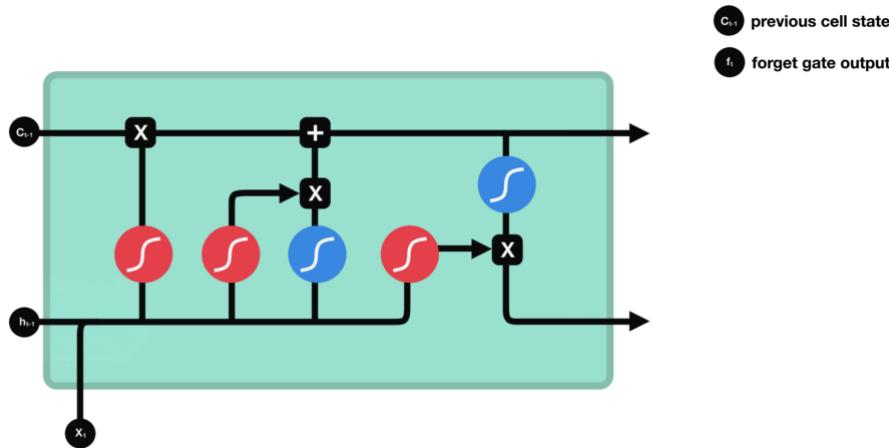
5
0.1
-0.5



Any number multiplied by 1 is the same value therefore that value stays **the same or is “kept.”**

Any number getting multiplied by 0 is 0, causing values to disappear or be “forgotten.”

LSTM (Long Short-Term Memory) – Forget Gate

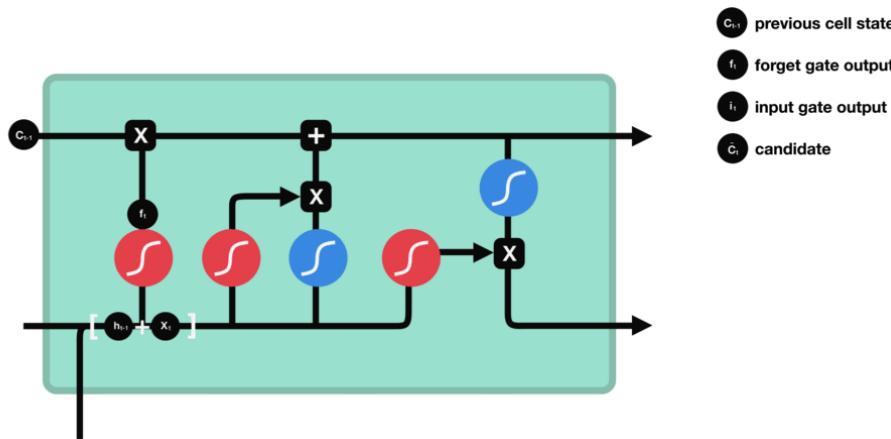


$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

Decides what information should be thrown away or kept

Information from the previous hidden state and information from the current input is passed through the **sigmoid function**. Values come out between 0 and 1. The closer to 0 means to forget, and the closer to 1 means to keep.

LSTM (Long Short-Term Memory) – Input Gate

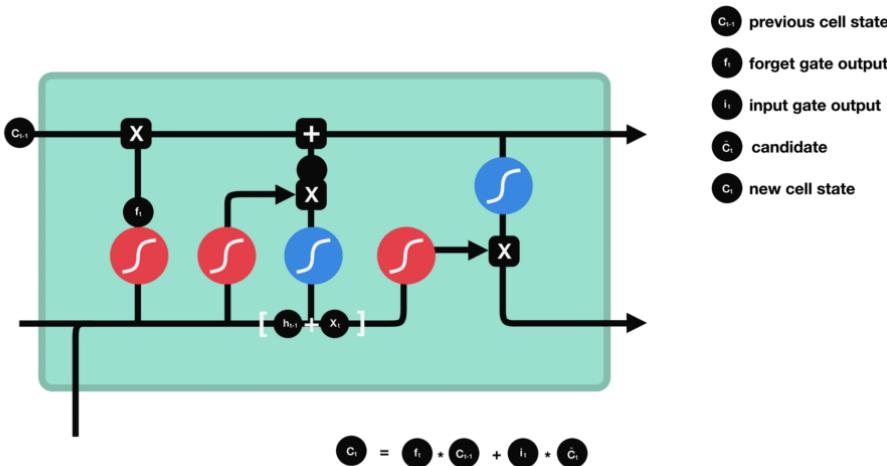


$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

$$\tilde{c}_t = \tanh(W_C[h_{t-1}, x_t] + b_C)$$

1. Pass the previous hidden state and current input into a sigmoid function
 2. Pass the hidden state and current input into the tanh function to squish values between -1 and 1 to help regulate the network
 3. Multiply the tanh output with the sigmoid output
- *sigmoid output will decide which information is important to keep from the tanh output

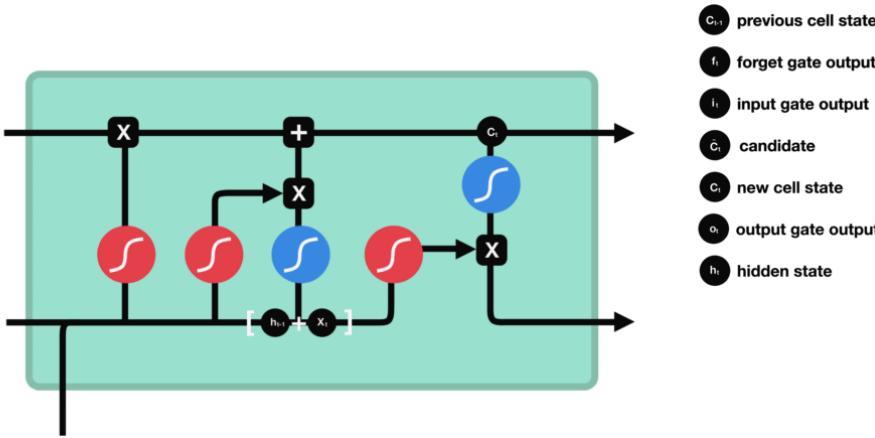
LSTM (Long Short-Term Memory) – Cell States



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- the cell state gets pointwise multiplied by the forget vector
- take the output from the input gate and do a pointwise addition which updates the cell state to new values that the neural network finds relevant
- That gives us our new cell state

LSTM (Long Short-Term Memory) – Output Gate



- c_{t-1} previous cell state
- f_t forget gate output
- i_t input gate output
- \tilde{c}_t candidate
- c_t new cell state
- o_t output gate output
- h_t hidden state

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

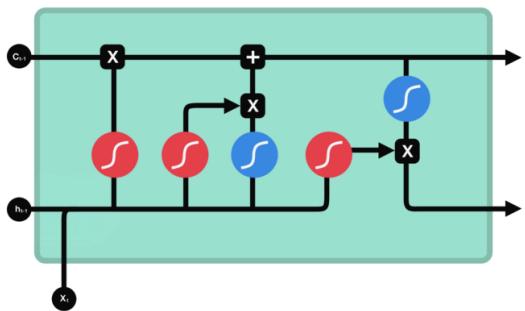
$$h_t = o_t * \tanh(c_t)$$

decides what the next hidden state should be.

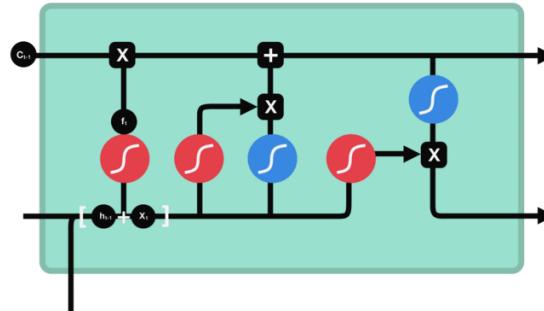
- pass the previous hidden state and the current input into a sigmoid function
- pass the newly modified cell state to the tanh function
- multiply the tanh output with the sigmoid output to decide what information the hidden state should carry

LSTM (Long Short-Term Memory) - Overall

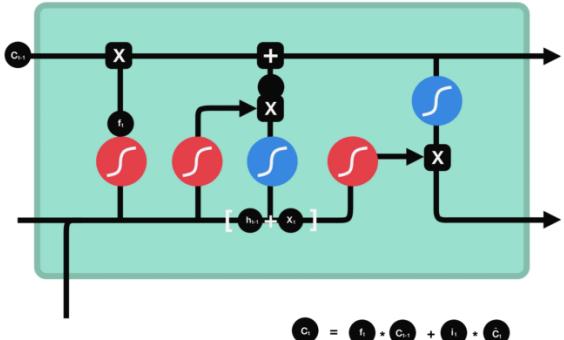
Forget gate



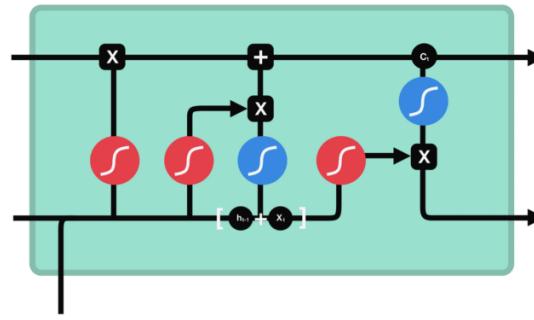
Input gate



Cell state



Output gate



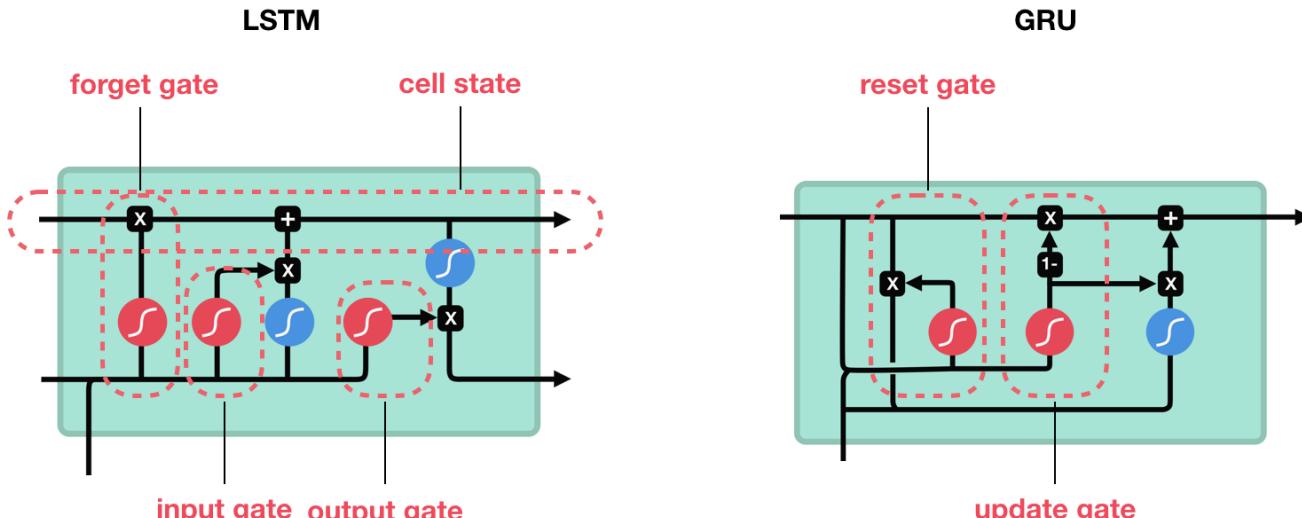
c_{t-1} previous cell state
 f_t forget gate output

c_{t-1} previous cell state
 f_t forget gate output
 i_t input gate output
 \tilde{c}_t candidate

c_{t-1} previous cell state
 f_t forget gate output
 i_t input gate output
 \tilde{c}_t candidate
 c_t new cell state

c_{t-1} previous cell state
 f_t forget gate output
 i_t input gate output
 \tilde{c}_t candidate
 c_t new cell state
 o_t output gate output
 h_t hidden state

Gated Recurrent Unit



sigmoid

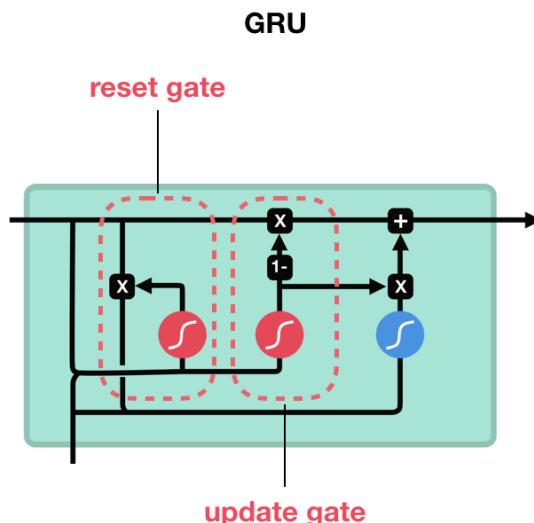


tanh

pointwise
multiplicationpointwise
additionvector
concatenation

Gated Recurrent Unit

- GRU first computes an **update gate** based on **current input word vector** and **hidden state**
- Compute reset gate similarly but with different weights
 - If reset gate unit is ~ 0 , then this ignores previous memory and only stores the new word information
- Final memory at time step combines current and previous time steps



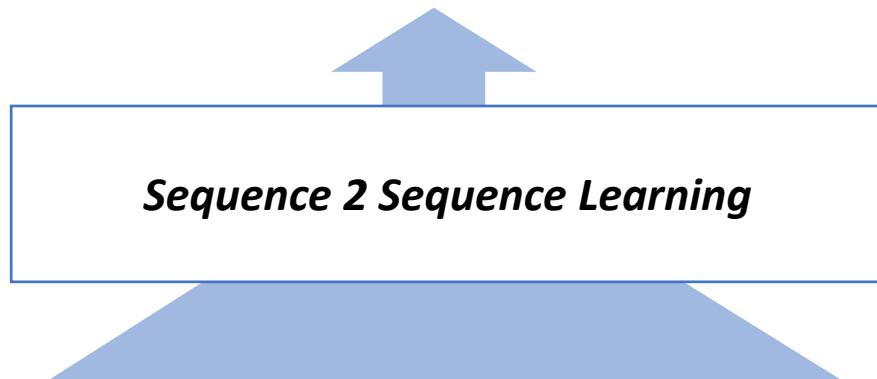
3

Seq2Seq Modelling

Seq2Seq – PoS tagger

ADV VERB DET NOUN NOUN

Output: Part of Speech

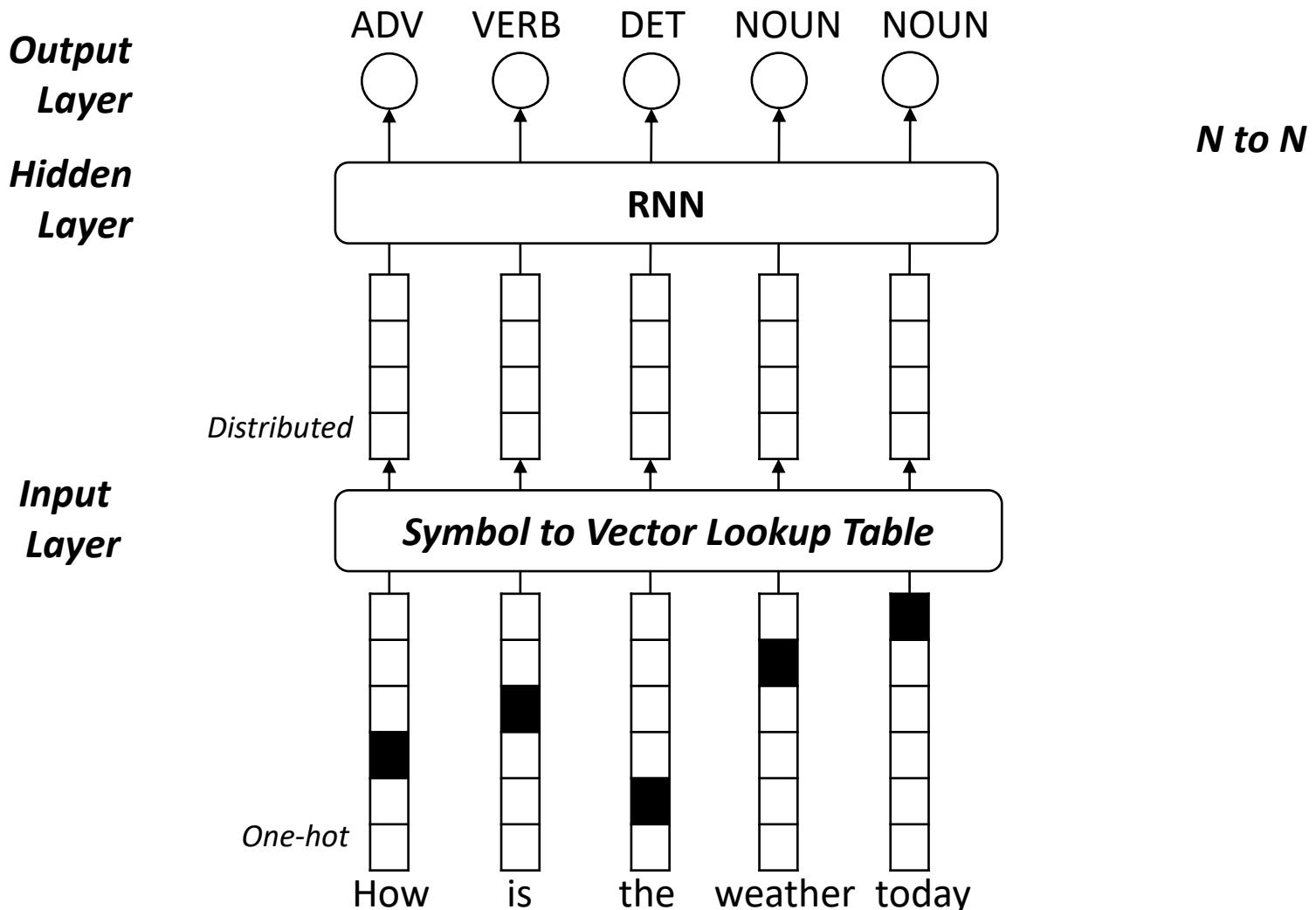


How is the weather today

Input: Text

3 Seq2Seq Modelling

Sequence Modelling for POS Tagging



0 LECTURE PLAN

Lecture 4: Word Classification and Machine Learning 2

1. Machine Learning and NLP: Finish
2. Seq2Seq Learning
3. Seq2Seq Deep Learning
 1. RNN (Recurrent Neural Network)
 2. LSTM (Long Short-Term Memory)
 3. GRU (Gated Recurrent Unit)
4. **Data Transformation for Deep Learning NLP**
5. Next Week Preview
 - Natural Language Processing Stack

ImageNet: Image Classification

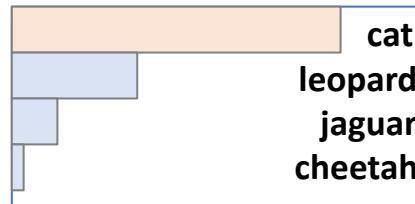
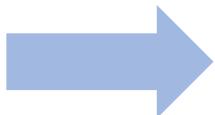
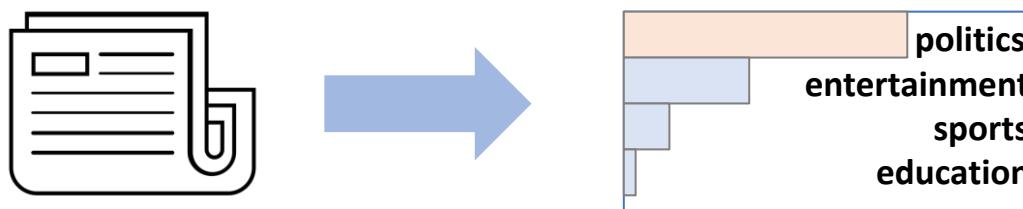


Image Pixel

Topic Classification



News Articles

Visual Question Answering



Visual Question Answering



What color of the shirt does he wear

Submit

Predicted top-5 answers with confidence:

orange

99.999%

yellow

0.001%

orange and
white

0.000%

yellow and
orange
orange and
black

0.000%

Visual Question Answering



Where is he sitting

Submit

Predicted top-5 answers with confidence:

couch

71.669%

chair

21.119%

sofa

2.730%

living room

1.376%

room

1.276%

Visual Question Answering



Why is he surprised

Submit

Predicted top-5 answers with confidence:

playing

36.734%

game

13.713%

game

5.481%

playing
video games

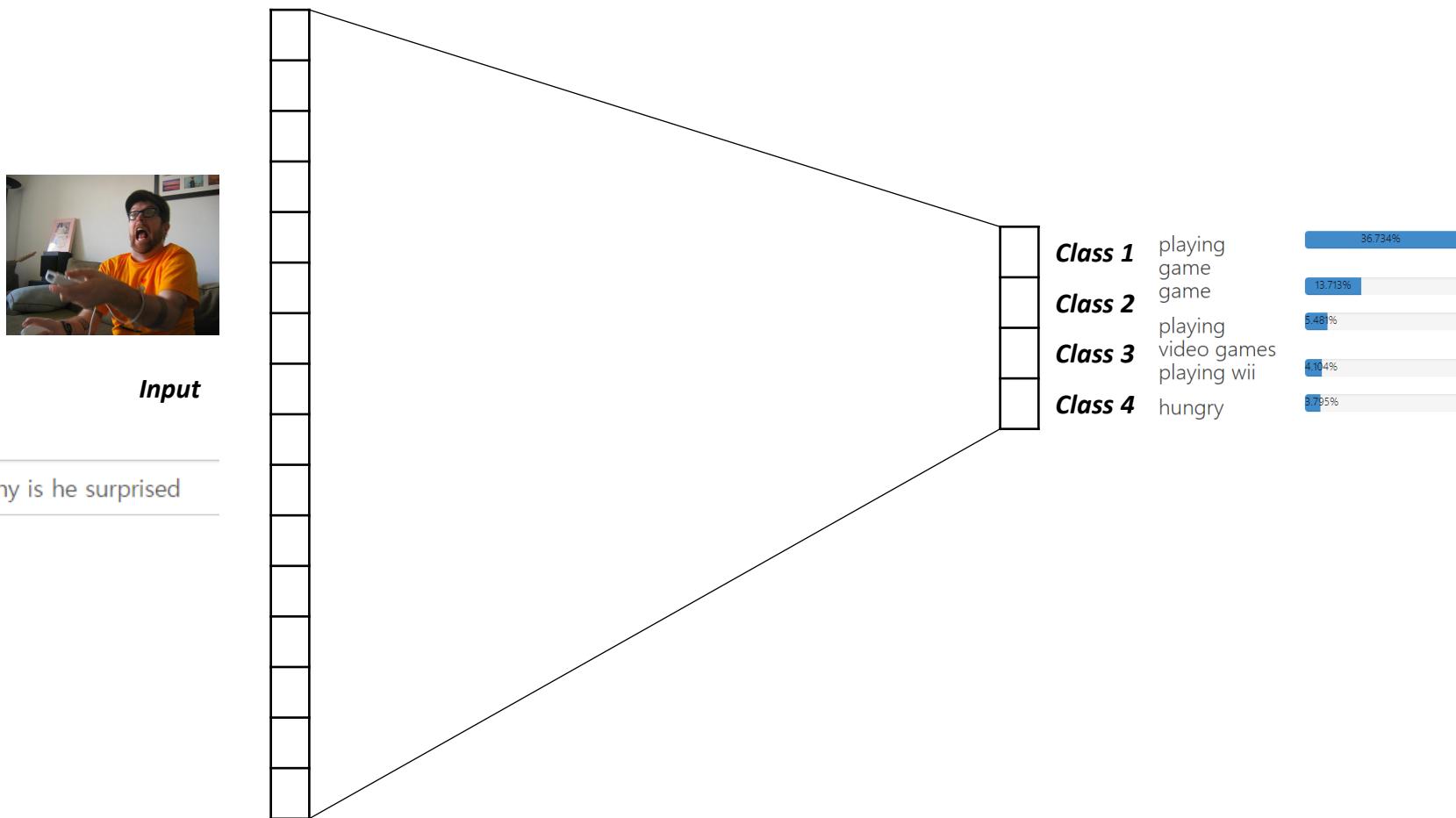
4.104%

playing wii

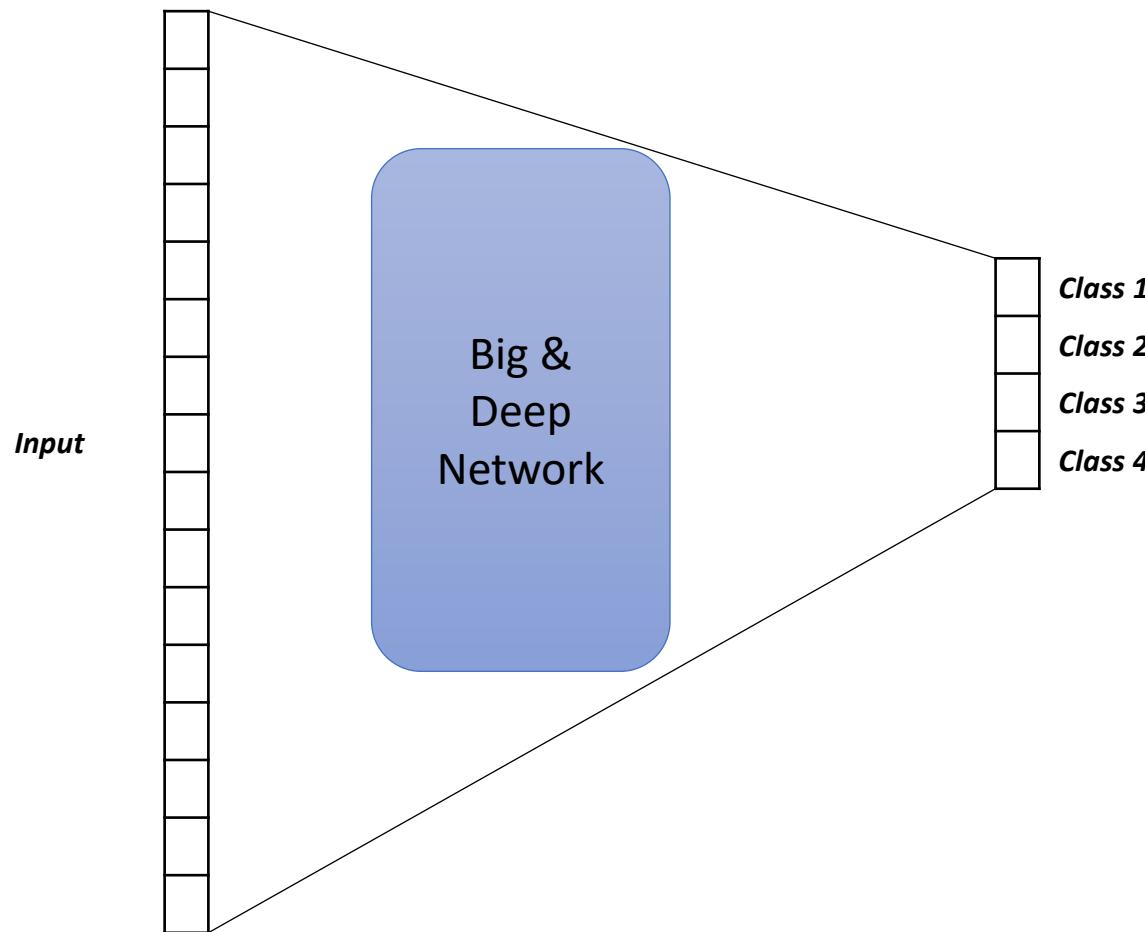
3.795%

hungry

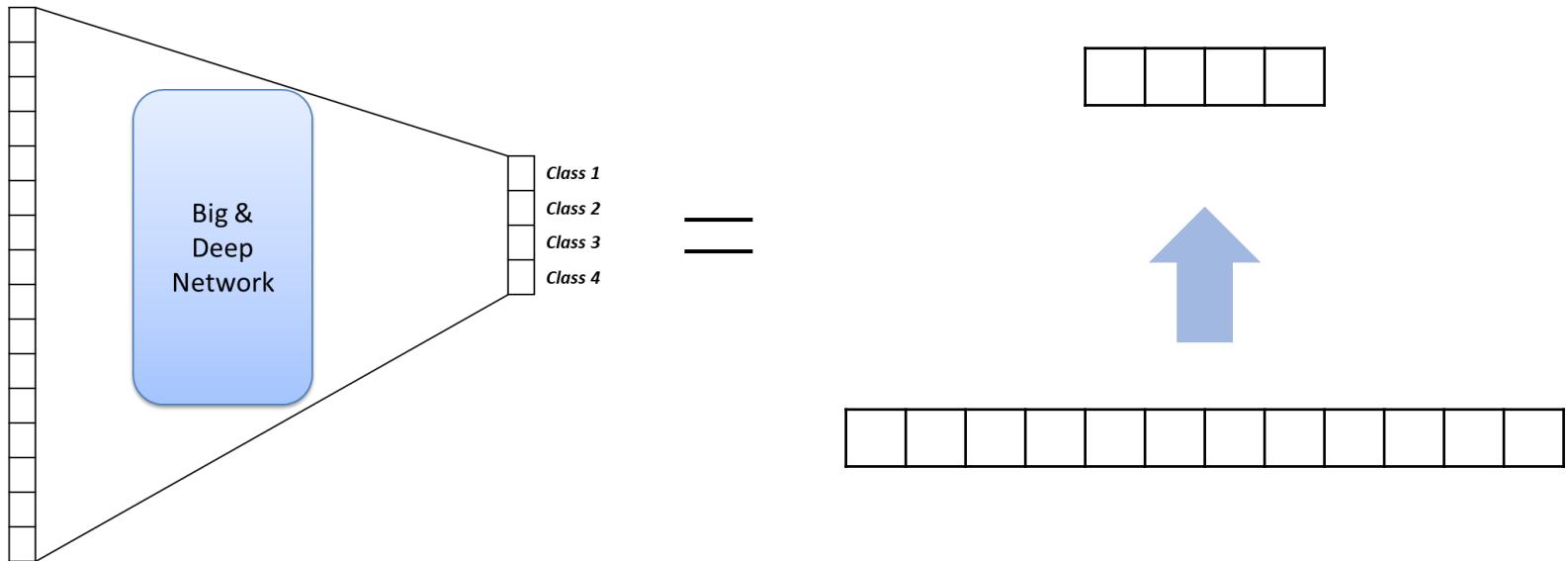
Classification Formulation



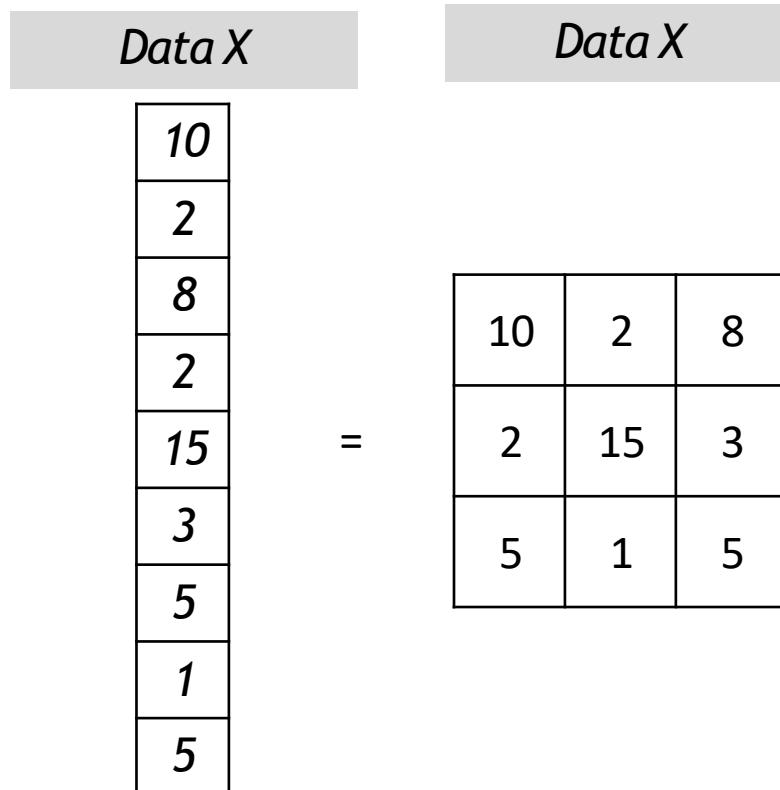
Classification Formulation



Classification



Graphical Notation for Data

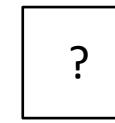
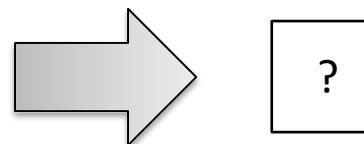


V to 1

10
2
8
2
15
3
5
1
5

=

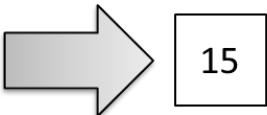
10	2	8
2	15	3
5	1	5



V to 1 – Simple Method

center one

10	2	8
2	15	3
5	1	5



15

average

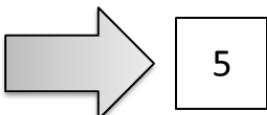
10	2	8
2	15	3
5	1	5



5.6

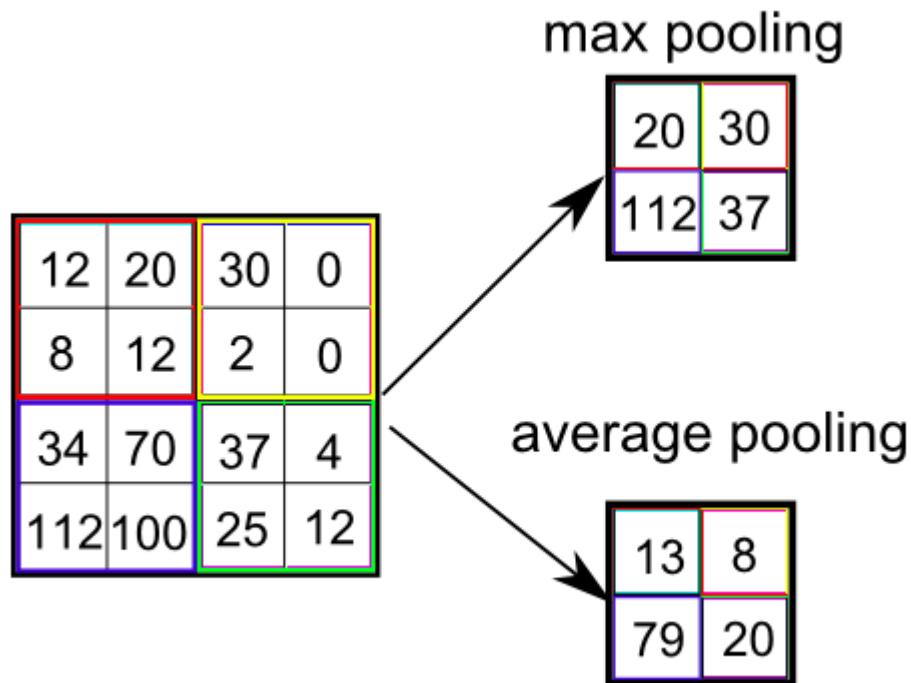
median

10	2	8
2	15	3
5	1	5



5

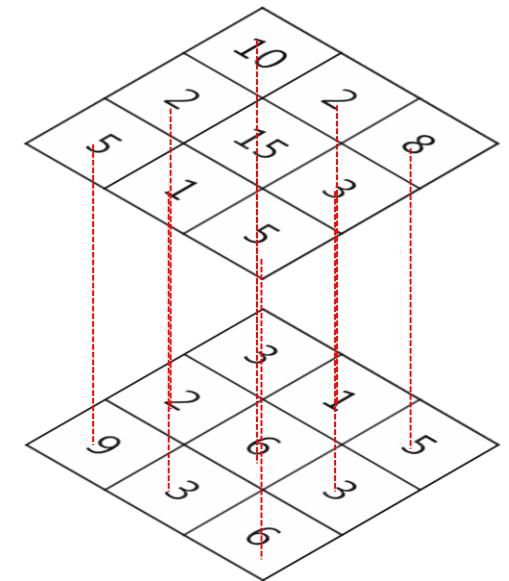
V to 1 – Simple Method



V to 1 – Weighted Method

Weighted Sum								
Value			Weight					
10	2	8	3	1	5	2	15	3
2	15	3	2	6	3	9	3	6
5	1	5	9	3	6			

→ 253

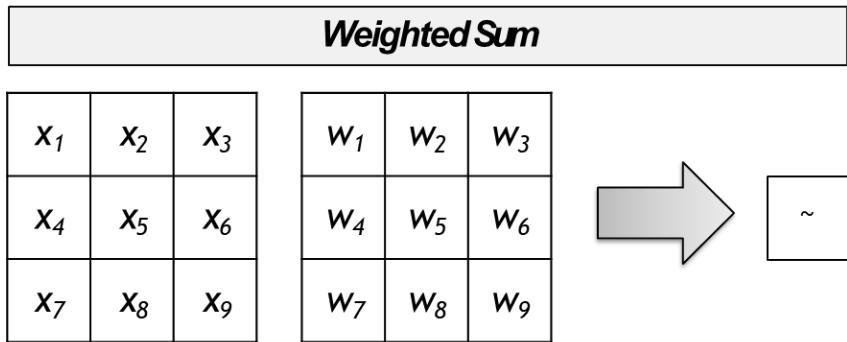


Element-wise multiplication

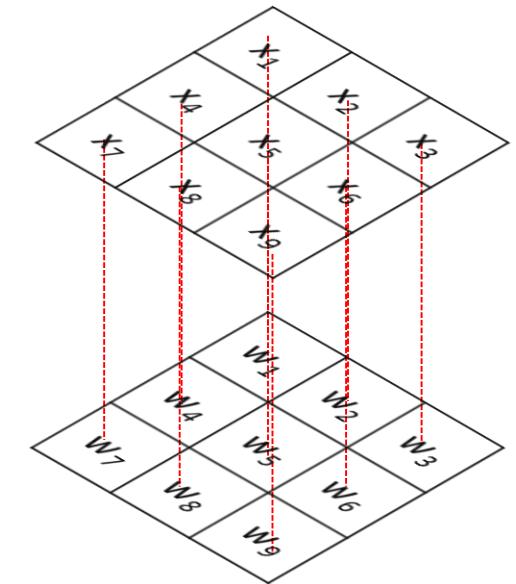
Weighted Average								
Value			Weight					
10	2	8	3/9	1/9	5/9	2/9	6/9	3/9
2	15	3	9/9	3/9	6/9	9/9	3/9	6/9
5	1	5						

→ 6.65

V to 1 – General Form



$$v = x_1 * w_1 + x_2 * w_2 + \dots + x_9 * w_9$$



Element-wise multiplication

V to 1 – Linear Algebra

[9x1] matrix

w_1
w_2
w_3
w_4
w_5
w_6
w_7
w_8
w_9

[1 x 9] matrix

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
-------	-------	-------	-------	-------	-------	-------	-------	-------

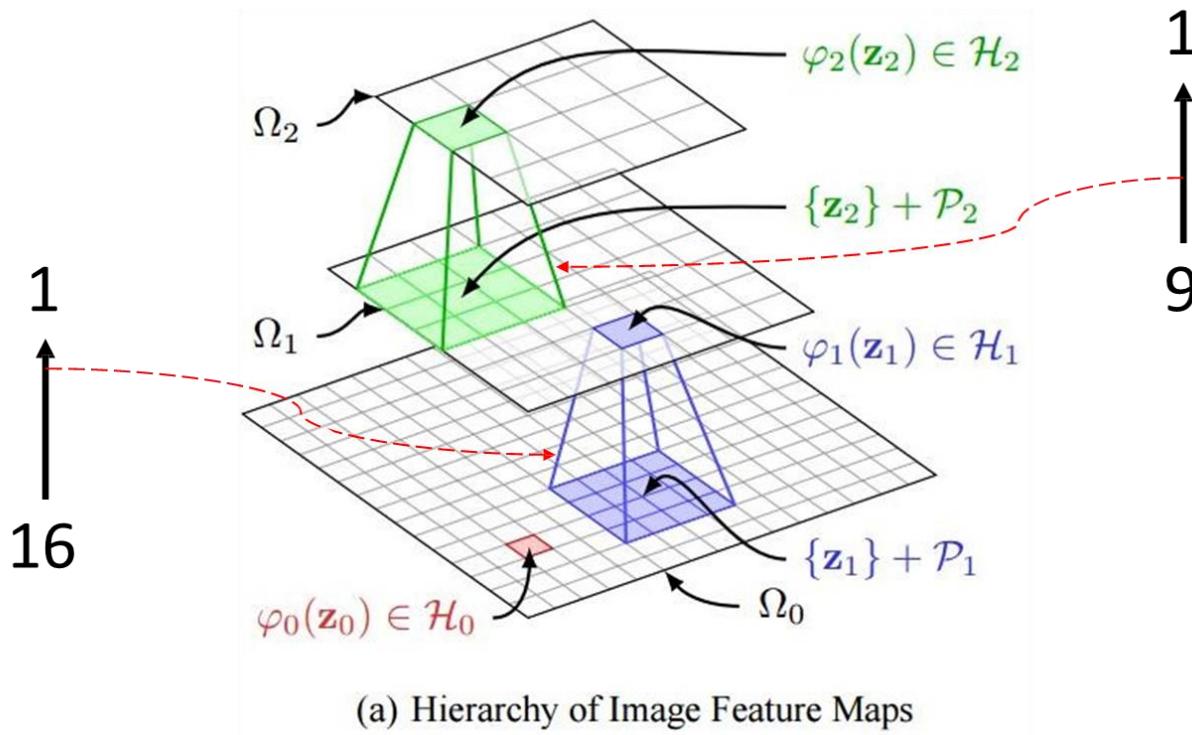
X

[1x1] matrix

$$\sum_i^9 x_i * w_i$$

=

Convolution Neural Network (1)



Data Abstraction

Convolution Neural Network (2)

1	0	1
0	1	0
1	0	1

filter

1 <small>$\times 1$</small>	1 <small>$\times 0$</small>	1 <small>$\times 1$</small>	0	0
0 <small>$\times 0$</small>	1 <small>$\times 1$</small>	1 <small>$\times 0$</small>	1	0
0 <small>$\times 1$</small>	0 <small>$\times 0$</small>	1 <small>$\times 1$</small>	1	1
0	0	1	1	0
0	1	1	0	0

Image

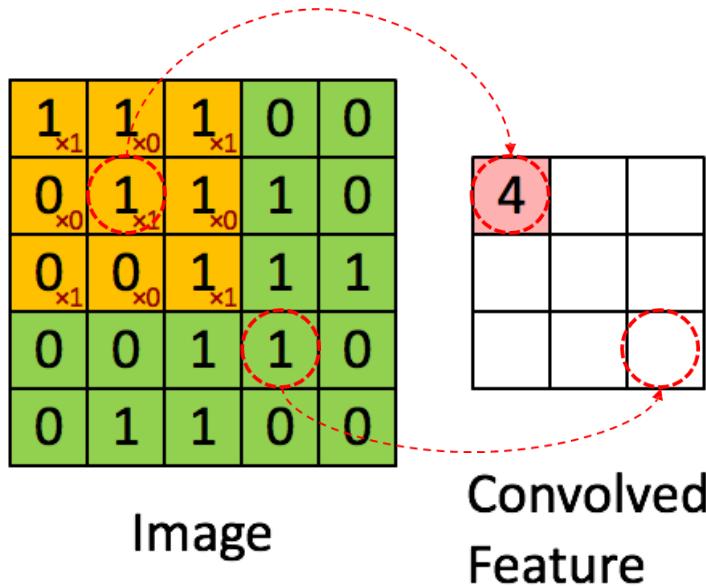
4		

Convolved
Feature

Convolution Neural Network (2)

1	0	1
0	1	0
1	0	1

filter



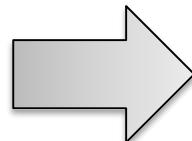
V to V'

$V = 9$

10	2	8
2	15	3
5	1	5

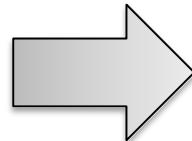
$V' = 2$

?
?

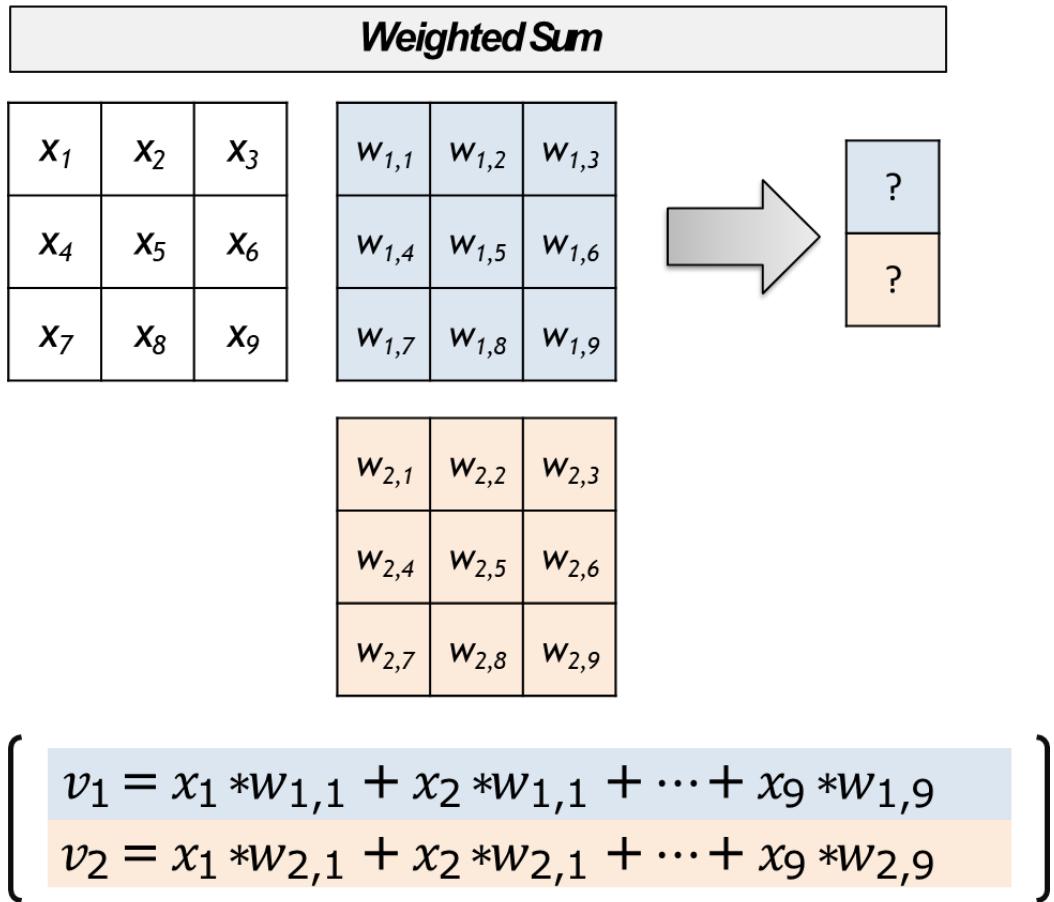


10	2	8	2	15	3	5	1	5
----	---	---	---	----	---	---	---	---

?	?
---	---



V to V' – generalized method



V to V' – generalized method

Weighted Sum

[9x2] matrix

$w_{1,1}$	$w_{2,1}$
$w_{1,2}$	$w_{2,2}$
$w_{1,3}$	$w_{2,3}$
$w_{1,4}$	$w_{2,4}$
$w_{1,5}$	$w_{2,5}$
$w_{1,6}$	$w_{2,6}$
$w_{1,7}$	$w_{2,7}$
$w_{1,8}$	$w_{2,8}$
$w_{1,9}$	$w_{2,9}$

[1 x 9] matrix

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
-------	-------	-------	-------	-------	-------	-------	-------	-------

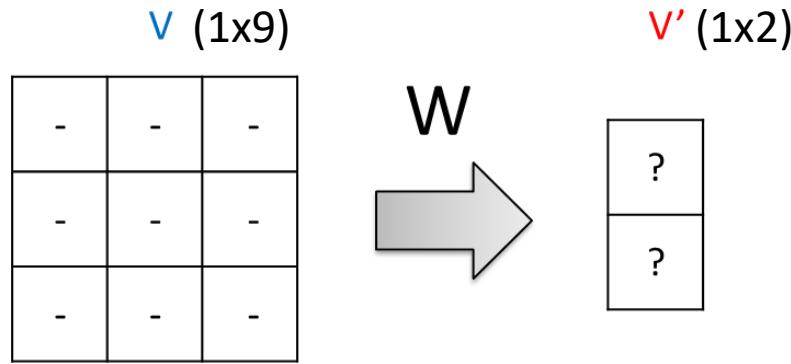
X

[1x2] matrix

$$= \left[\sum_i^9 x_i * w_{1,i} , \sum_i^9 x_i * w_{2,i} \right]$$

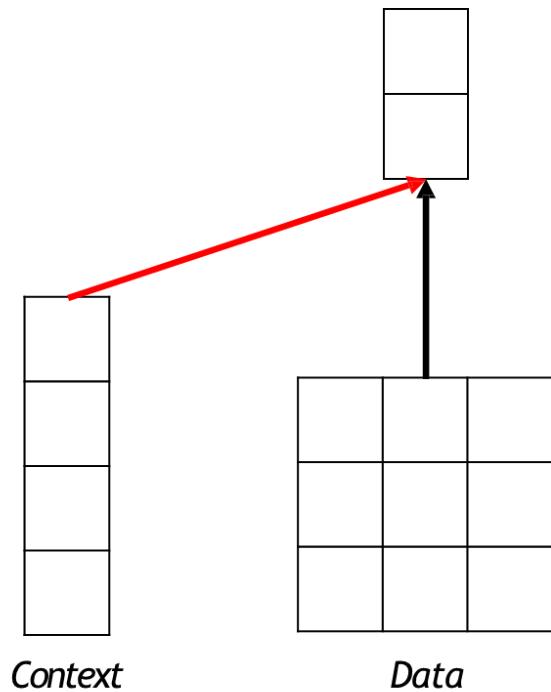
Fully Connected Network

V to V' – Projection Notation

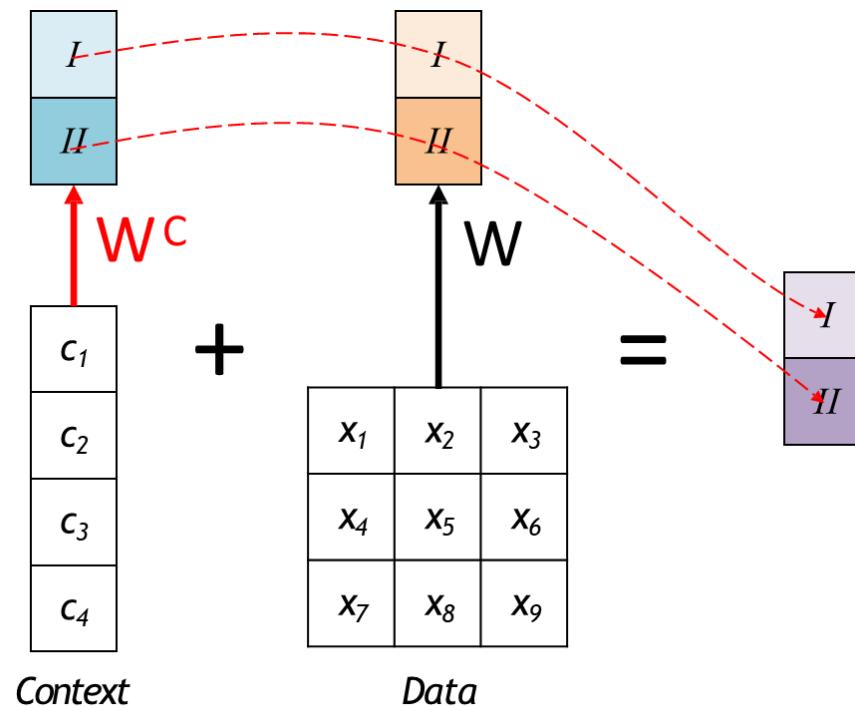


$$W = \textcolor{blue}{v} \left[\begin{array}{c} \textcolor{red}{V}' \\ \vdots \end{array} \right]$$

V to V' – Projection with Context (1)



V to V' – Projection with Context (2)



V to V' with Context - Linear Algebra

[1 x 9] matrix

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
-------	-------	-------	-------	-------	-------	-------	-------	-------

[9x2] matrix

$w_{1,1}$	$w_{2,1}$
$w_{1,2}$	$w_{2,2}$
$w_{1,3}$	$w_{2,3}$
$w_{1,4}$	$w_{2,4}$
$w_{1,5}$	$w_{2,5}$
$w_{1,6}$	$w_{2,6}$
$w_{1,7}$	$w_{2,7}$
$w_{1,8}$	$w_{2,8}$
$w_{1,9}$	$w_{2,9}$

[1x2] matrix

X

$$= \left(\sum_i^9 x_i * w_{1,i}, \sum_i^9 x_i * w_{2,i} \right)$$

I

II

[1 x 4] matrix

c_1	c_2	c_3	c_4
-------	-------	-------	-------

X

$w^C_{1,1}$	$w^C_{2,1}$
$w^C_{1,2}$	$w^C_{2,2}$
$w^C_{1,3}$	$w^C_{2,3}$
$w^C_{1,4}$	$w^C_{2,4}$

[1x2] matrix

$$= \left(\sum_i^4 c_i * w_{1,i}^C, \sum_i^4 c_i * w_{2,i}^C \right)$$

I

II

V to V' with Context - Linear Algebra (Simplified)

$[1 \times (9+4)]$ matrix

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	c_1	c_2	c_3	c_4
-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

$[(9+4) \times 2]$ matrix

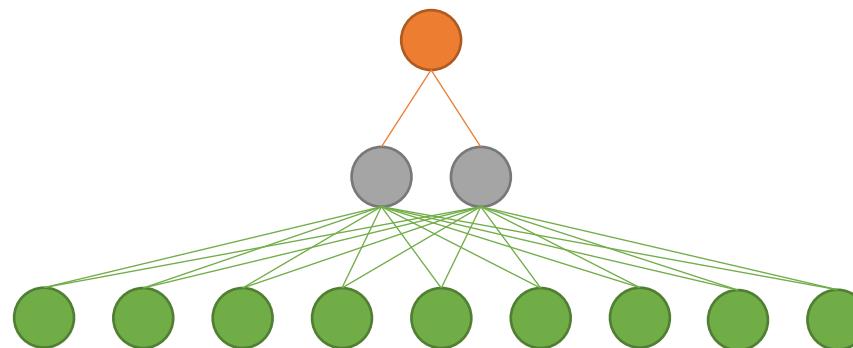
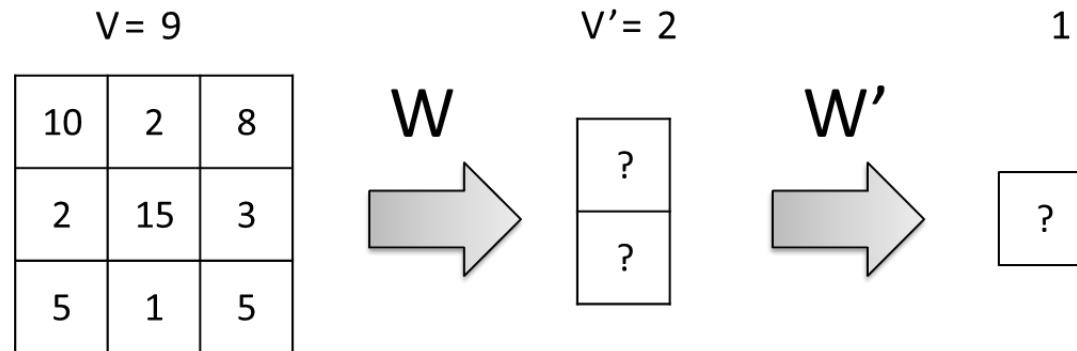
$w_{1,1}$	$w_{2,1}$
$w_{1,2}$	$w_{2,2}$
$w_{1,3}$	$w_{2,3}$
$w_{1,4}$	$w_{2,4}$
$w_{1,5}$	$w_{2,5}$
$w_{1,6}$	$w_{2,6}$
$w_{1,7}$	$w_{2,7}$
$w_{1,8}$	$w_{2,8}$
$w_{1,9}$	$w_{2,9}$
$w^C_{1,1}$	$w^C_{2,1}$
$w^C_{1,2}$	$w^C_{2,2}$
$w^C_{1,3}$	$w^C_{2,3}$
$w^C_{1,4}$	$w^C_{2,4}$

$[1 \times 2]$ matrix

$$= \left[\begin{array}{c|c} \sum_i^9 x_i * w_{1,i} & \sum_i^9 x_i * w_{2,i} \\ + \sum_i^4 c_i * w_{1,i}^C & + \sum_i^4 c_i * w_{2,i}^C \end{array} \right]$$

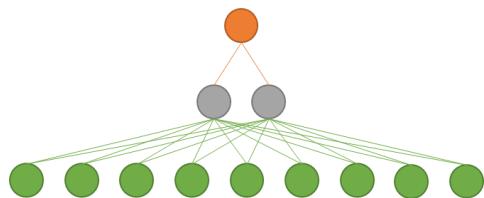
I

II

$V \rightarrow V' \rightarrow 1$ 

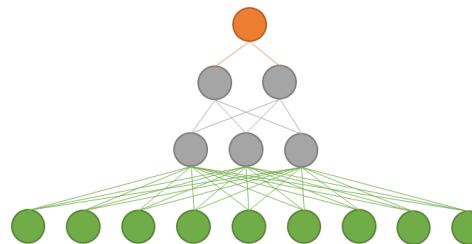
Data Transformation for Deep Learning NLP

$V \rightarrow V' \rightarrow 1$



Single Layer

$V \rightarrow V' \rightarrow 1$

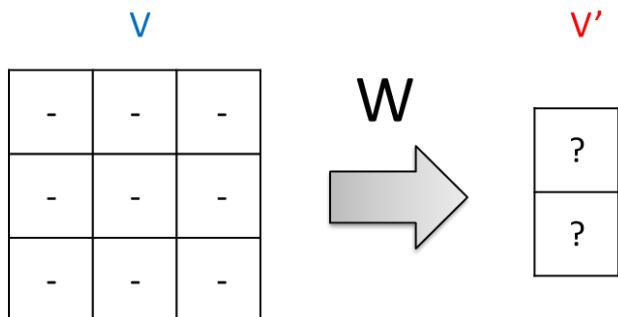


Multilayer

$V \rightarrow V' \rightarrow V'' \rightarrow 1$

Seq2Seq Encoding

*Single Item
Summarisation*



*Multiple Item
Summarisation*

?

Multiple Item Summarisation

10	2	8
2	15	3
5	1	5

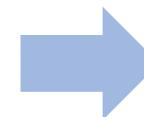
Data 1

13	4	8
4	5	2
1	45	31

Data 2

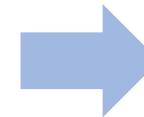
6	3	4
1	7	1
3	4	0

Data 3



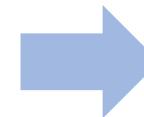
?	?	?
?	?	?
?	?	?

V



?
?

V'



?

1

Data Transformation for Deep Learning NLP

V_s to V'

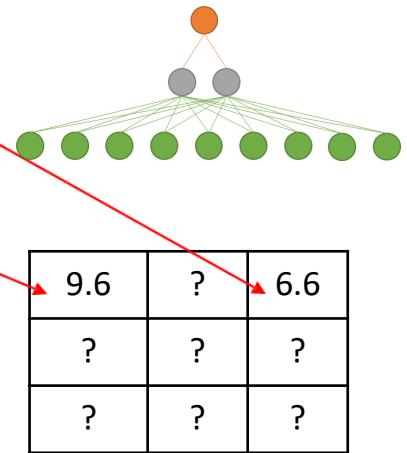
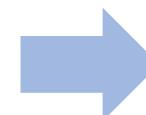
10	2	8
2	15	3
5	1	5

$$(10+ 13+ 6)/3$$

13	4	8
4	5	2
1	45	31

$$(8+8+4)/3$$

6	3	4
1	7	1
3	4	0



Element-wise Average

Vs to V'

10	2	8
2	15	3
5	1	5

13	4	8
4	5	2
1	45	31

6	3	4
1	7	1
3	4	0

$$w^1 \stackrel{x}{=} 0.2$$



2	0.4	1.6
0.4	3	0.6
1	0.2	1.0

$$w^2 \stackrel{x}{=} 0.4$$



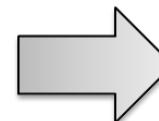
5.2	1.6	3.2
1.6	2	0.8
0.4	18	12.4

$$w^3 \stackrel{x}{=} 0.4$$



2.4	1.2	1.6
0.4	2.8	0.4
1.2	1.6	0

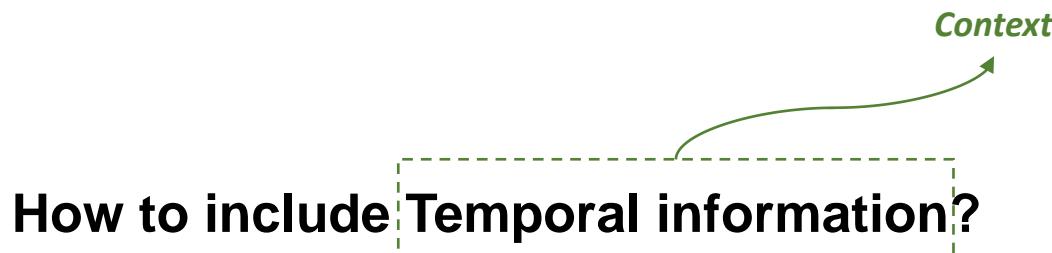
Element-wise multiplication



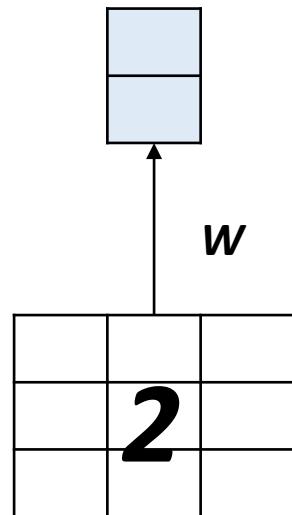
9.6	3.2	6.4
2.4	7.8	1.8
2.6	19.8	13.4

Element-wise summation

Temporal Summarisation

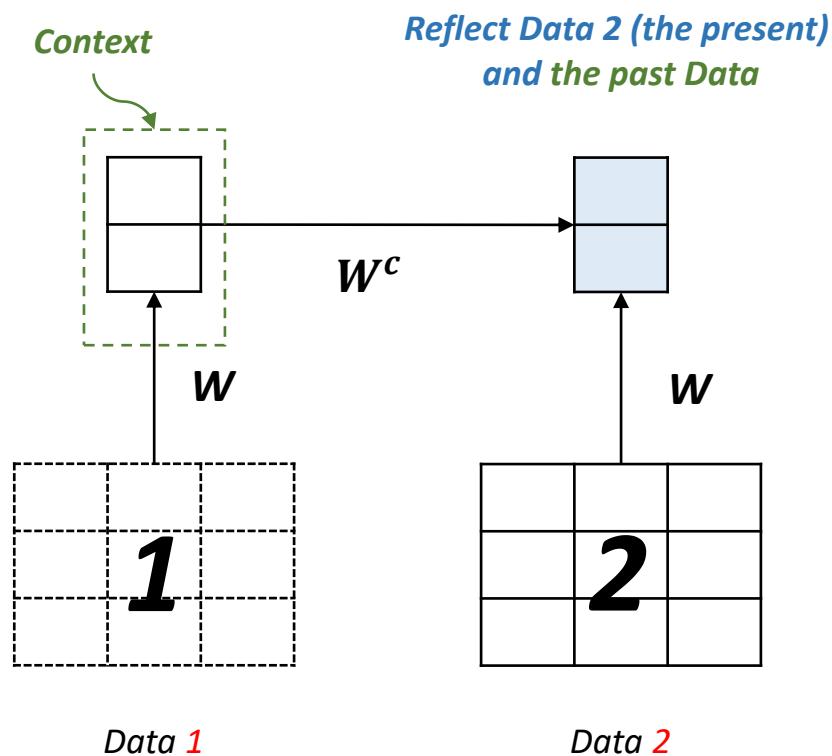


$V_s \rightarrow V's \rightarrow V'$



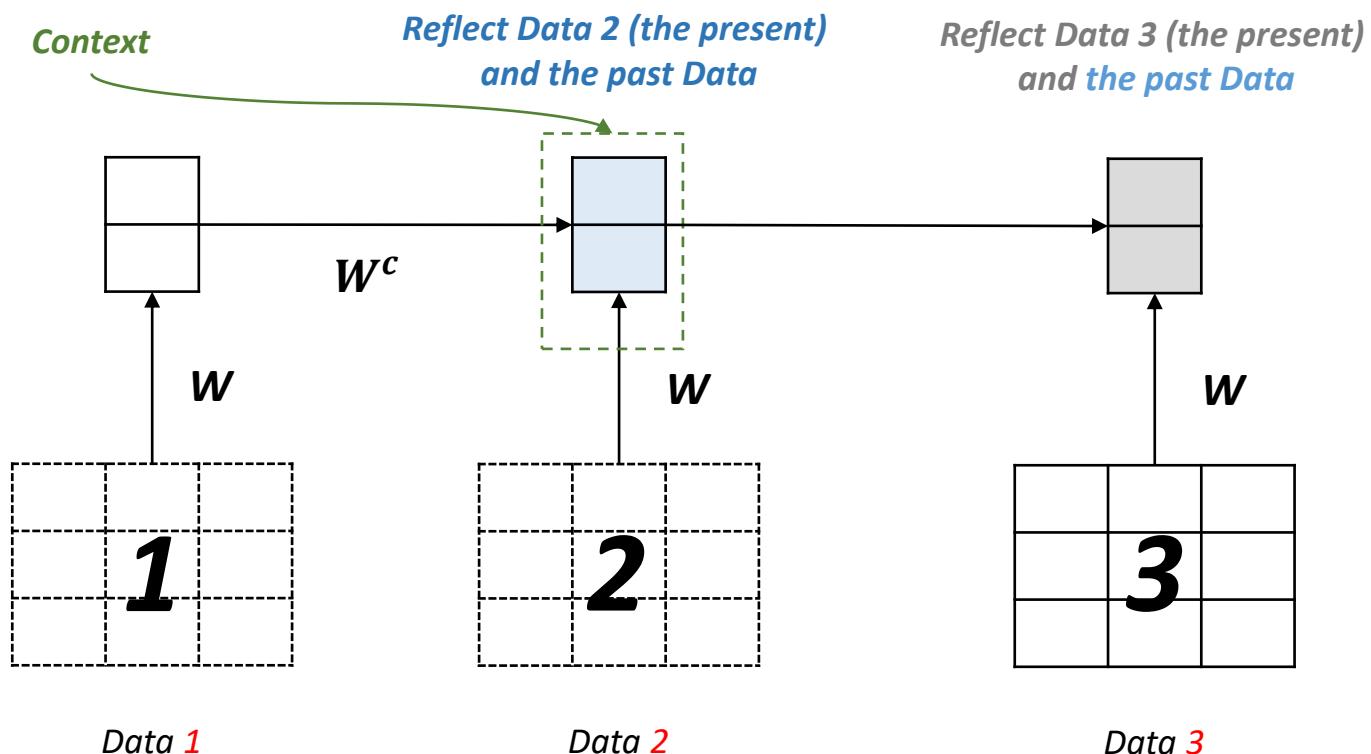
Data 2

$V_s \rightarrow V'_s \rightarrow V'$



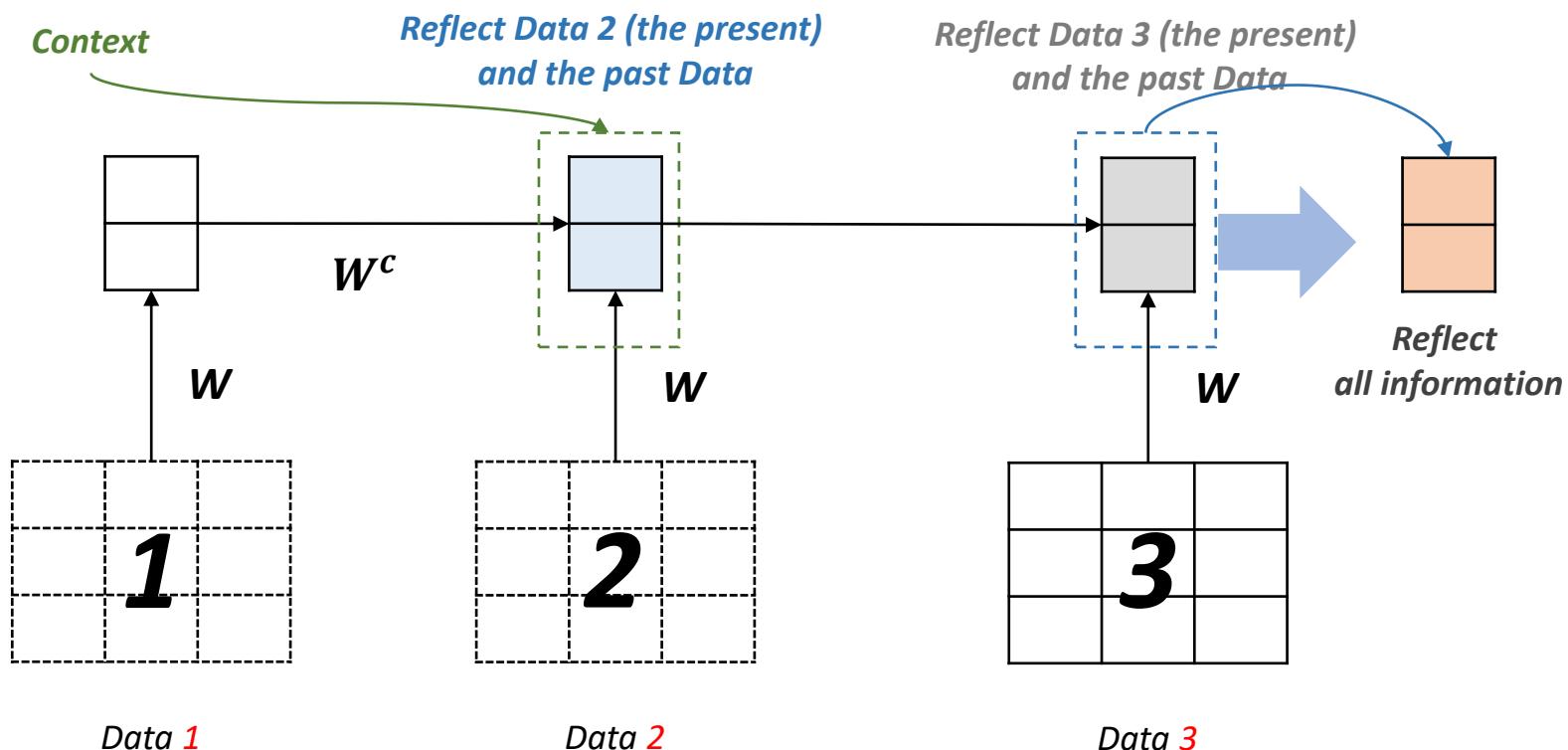
Data Transformation for Deep Learning NLP

$V_s \rightarrow V'_s \rightarrow V'$

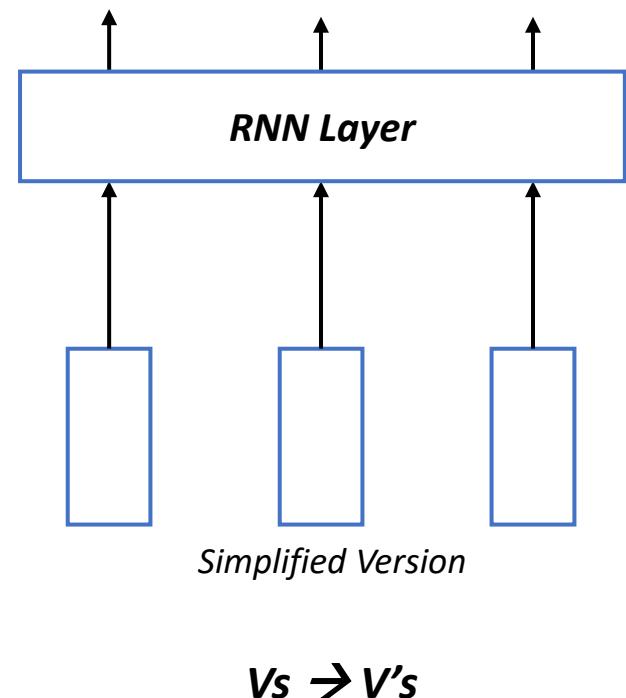
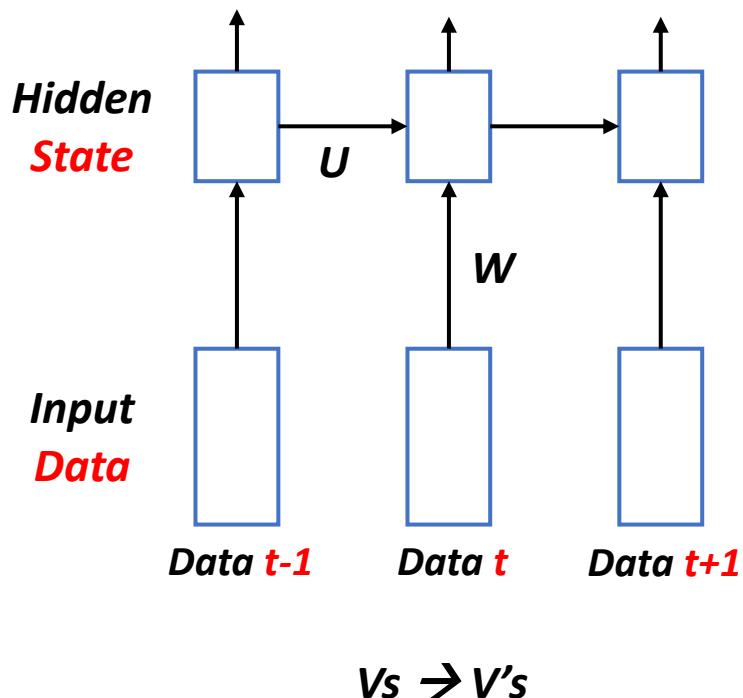


Data Transformation for Deep Learning NLP

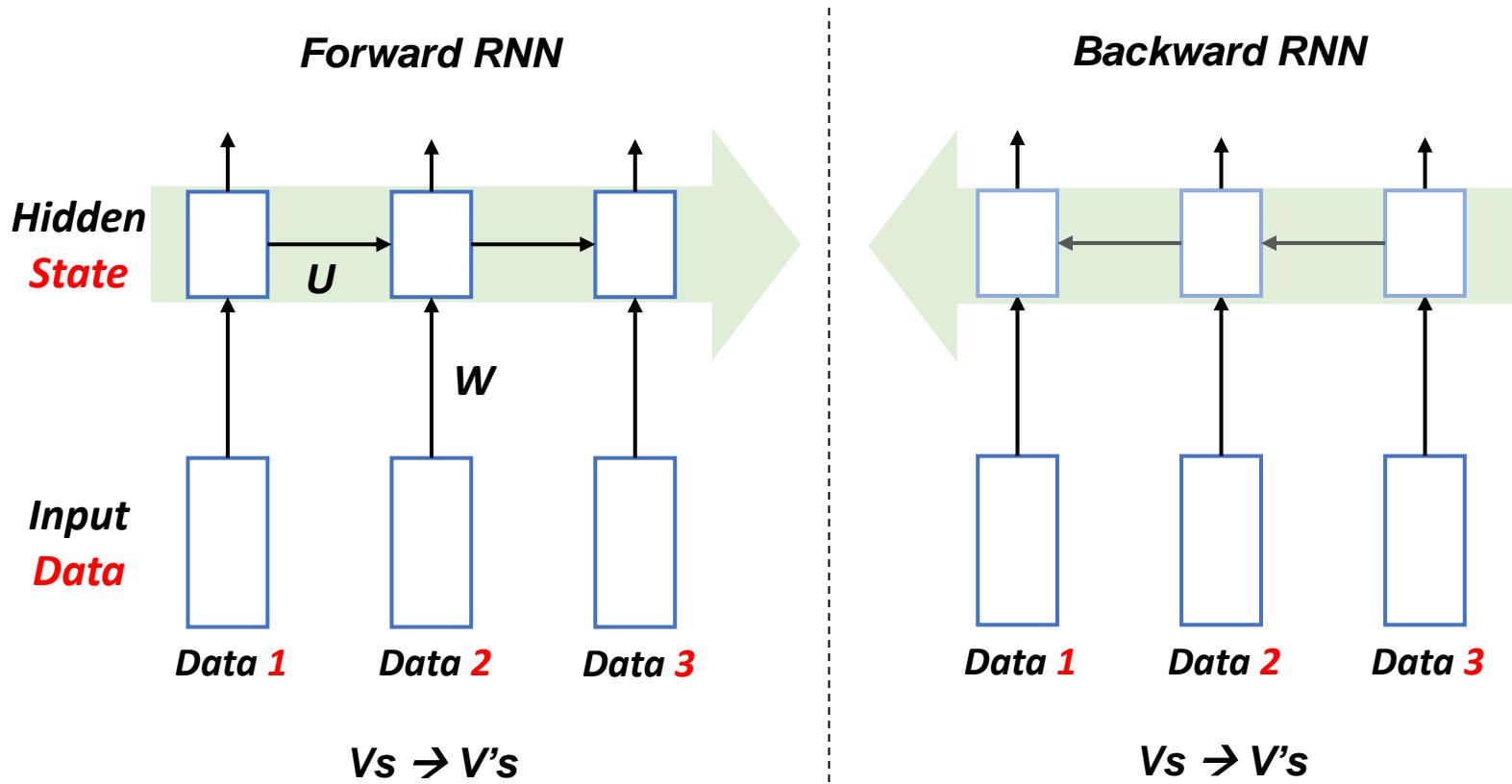
$V_s \rightarrow V's \rightarrow V'$



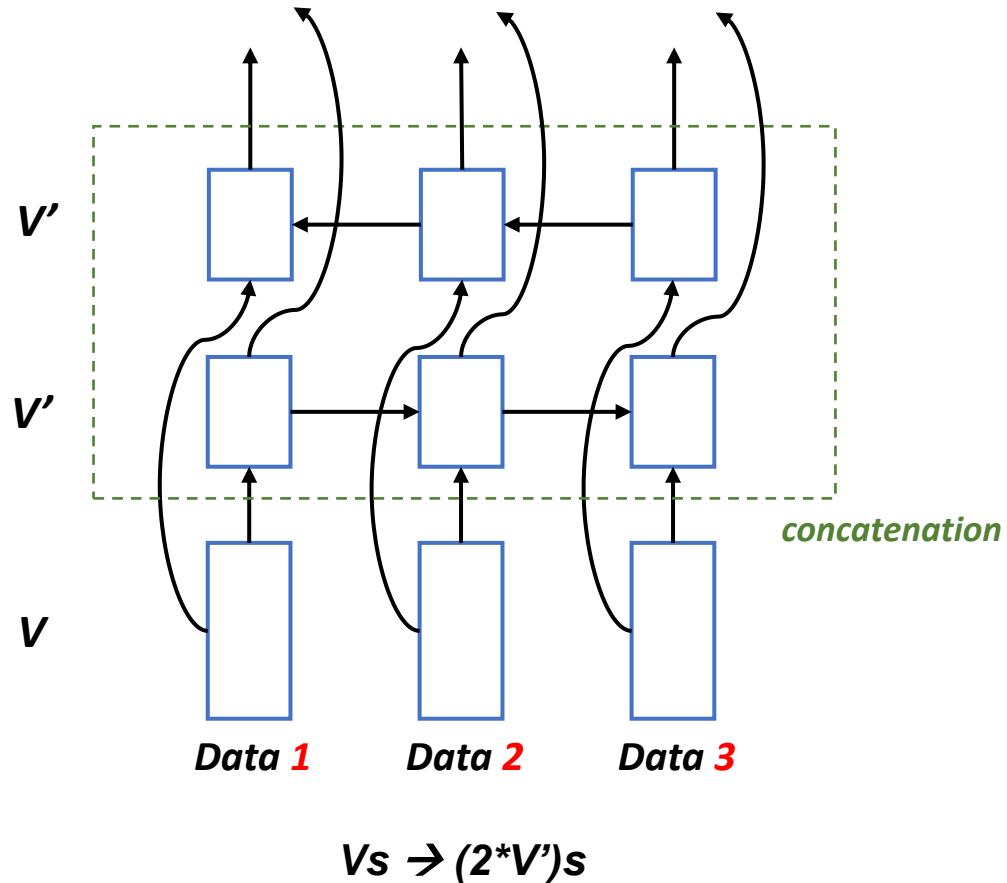
Graphical Notation



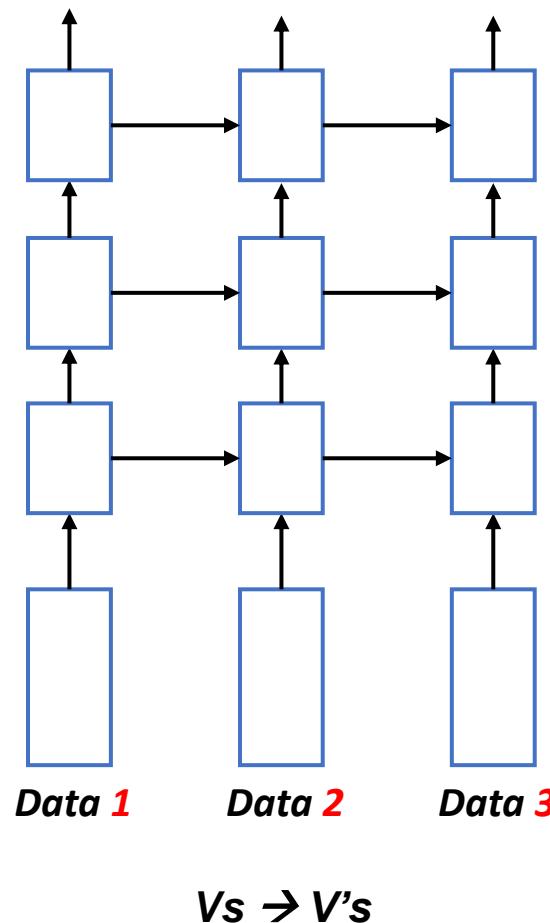
Forward/Backward RNN



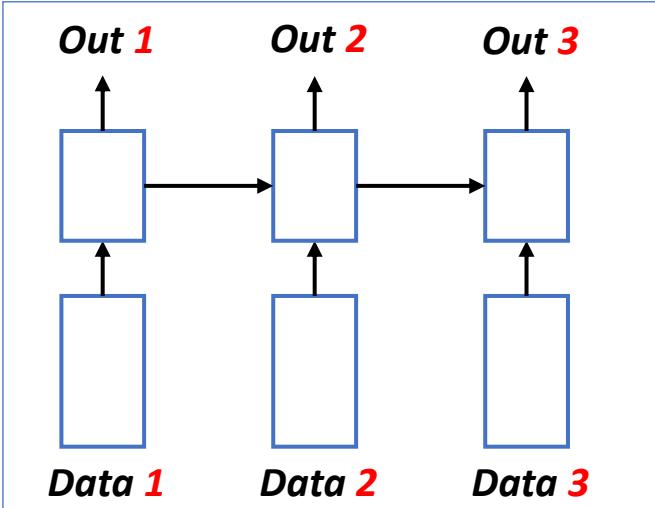
Bidirectional RNN



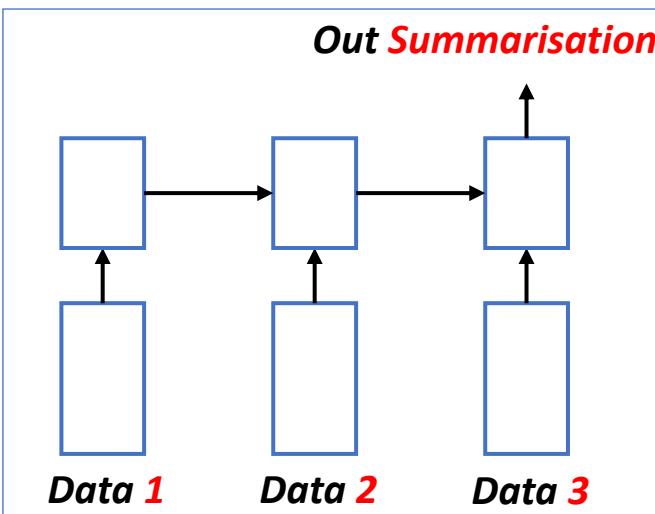
Stacking RNN



RNN: Input and Output

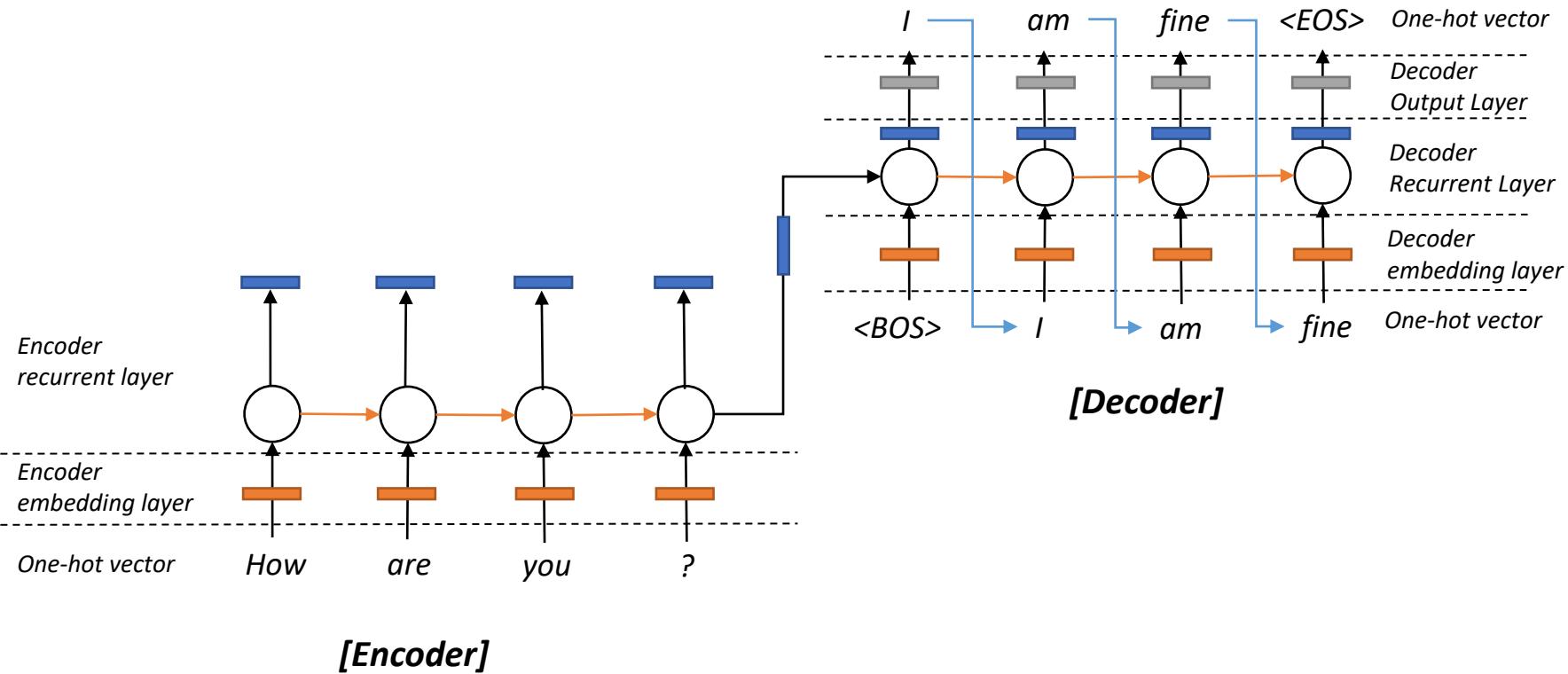


- ✓ $Vs \rightarrow V's$
- ✓ $\text{Len}(Vs) \rightarrow \text{Len}(V's)$



- ✓ $Vs \rightarrow 1$

Seq2Seq Encoding and Decoding- Dialog System



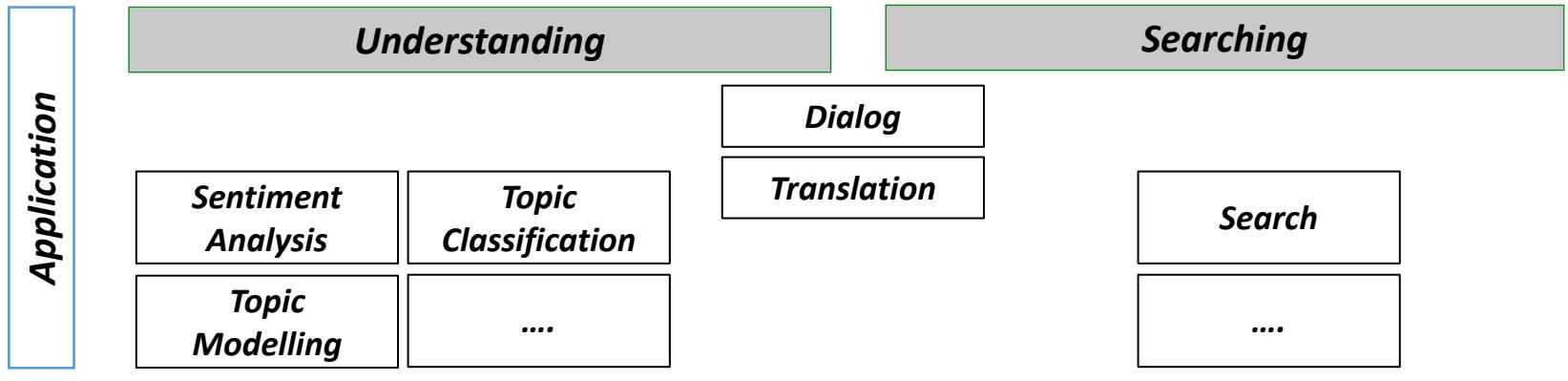
0 LECTURE PLAN

Lecture 4: Word Classification and Machine Learning 2

1. Machine Learning and NLP: Finish
 2. Seq2Seq Learning
 3. Seq2Seq Deep Learning
 1. RNN (Recurrent Neural Network)
 2. LSTM (Long Short-Term Memory)
 3. GRU (Gated Recurrent Unit)
 4. Data Transformation for Deep Learning NLP
- 5. Next Week Preview**
- Natural Language Processing Stack

5 Next Week Preview

The purpose of Natural Language Processing: Overview



NLP Stack	Entity Extraction	When Sebastian Thrun ...	When Sebastian Thrun PERSON started at Google ORG in 2007 DATE
	Parsing	Claudia sat on a stool	<pre> graph TD S --- NP1[NP] S --- VP NP1 --- N1[Claudia] NP1 --- V1[sat] VP --- PP VP --- NP2[NP] PP --- P1[on] PP --- AT1[a] NP2 --- N2[stool] </pre>
	PoS Tagging	She sells seashells	[she/PRP] [sells/VBZ] [seashells/NNS]
	Stemming	Drinking, Drank, Drunk	Drink
	Tokenisation	How is the weather today	[How] [is] [the] [weather] [today]

/ Reference

Reference for this lecture

- Deng, L., & Liu, Y. (Eds.). (2018). Deep Learning in Natural Language Processing. Springer.
- Rao, D., & McMahan, B. (2019). Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. " O'Reilly Media, Inc.".
- Manning, C. D., Manning, C. D., & Schütze, H. (1999). Foundations of statistical natural language processing. MIT press.
- Blunsom, P 2017, Deep Natural Language Processing, lecture notes, Oxford University
- Manning, C 2017, Natural Language Processing with Deep Learning, lecture notes, Stanford University
- Sordoni, A., Bengio, Y., Vahabi, H., Lioma, C., Grue Simonsen, J., & Nie, J. Y. (2015, October). A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (pp. 553-562). ACM.

Figure Reference

- <https://towardsdatascience.com/types-of-optimization-algorithms-used-in-neural-networks-and-ways-to-optimize-gradient-95ae5d39529f>
- <https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>

COMP5046

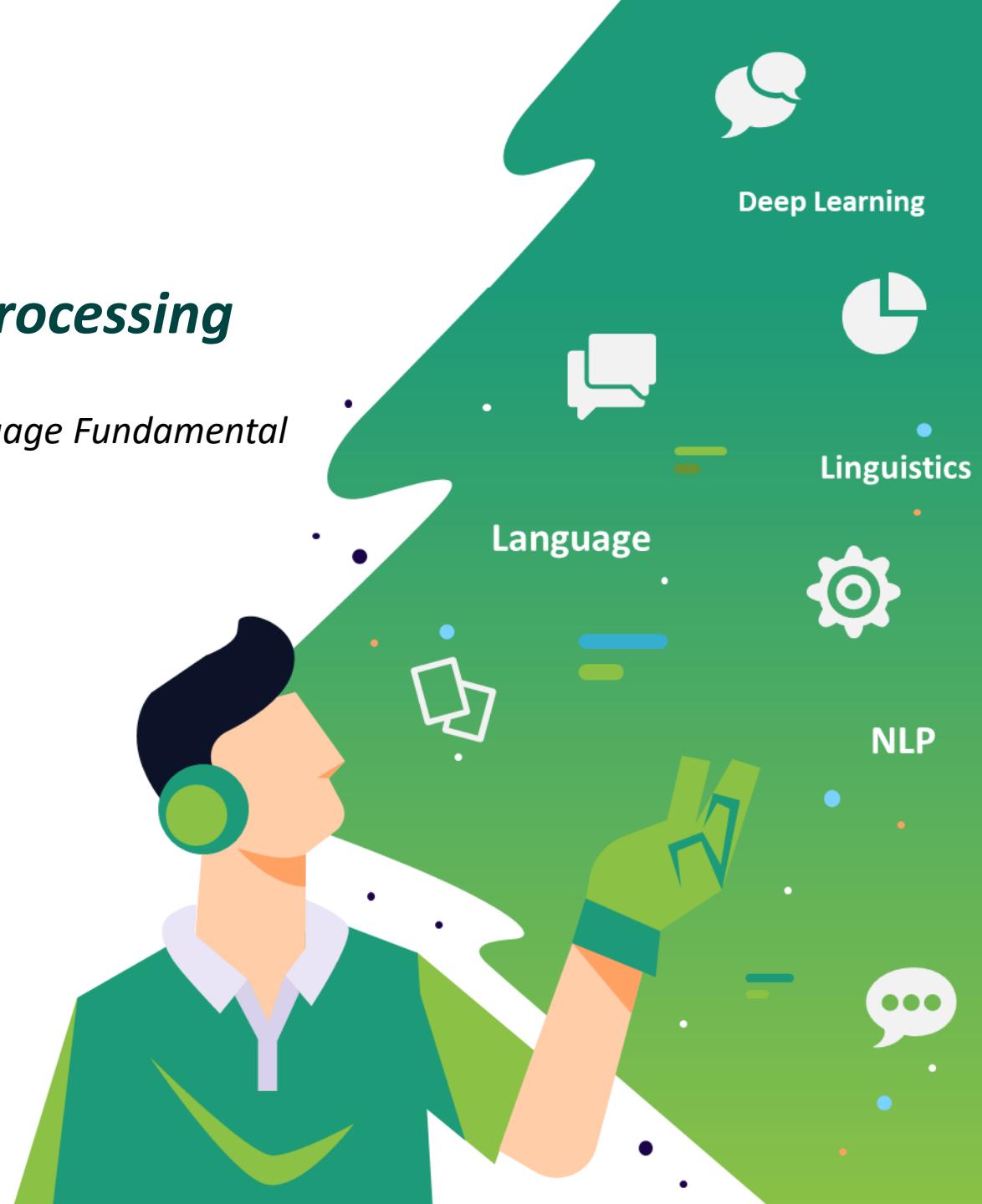
Natural Language Processing

Lecture 5: Assignment1 and Language Fundamental

Dr. Caren Han

Semester 1, 2022

School of Computer Science,
University of Sydney

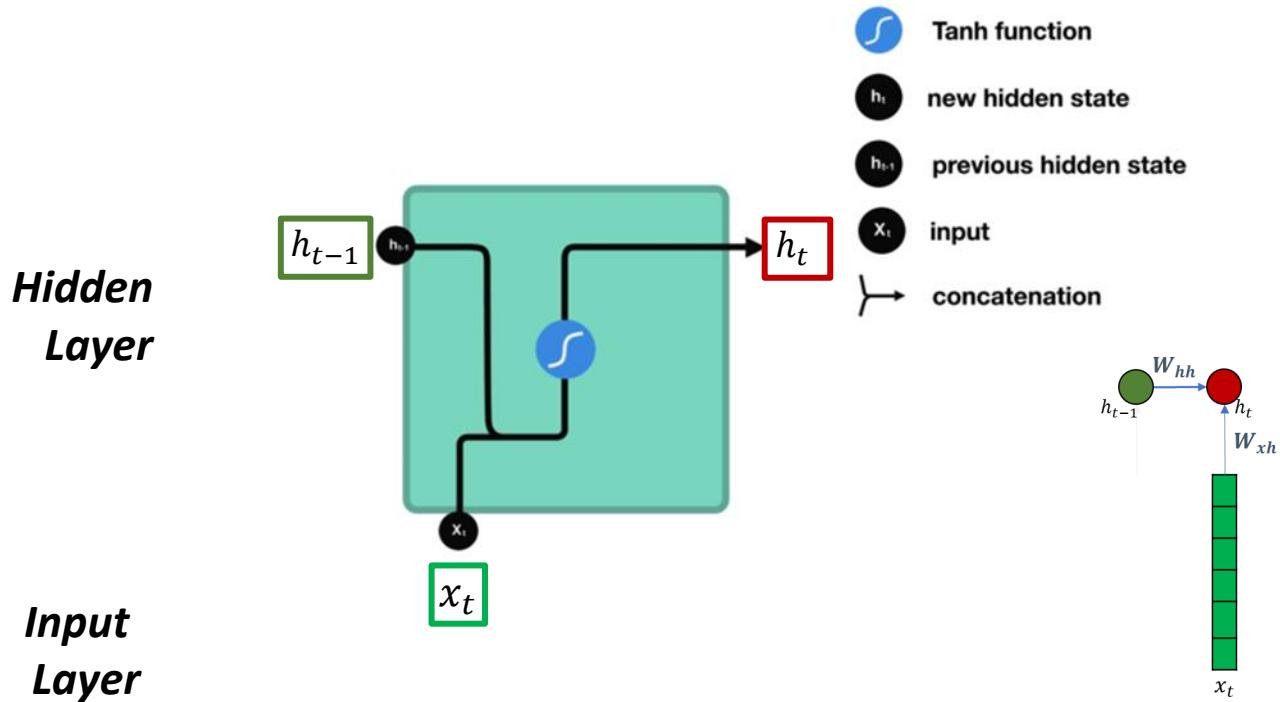


0 LECTURE PLAN

Lecture 5: Assignment1 and Language Fundamental

1. RNN/LSTM, Dealing Context Review
2. Assignment 1 Discussion
3. Sentiment Analysis
 1. Sentiment Analysis
 2. Sentiment Analysis: Examples
 3. Sentiment Analysis: Lexicons
4. Language Fundamental
 - Phonology, Morphology, Syntax, Semantics, Pragmatics
5. Text Preprocessing
 1. Tokenization
 2. Cleaning and Normalisation
 3. Stemming and Lemmatisation
 4. Stopword
 5. Regular Expression

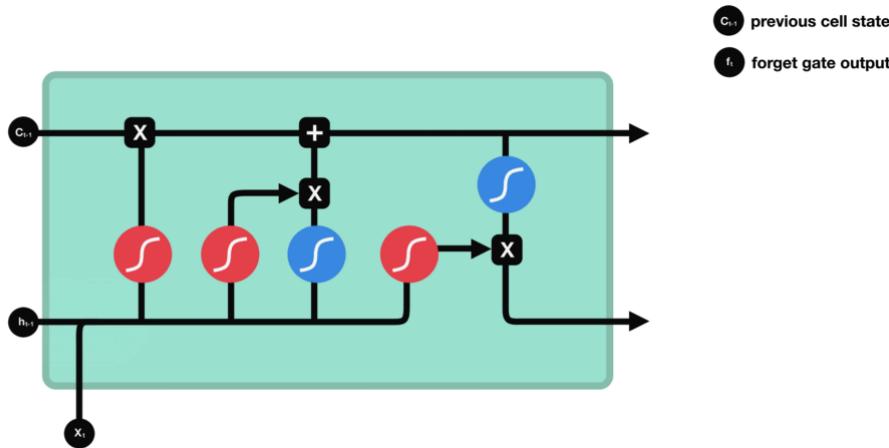
Neural Network + Memory = Recurrent Neural Network



$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$

New hidden state A function with parameters W Previous state input
 h_t \tanh W_{hh} h_{t-1} W_{xh} x_t b_h

LSTM (Long Short-Term Memory) – Forget Gate

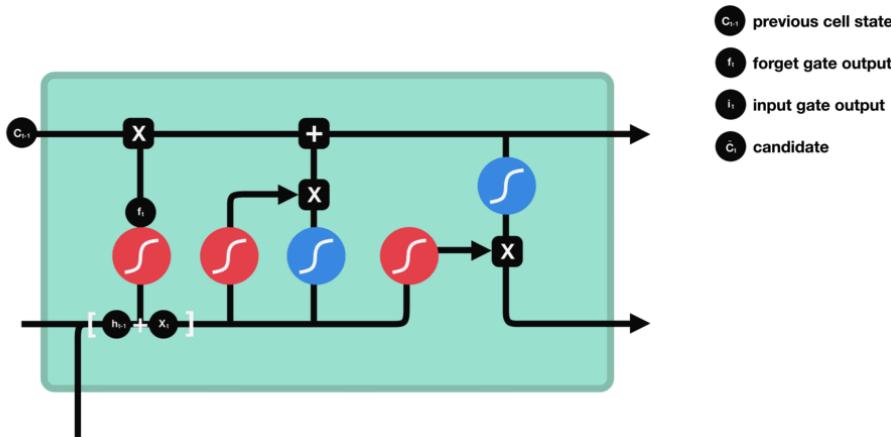


$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

Decides what information should be thrown away or kept

Information from the **previous hidden state** and information from the **current input** is passed through the **sigmoid function**. Values come out between 0 and 1. The closer to 0 means to forget, and the closer to 1 means to keep.

LSTM (Long Short-Term Memory) – Input Gate

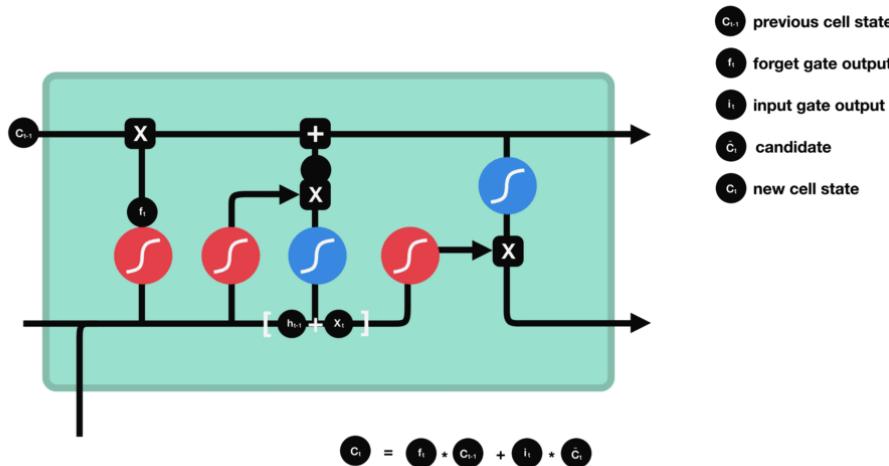


$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C)$$

1. Pass the previous hidden state and current input into a sigmoid function
2. Pass the hidden state and current input into the tanh function to squish values between -1 and 1 to help regulate the network
3. Multiply the tanh output with the sigmoid output
**sigmoid output will decide which information is important to keep from the tanh output*

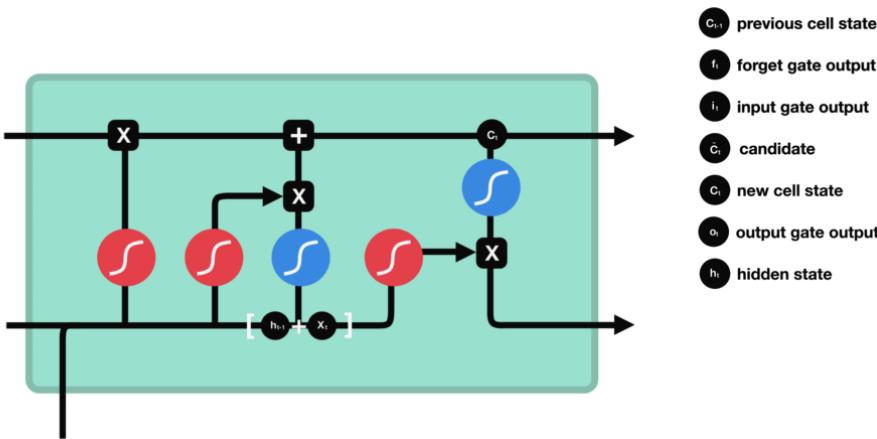
LSTM (Long Short-Term Memory) – Cell States



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- *the cell state gets pointwise multiplied by the forget vector*
- *take the output from the input gate and do a pointwise addition which updates the cell state to new values that the neural network finds relevant*
- *That gives us our new cell state*

LSTM (Long Short-Term Memory) – Output Gate



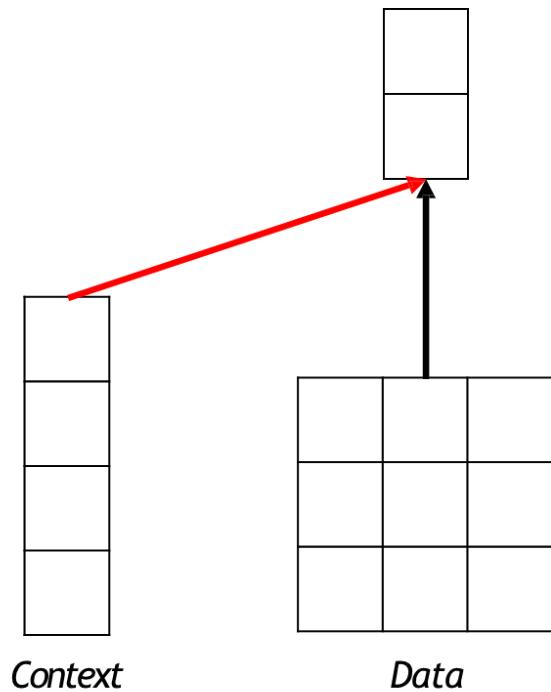
$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t)$$

decides what the next hidden state should be.

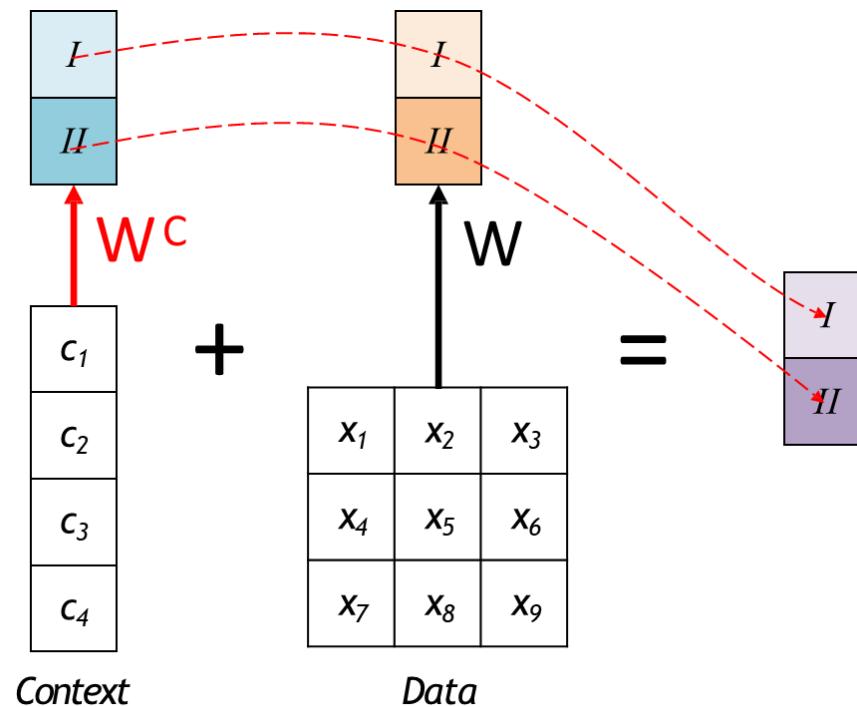
- pass the previous hidden state and the current input into a sigmoid function
- pass the newly modified cell state to the tanh function
- multiply the tanh output with the sigmoid output to decide what information the hidden state should carry

1 Dealing Context: Review

V to V' – Projection with Context (1)

1 Dealing Context: Review

V to V' – Projection with Context (2)



1 Dealing Context: Review

V to V' with Context - Linear Algebra

[1 x 9] matrix

$$\begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 \end{bmatrix}$$

[9x2] matrix

$$\begin{array}{|c|c|} \hline w_{1,1} & w_{2,1} \\ \hline w_{1,2} & w_{2,2} \\ \hline w_{1,3} & w_{2,3} \\ \hline w_{1,4} & w_{2,4} \\ \hline w_{1,5} & w_{2,5} \\ \hline w_{1,6} & w_{2,6} \\ \hline w_{1,7} & w_{2,7} \\ \hline w_{1,8} & w_{2,8} \\ \hline w_{1,9} & w_{2,9} \\ \hline \end{array}$$

[1x2] matrix

$$= \left(\sum_i^9 x_i * w_{1,i}, \sum_i^9 x_i * w_{2,i} \right)$$

I
II

[1 x 4] matrix

$$\begin{bmatrix} c_1 & c_2 & c_3 & c_4 \end{bmatrix}$$

X [1x2] matrix

$$\begin{array}{|c|c|} \hline w_{1,1}^c & w_{2,1}^c \\ \hline w_{1,2}^c & w_{2,2}^c \\ \hline w_{1,3}^c & w_{2,3}^c \\ \hline w_{1,4}^c & w_{2,4}^c \\ \hline \end{array}$$

$$= \left(\sum_i^4 c_i * w_{1,i}^c, \sum_i^4 c_i * w_{2,i}^c \right)$$

I
II

1 Dealing Context: Review

V to V' with Context - Linear Algebra (Simplified)

[1 x (9+4)] matrix

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	c_1	c_2	c_3	c_4
-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

X

[(9+4) x 2] matrix

$w_{1,1}$	$w_{2,1}$
$w_{1,2}$	$w_{2,2}$
$w_{1,3}$	$w_{2,3}$
$w_{1,4}$	$w_{2,4}$
$w_{1,5}$	$w_{2,5}$
$w_{1,6}$	$w_{2,6}$
$w_{1,7}$	$w_{2,7}$
$w_{1,8}$	$w_{2,8}$
$w_{1,9}$	$w_{2,9}$
$w_{1,C,1}$	$w_{2,C,1}$
$w_{1,C,2}$	$w_{2,C,2}$
$w_{1,C,3}$	$w_{2,C,3}$
$w_{1,C,4}$	$w_{2,C,4}$

[1x2] matrix

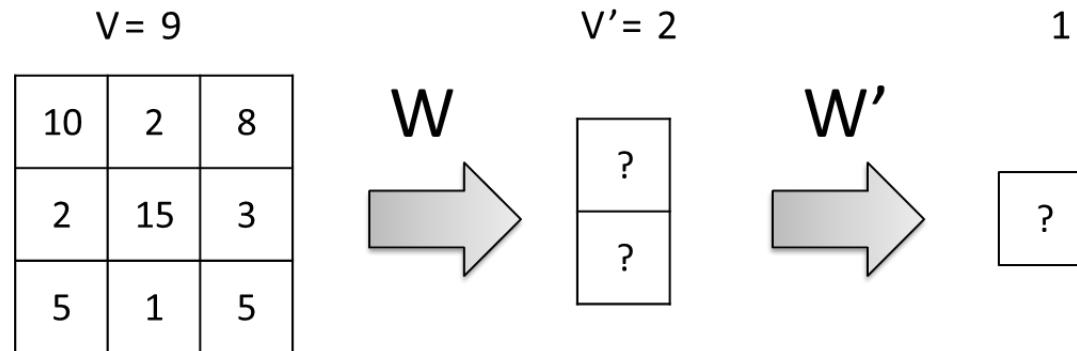
$$= \begin{pmatrix} \sum_i^9 x_i * w_{1,i} & \sum_i^9 x_i * w_{2,i} \\ + \sum_i^4 c_i * w_{1,i}^C & + \sum_i^4 c_i * w_{2,i}^C \end{pmatrix}$$

I

II

1

Dealing Context: Review

 $V \rightarrow V' \rightarrow 1$ 

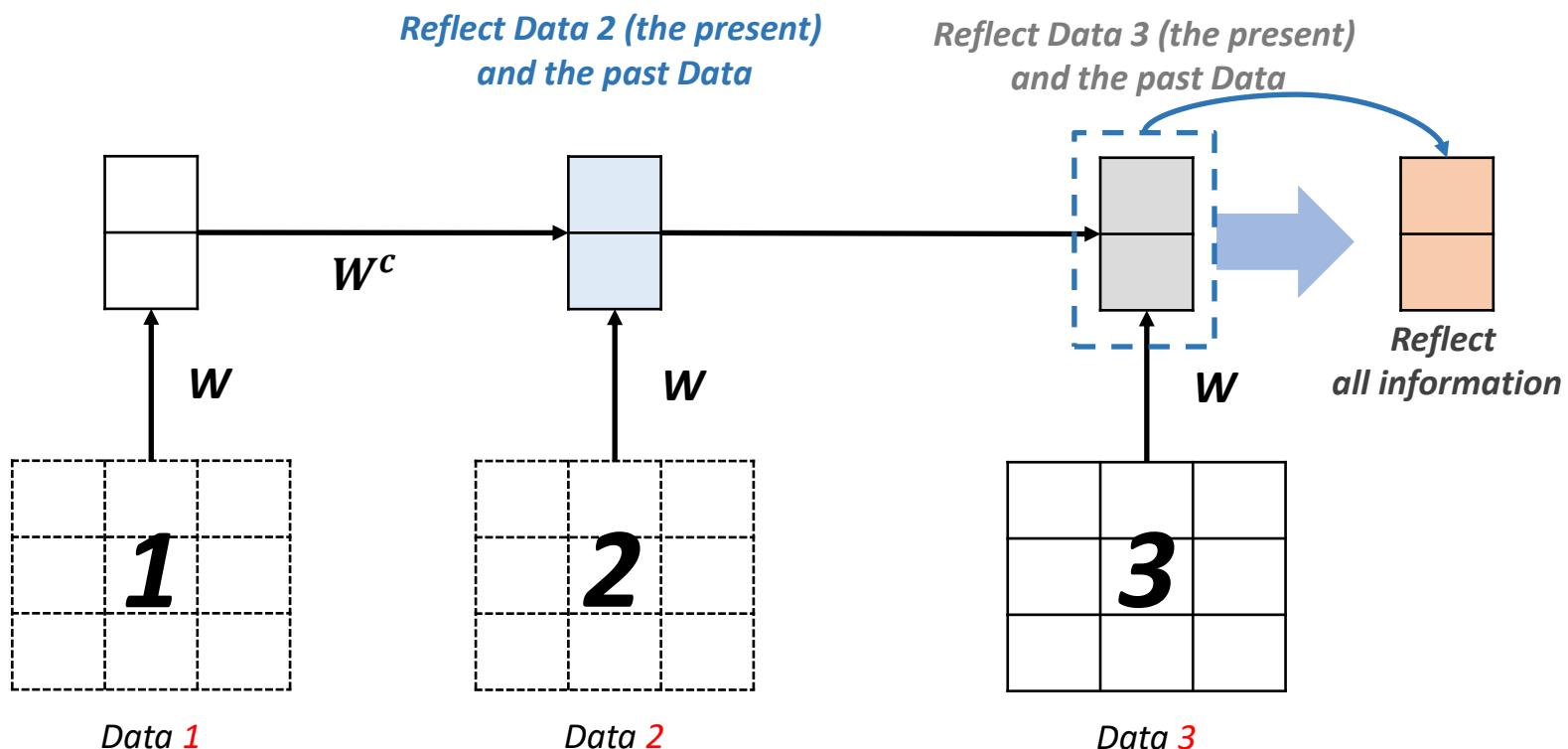
0 LECTURE PLAN

Lecture 5: Assignment1 and Language Fundamental

1. RNN/LSTM, Dealing Context Review
2. **Assignment 1 Discussion**
3. Sentiment Analysis
 1. Sentiment Analysis
 2. Sentiment Analysis: Examples
 3. Sentiment Analysis: Lexicons
4. Language Fundamental
 - Phonology, Morphology, Syntax, Semantics, Pragmatics
5. Text Preprocessing
 1. Tokenization
 2. Cleaning and Normalisation
 3. Stemming and Lemmatisation
 4. Stopword
 5. Regular Expression

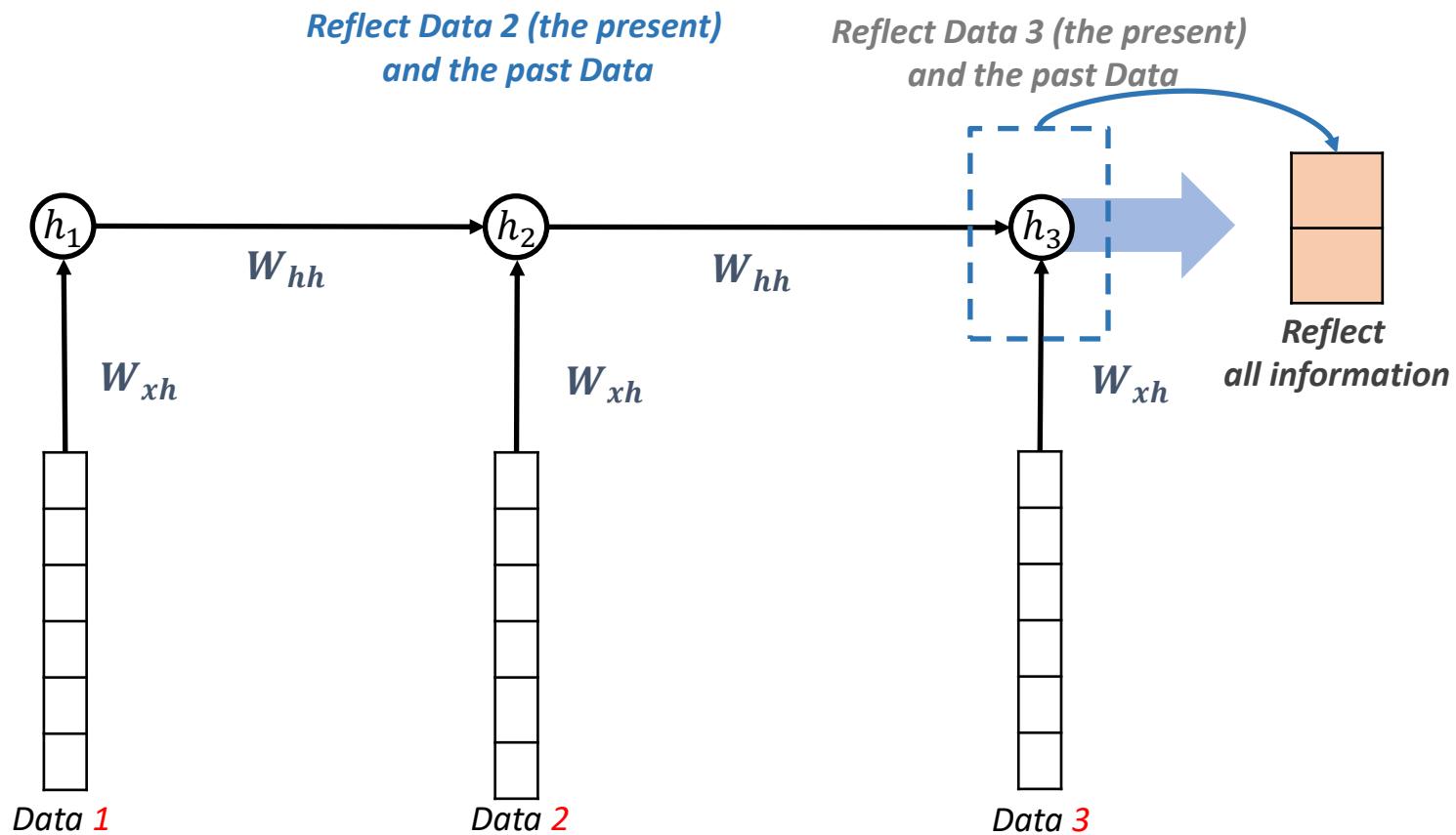
2 Assignment 1 Discussion

$V_s \rightarrow V's \rightarrow V'$



2 Assignment 1 Discussion

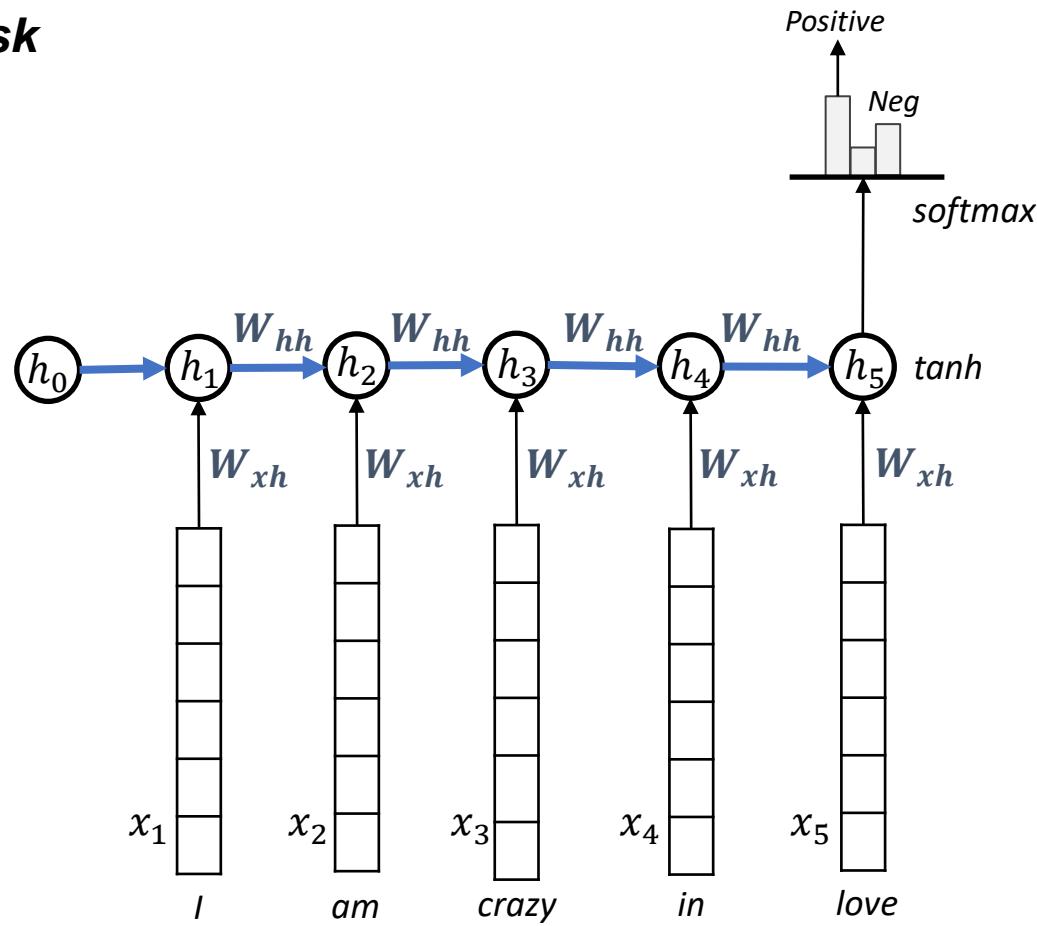
$V_s \rightarrow V's \rightarrow V'$



2 Assignment 1 Discussion

RNN

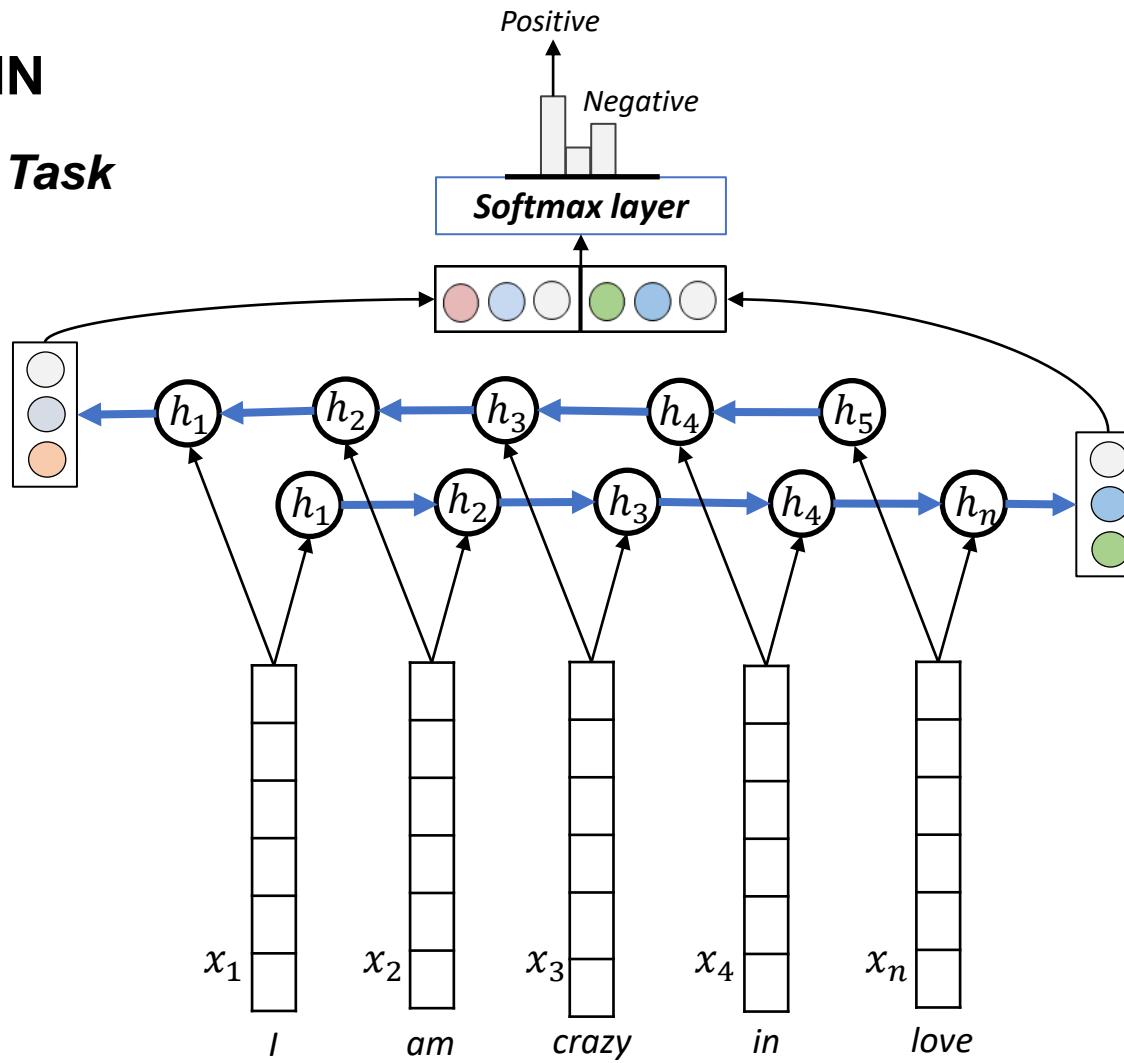
N to 1 Task



2 Assignment 1 Discussion

Bi-RNN

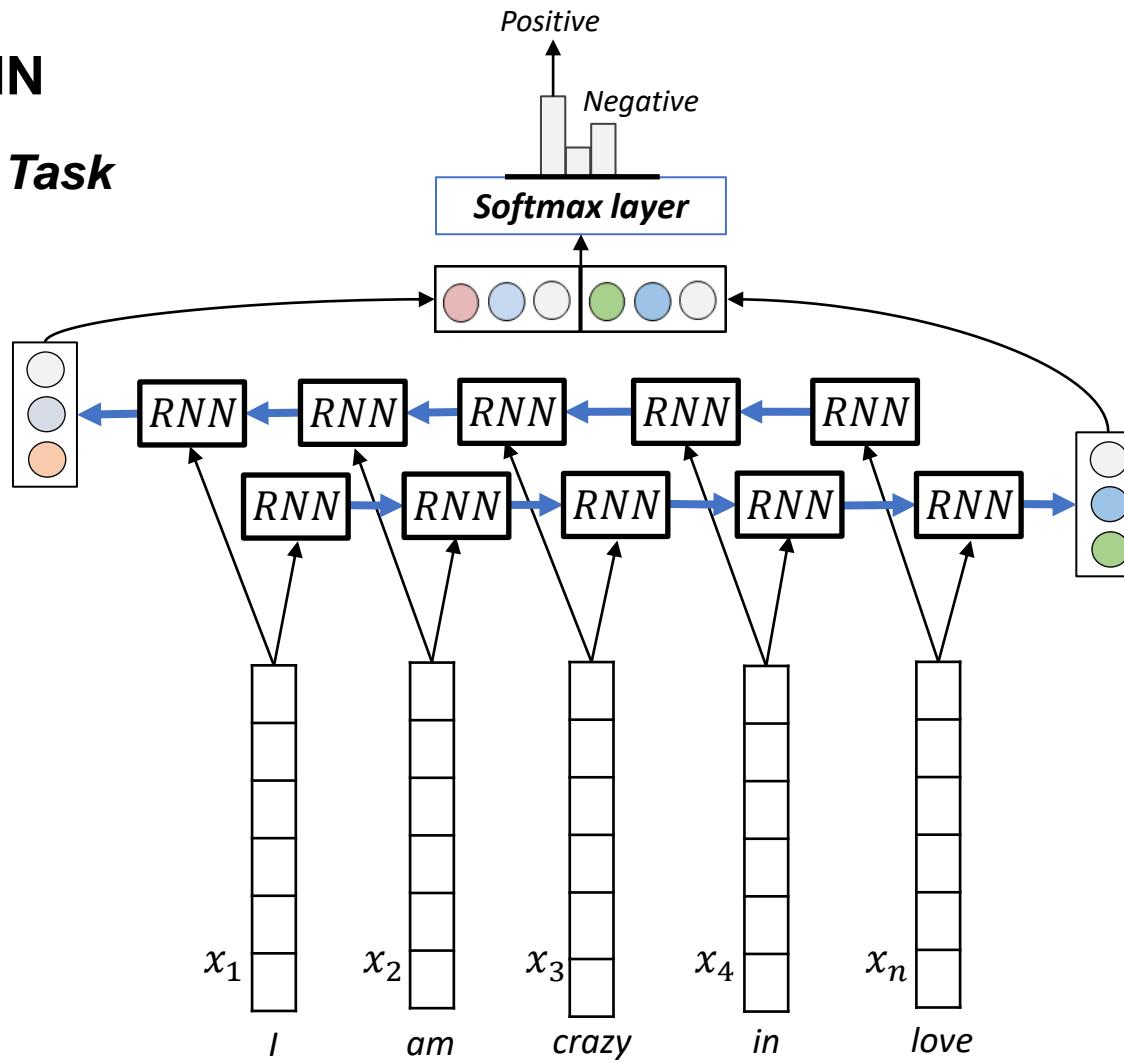
N to 1 Task



2 Assignment 1 Discussion

Bi-RNN

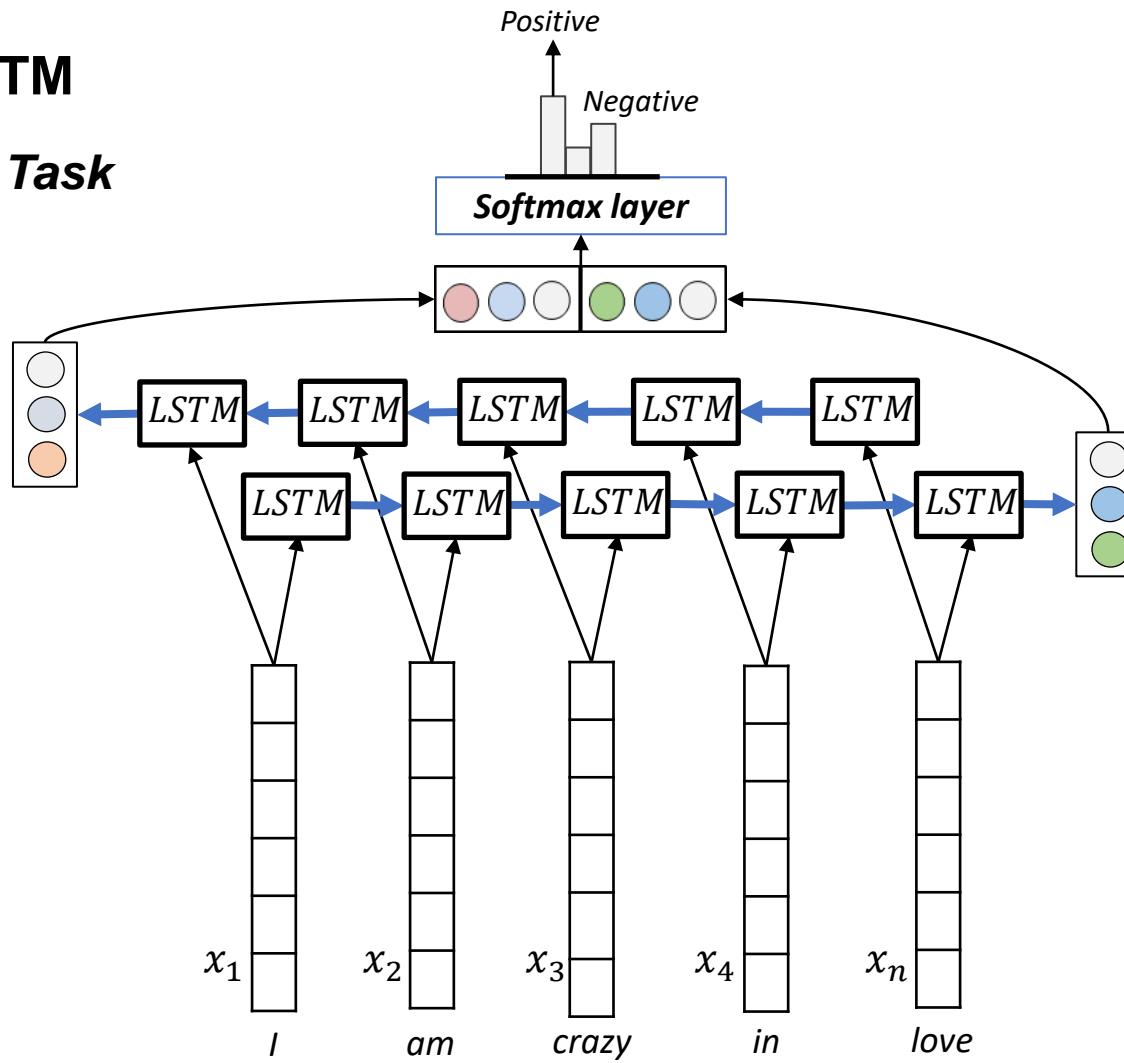
N to 1 Task



2 Assignment 1 Discussion

Bi-LSTM

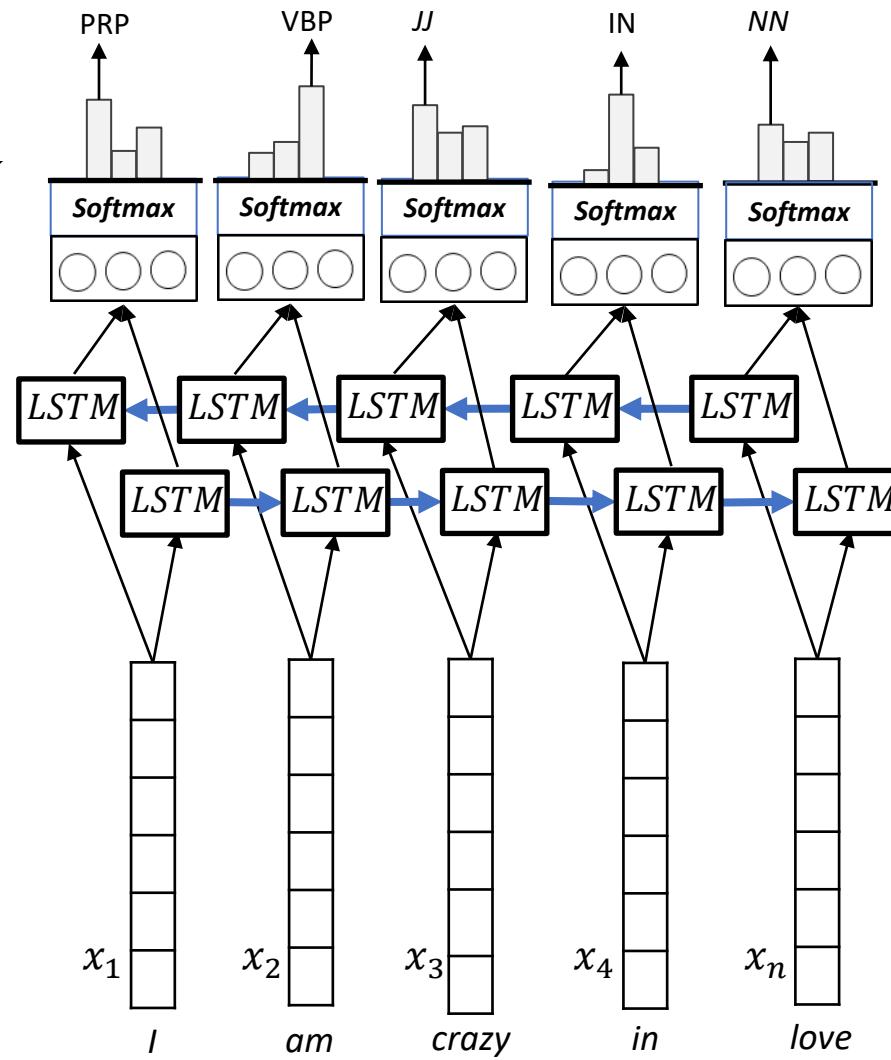
N to 1 Task



2 Assignment 1 Discussion

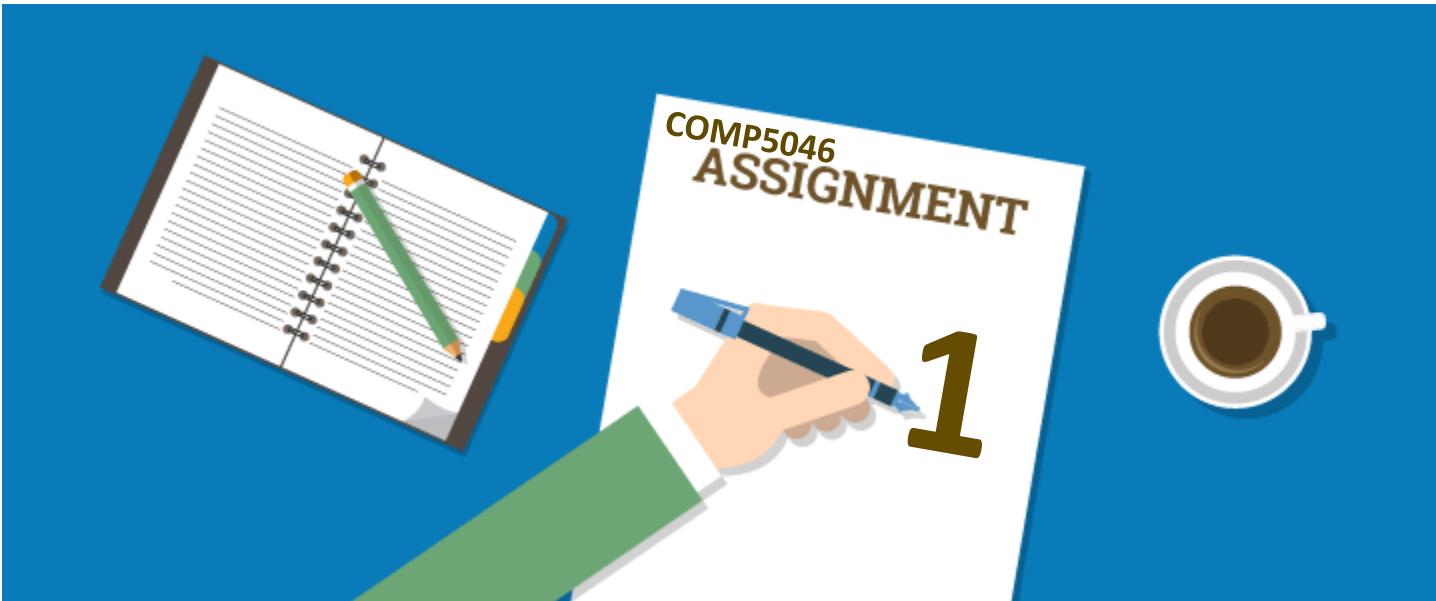
Bi-LSTM

N to N Task



2 Assignment 1 Discussion

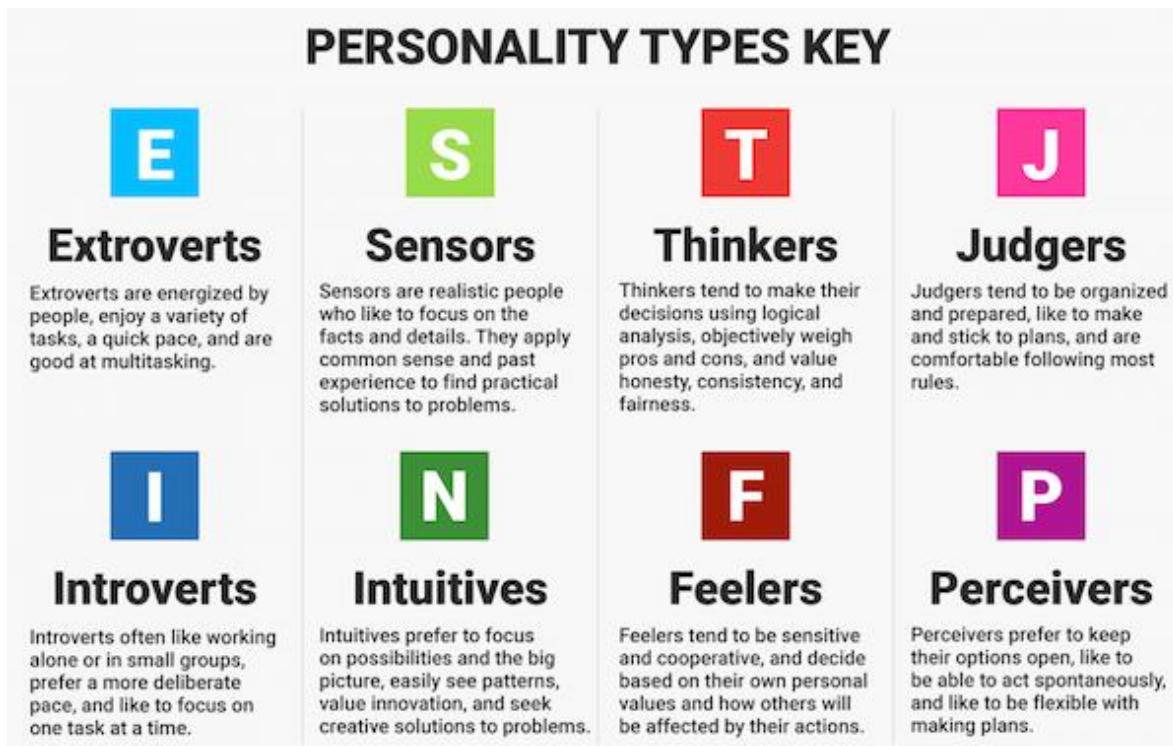
Let's discuss our Assignment 1



2 Assignment 1 Discussion

Understanding Personality:

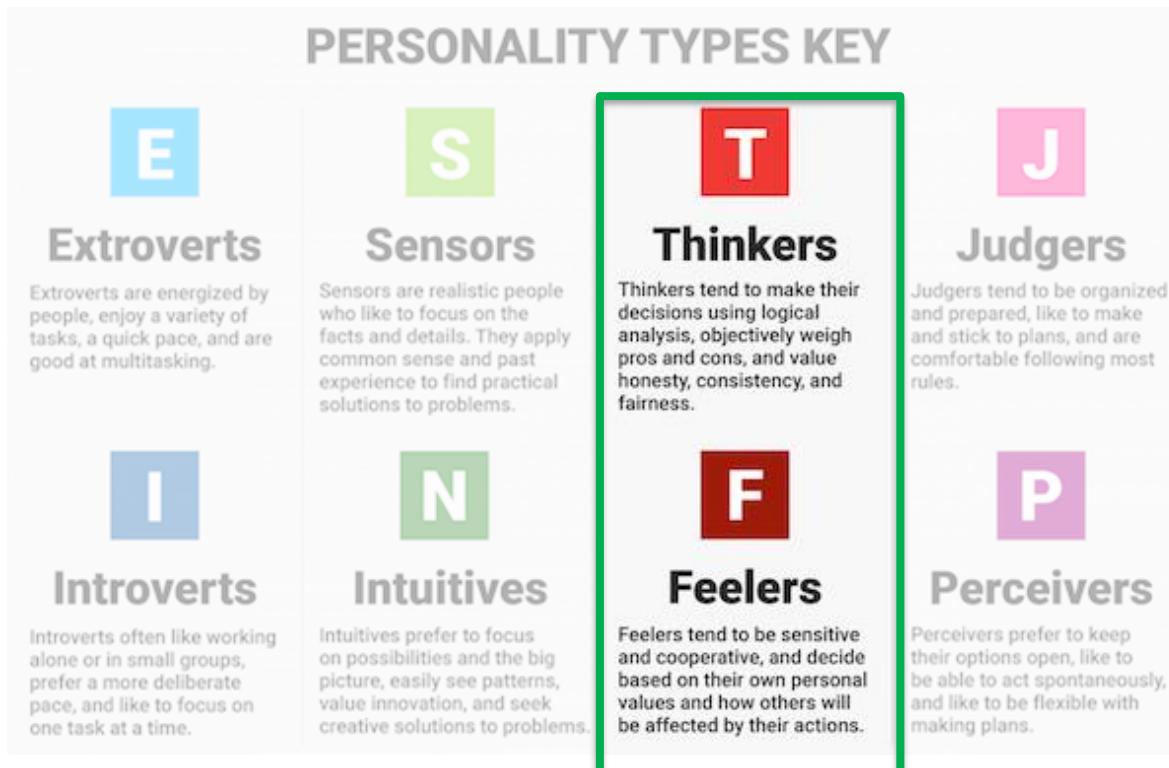
MYERS – BRIGGS TYPE INDICATOR (MBTI)



2 Assignment 1 Discussion

Understanding Personality:

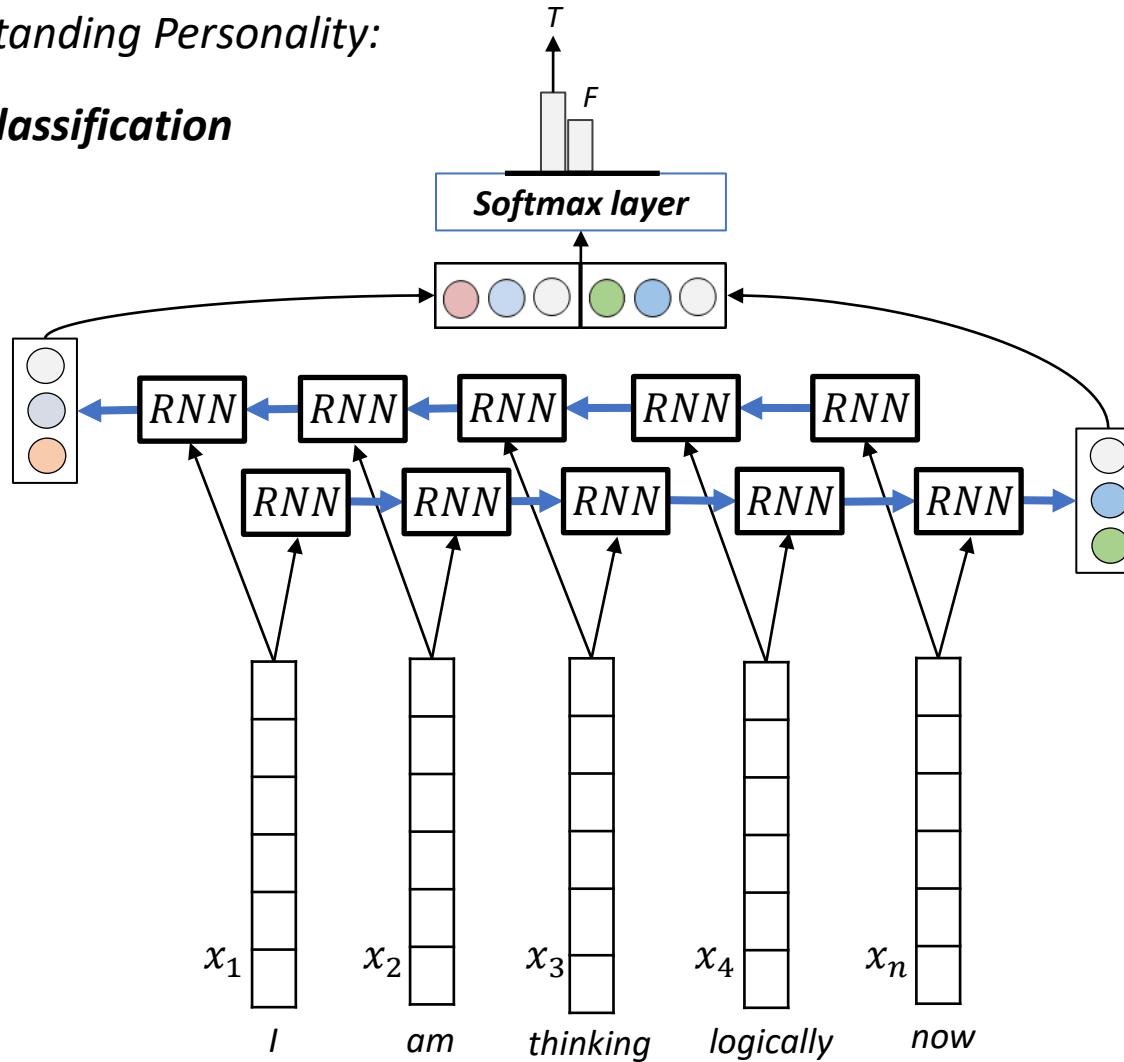
MYERS – BRIGGS TYPE INDICATOR (MBTI)



2 Assignment 1 Discussion

Understanding Personality:

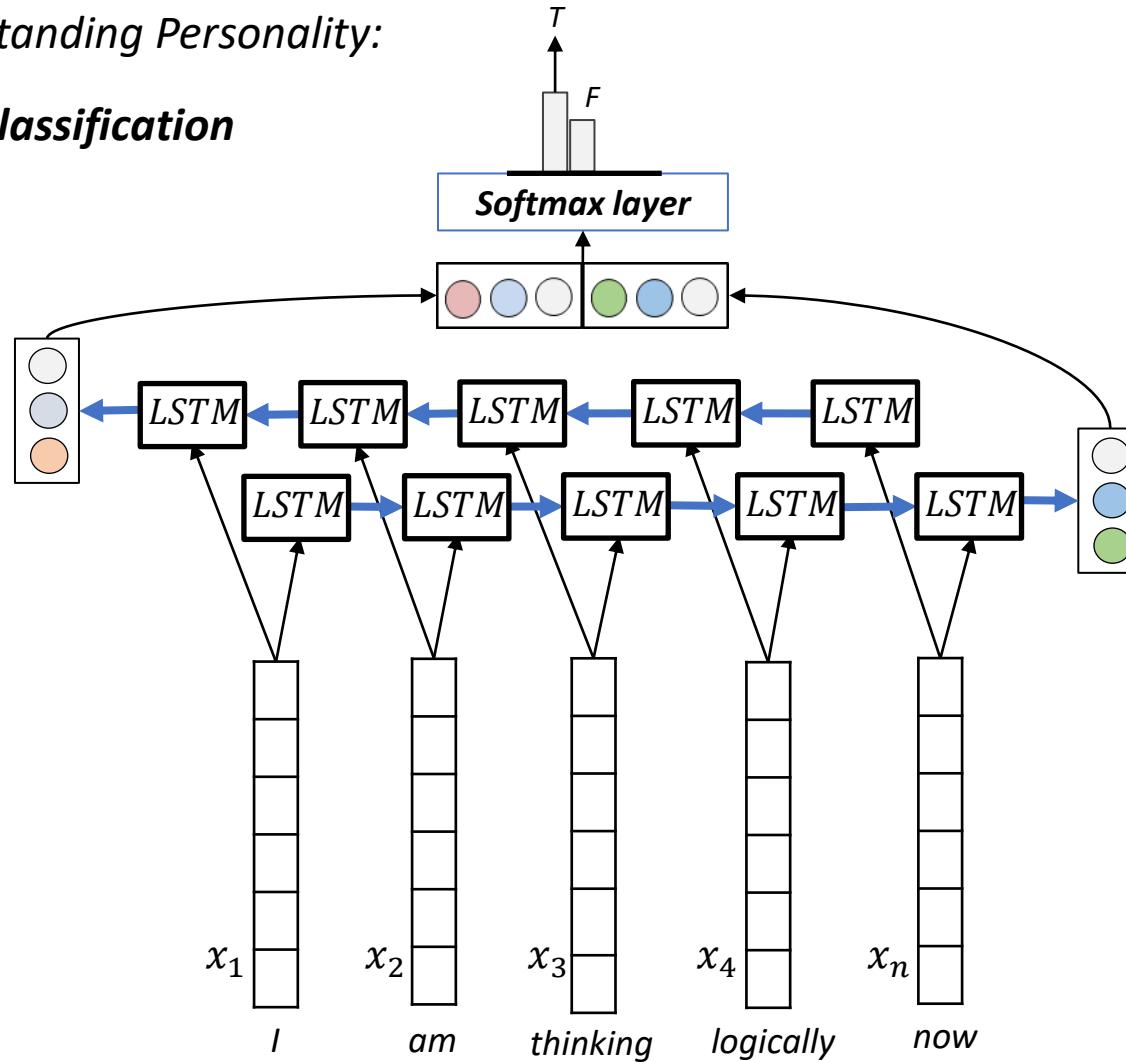
MBTI Classification



2 Assignment 1 Discussion

Understanding Personality:

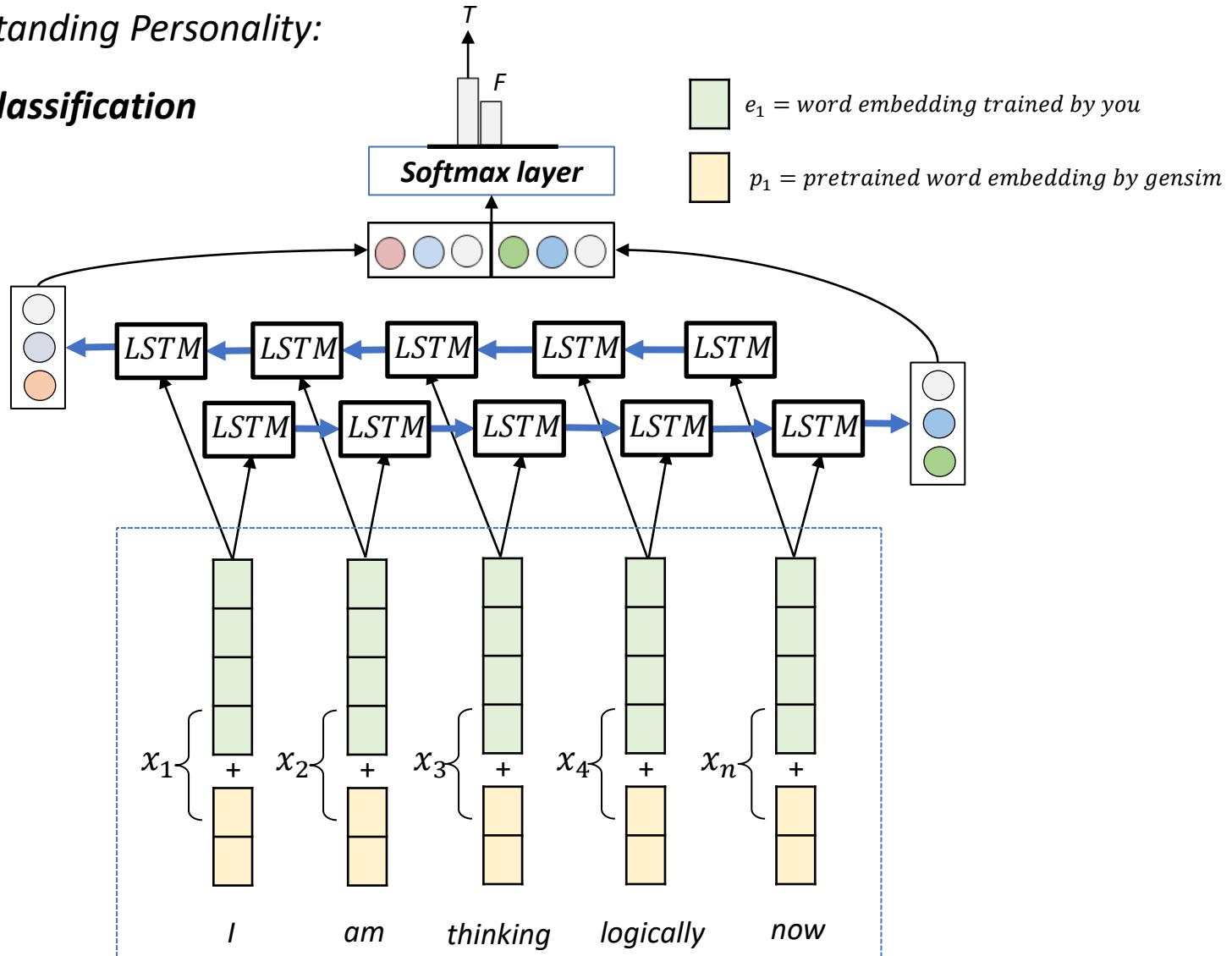
MBTI Classification



2 Assignment 1 Discussion

Understanding Personality:

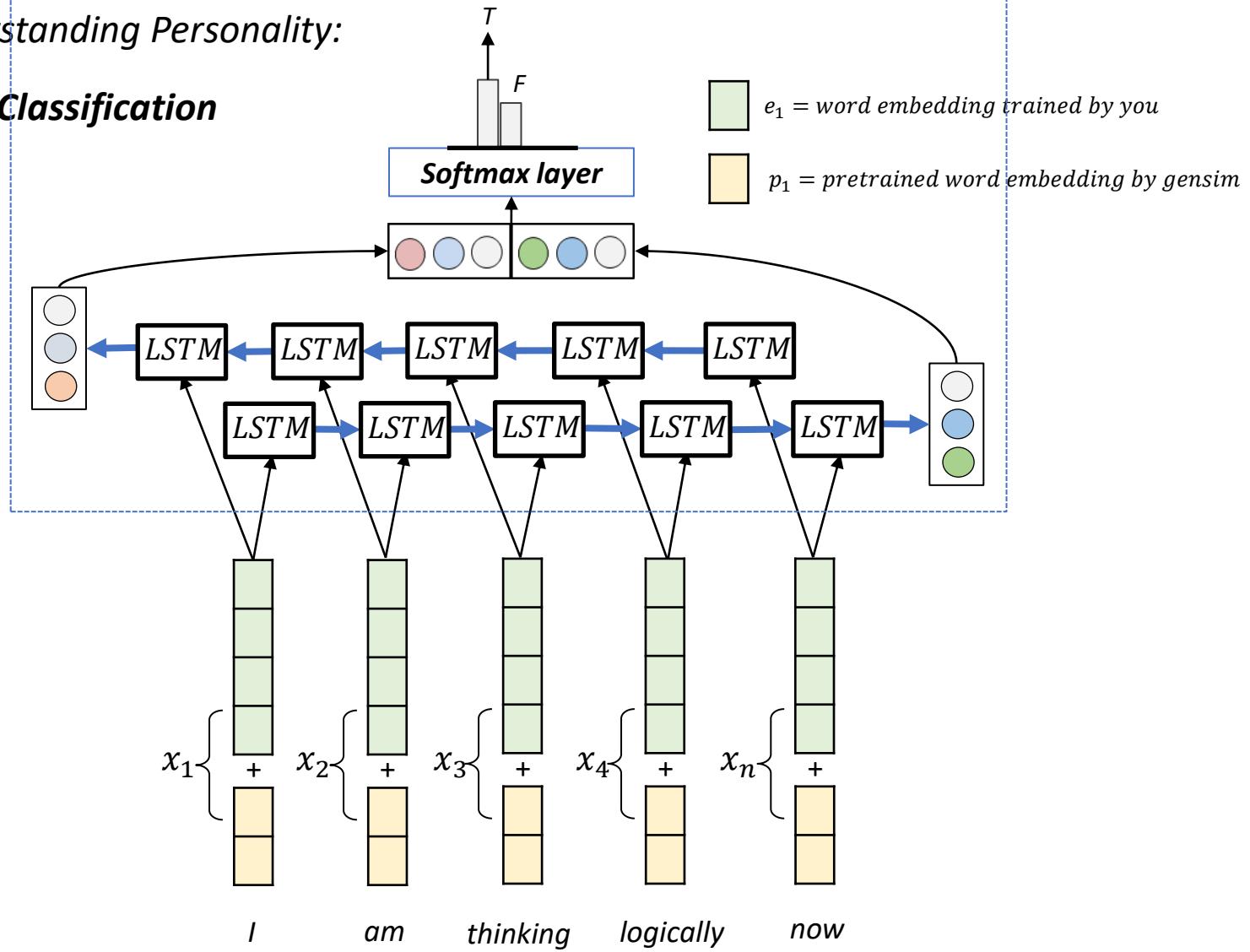
MBTI Classification



2 Assignment 1 Discussion

Understanding Personality:

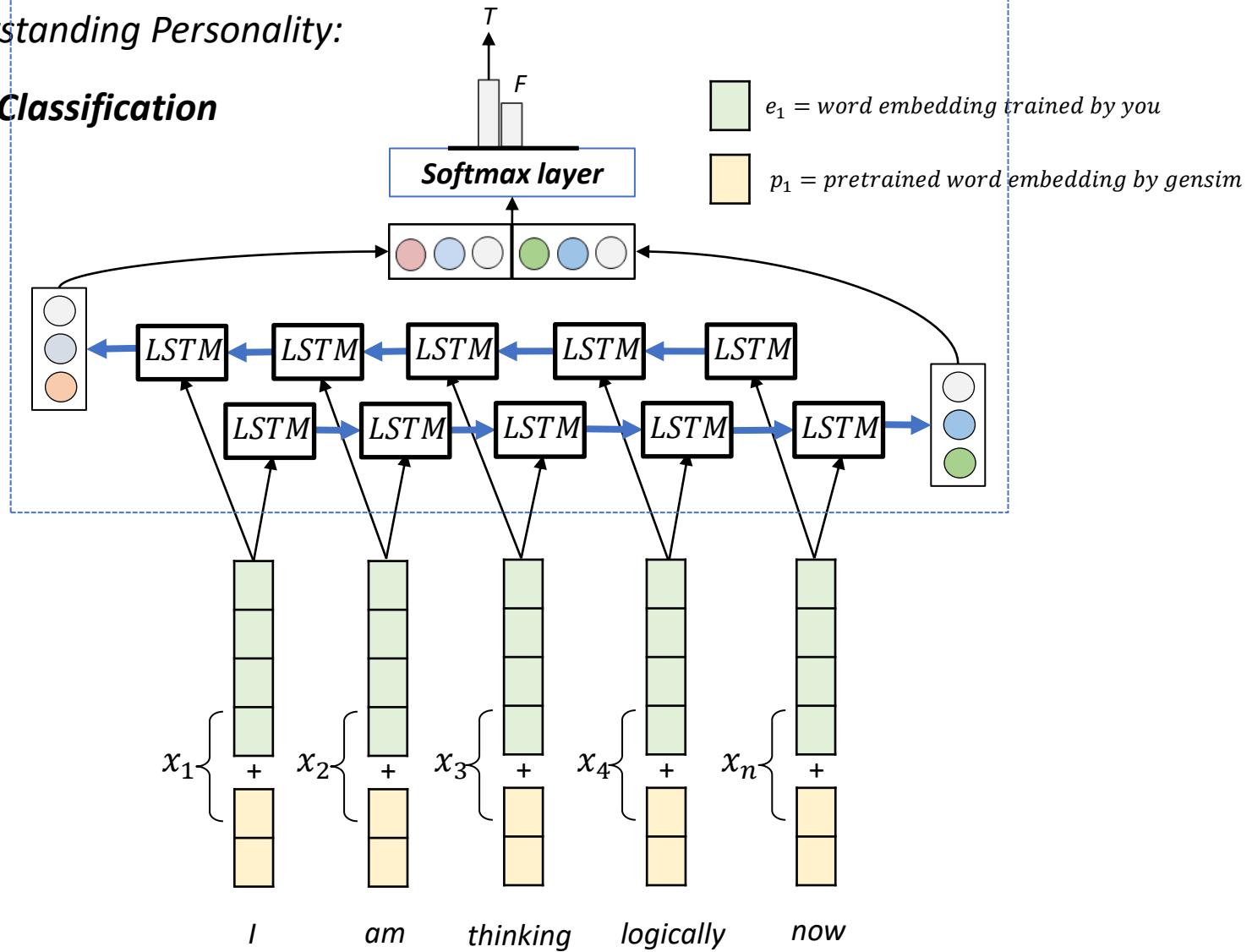
MBTI Classification



2 Assignment 1 Discussion

Understanding Personality:

MBTI Classification



2 Assignment 1 Discussion

Intrinsic word vector evaluation

Evaluation Result Comparison

The Semantic-Syntactic word relationship tests for understanding of a wide variety of relationships as shown below.

Table 2: Results on the word analogy task, given as percent accuracy. Underlined scores are best within groups of similarly-sized models; bold scores are best overall. HPCA vectors are publicly available²; (i)vLBL results are from (Mnih et al., 2013); skip-gram (SG) and CBOW results are from (Mikolov et al., 2013a,b); we trained SG[†] and CBOW[†] using the word2vec tool³. See text for details and a description of the SVD models.

Model	Dim.	Size	Sem.	Syn.	Tot.
ivLBL	100	1.5B	55.9	50.1	53.2
HPCA	100	1.6B	4.2	16.4	10.8
GloVe	100	1.6B	<u>67.5</u>	<u>54.3</u>	<u>60.3</u>
SG	300	1B	61	61	61
CBOW	300	1.6B	16.1	52.6	36.1
vLBL	300	1.5B	54.2	<u>64.8</u>	60.0
ivLBL	300	1.5B	65.2	63.0	64.0
GloVe	300	1.6B	80.8	61.5	<u>70.3</u>
SVD	300	6B	6.3	8.1	7.3
SVD-S	300	6B	36.7	46.6	42.1
SVD-L	300	6B	56.6	63.0	60.1
CBOW [†]	300	6B	63.6	<u>67.4</u>	65.7
SG [†]	300	6B	73.0	66.0	69.1
GloVe	300	6B	<u>77.4</u>	67.0	71.7
CBOW	1000	6B	57.3	68.9	63.7
SG	1000	6B	66.1	65.1	65.6
SVD-L	300	42B	38.4	58.2	49.2
GloVe	300	42B	81.9	69.3	75.0

(Original Glove Paper - Pennington et al.2014)

2 Assignment 1 Discussion

Intrinsic word vector evaluation

Evaluation Result Comparison

The Semantic-Syntactic word relationship tests for understanding of a wide variety of relationships as shown below.

Window-Size (m) and Vector Dimension (N)

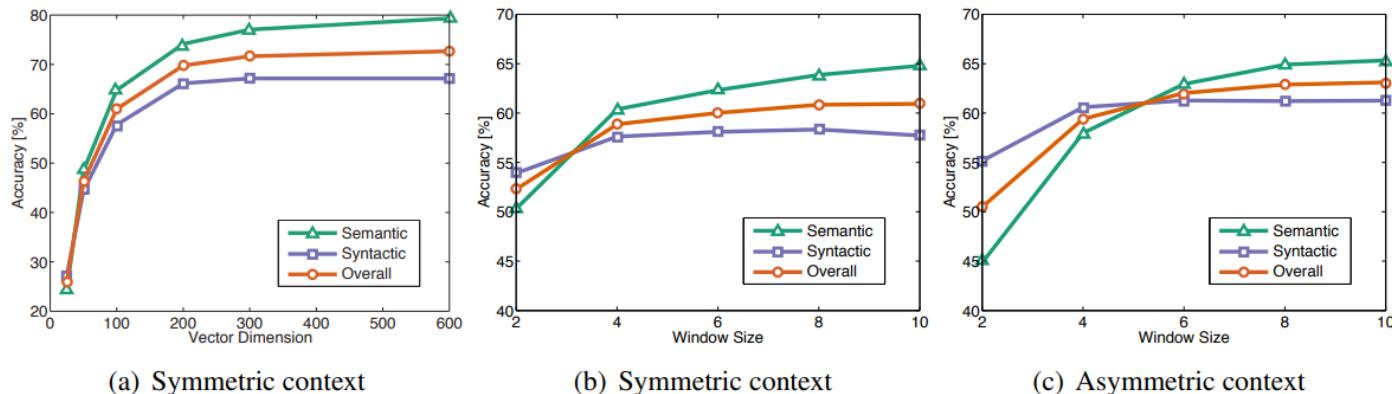
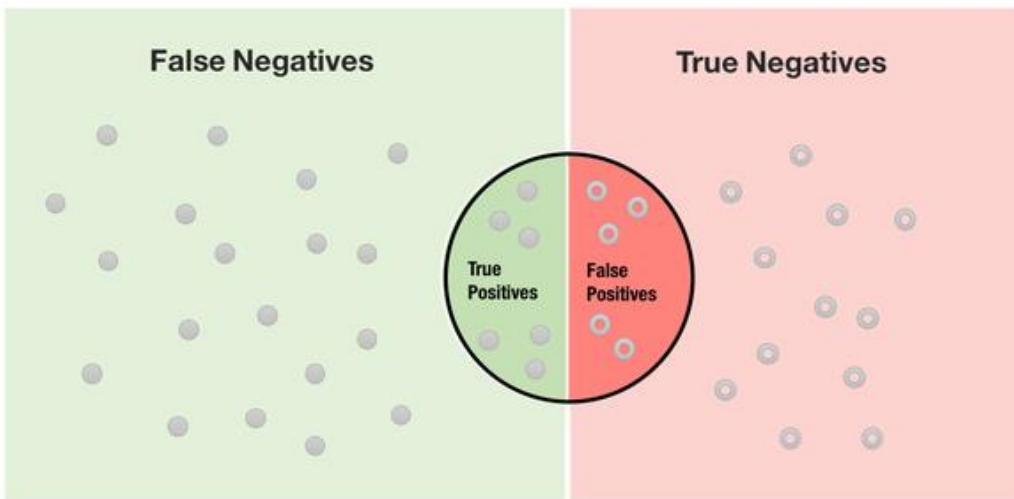


Figure 2: Accuracy on the analogy task as function of vector size and window size/type. All models are trained on the 6 billion token corpus. In (a), the window size is 10. In (b) and (c), the vector size is 100.

(Original Glove Paper - Pennington et al.2014)

2 Assignment 1 Discussion

Assignment 1: Performance Evaluation



$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$specificity = \frac{TN}{TN + FP}$$

2 Assignment 1 Discussion

Assignment 1: Performance Evaluation

Evaluation 2)

Model	F1
Bi-LSTM With URL	Xxx
Bi-LSTM Without URL	Xxx
Bi-LSTM With stopword	Xxx
Bi-LSTM Without stopword	Xxx
...	...

Evaluation 3)

Model	F1
Bi-LSTM with Word2vec (SG)	Xxx
Bi-LSTM with Word2vec (CBOW)	Xxx
Bi-LSTM with Word2vec (CBOW) + glove-twitter-100	Xxx
...	...

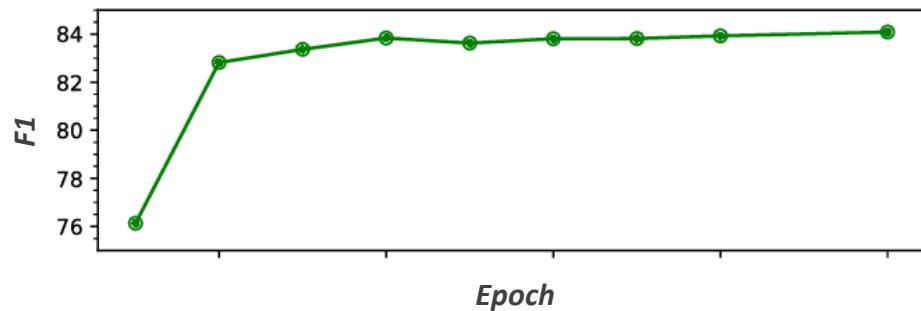
2 Assignment 1 Discussion

Assignment 1: Performance Evaluation

Evaluation 4)

Model	F1
Bi-RNN	Xxx
Bi-LSTM	Xxx
Bi-GRU	Xxx
...	...

Evaluation 5)



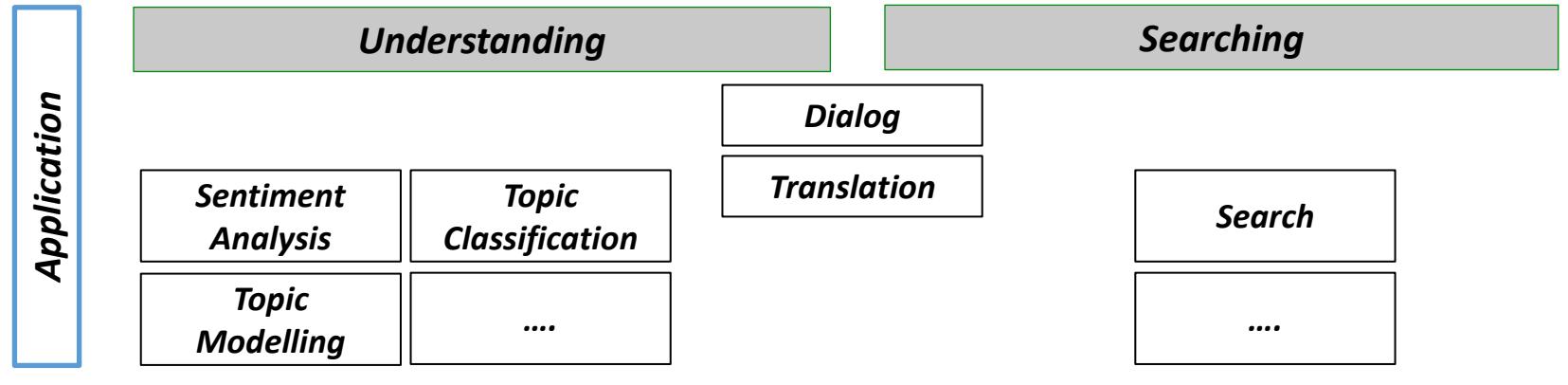
0 LECTURE PLAN

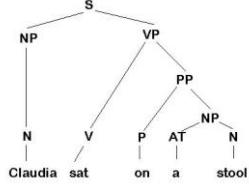
Lecture 5: Assignment1 and Language Fundamental

1. RNN/LSTM, Dealing Context Review
2. Assignment 1 Discussion
3. **Sentiment Analysis**
 1. Sentiment Analysis
 2. Sentiment Analysis: Examples
 3. Sentiment Analysis: Lexicons
4. Language Fundamental
 - Phonology, Morphology, Syntax, Semantics, Pragmatics
5. Text Preprocessing
 1. Tokenization
 2. Cleaning and Normalisation
 3. Stemming and Lemmatisation
 4. Stopword
 5. Regular Expression

3 The NLP Big Picture

The purpose of Natural Language Processing: Overview

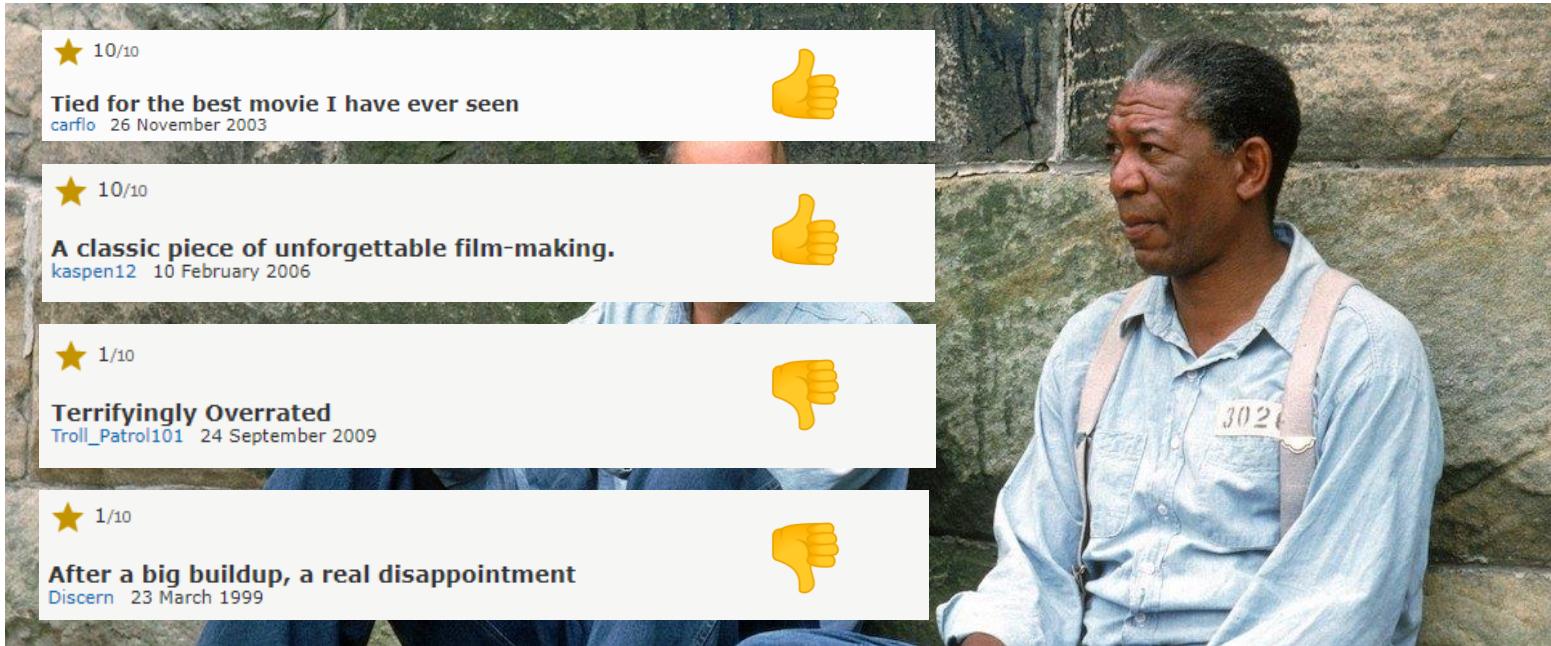


Entity Extraction	When Sebastian Thrun ...	When Sebastian Thrun PERSON started at Google ORG in 2007 DATE
Parsing	Claudia sat on a stool	
PoS Tagging	She sells seashells	[she/PRP] [sells/VBZ] [seashells/NNS]
Stemming	Drinking, Drank, Drunk	Drink
Tokenisation	How is the weather today	[How] [is] [the] [weather] [today]

3

Sentiment Analysis

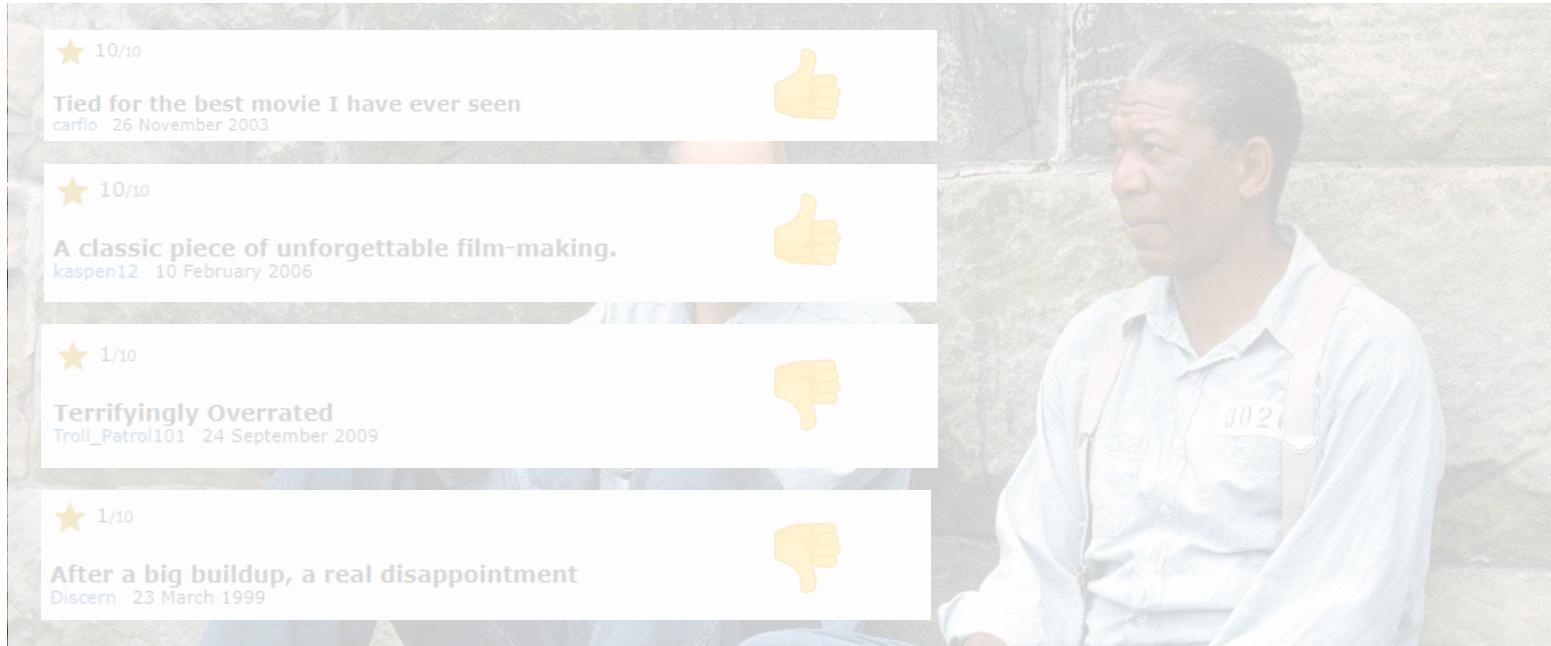
Movie Review – Positive or Negative



Too easy? 😊

3 Sentiment Analysis

What is Sentiment Analysis?



3

Sentiment Analysis

What is Sentiment Analysis?

“Sentiment analysis is the operation of understanding the intent or emotion behind a given piece of text. It is part of text classification, but it is useful for extracting structured information”



Different Names of a ‘Sentiment Analysis’

- *Opinion extraction*
- *Opinion mining*
- *Sentiment mining*
- *Subjectivity analysis*

3

Sentiment Analysis

Sentiment Analysis



Customer reviews

 4.6 out of 5

11,351 customer ratings



[▼ How does Amazon calculate star ratings?](#)

Review this product

Share your thoughts with other customers

[Write a customer review](#)

Top international reviews

 MustLoveDogs

 Just because you CAN flush it, doesn't mean you should!

Reviewed in the United States on 14 July 2018

Style: 8 Packs of Flushable Wipes | [Verified Purchase](#)

Flushable? Not according to the plumber I just paid \$200 to. Be careful folks. Other than the misleading "flushable" advertising, I liked product, but can't afford plumbing bills.

354 people found this helpful

[Helpful](#)

| [Report abuse](#)

 Zack Fischmann

 These are NOT unscented -- one of the ingredients is "fragrance/parfum"

Reviewed in the United States on 15 January 2019

Style: 8 Packs of Flushable Wipes | [Verified Purchase](#)

Sentiment Analysis

What is Sentiment Analysis?

Emotion, Mood, Interpersonal stances, **Attitude**, Personality traits

Typology of Affective States (Scherer et al. 2006)

Attitudes

Enduring, affectively colored beliefs, dispositions towards objects/persons

- *liking, loving, hating, valuing, desiring*

3

Sentiment Analysis

Sentiment Analysis: Examples

Apple iPhone 7 - 128GB - Rose Gold (Unlocked)

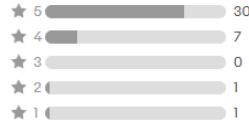
★★★★★ 39 product ratings | About this product



Ratings and reviews

4.6

★★★★★
39 product ratings



Aspects



[Write a review](#)

Most relevant reviews

[See all 24 reviews](#)

★★★★★

by [judeel2](#)
18 Jul, 2019

Excellent phone

Works excellently well, the screen is very very clear. Photos are better than my iPhone 5se, even though they are both 12mp. Front facing camera is 7mp, 5se is less. The only downside is the battery life. It doesn't last all day for me. I have small hands but the larger size isn't too big. Can highly recommend, good value.

Verified purchase: Yes | Condition: Pre-Owned

★★★★★

by [noadaughert_31](#)
26 Apr, 2018

Really good for price

Had virtually no scratches and battery life is optimal despite being refurbished. Good value for your money. Only complaint was that there wasn't any accessories such as the bluetooth ear buds required for listening to music or the lightning to AUX adapter. But no accessories were listed in the description.

Verified purchase: Yes | Condition: Pre-Owned

★★★★★

by [diannpedro_0](#)
03 Jan, 2019

Good practical iPhone.

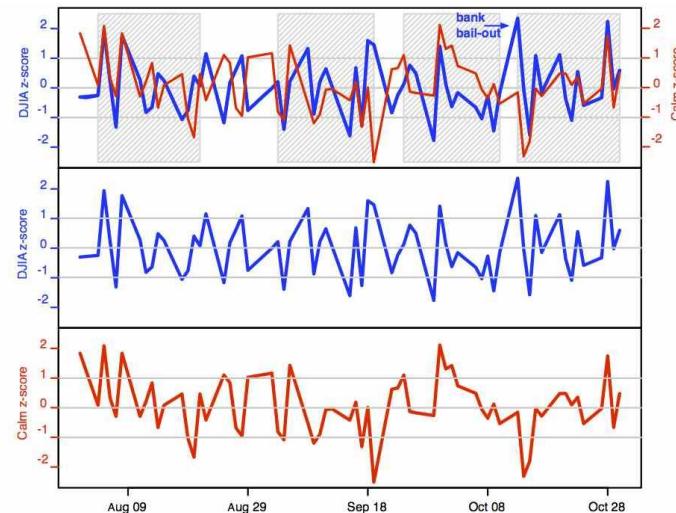
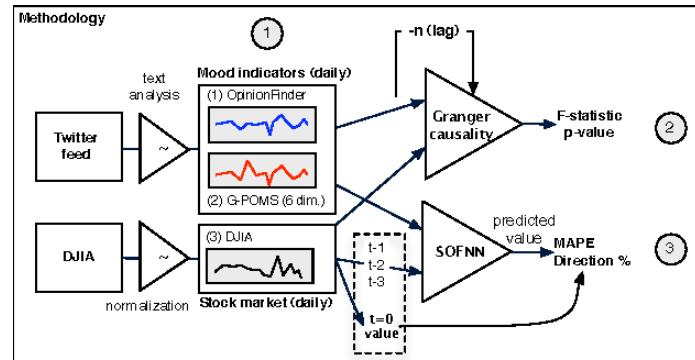
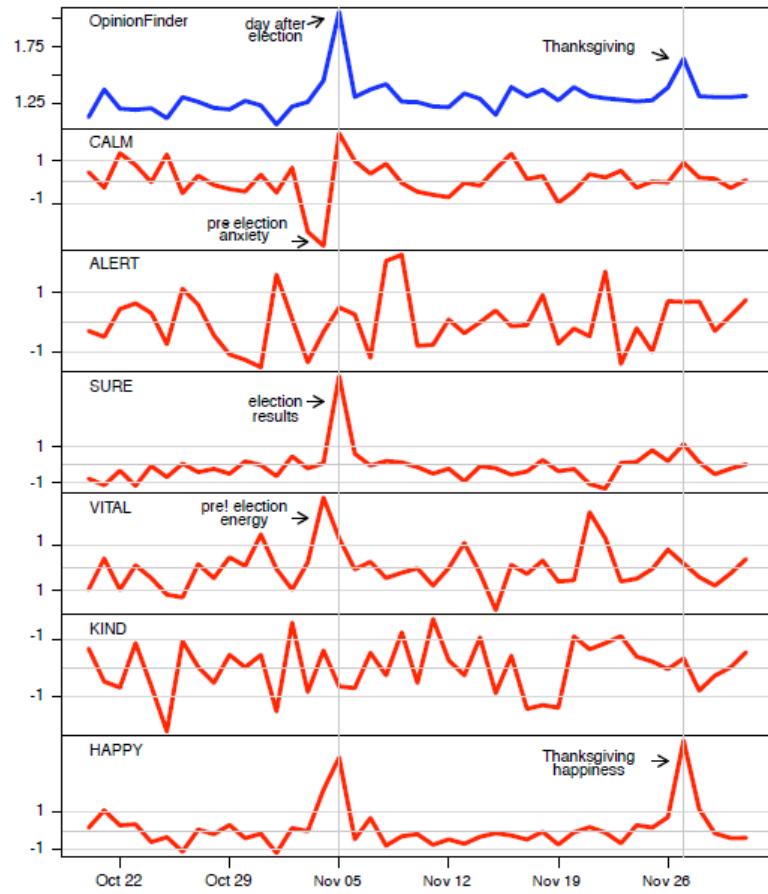
It's just so much better than my previous iPhone 6 as it was damaged & difficult to use. The iPhone 7 feels good to use. I'm not really sure it was the best price as I didn't shop around but am happy regardless,

Verified purchase: Yes | Condition: New

Sentiment Analysis

Sentiment Analysis: Examples

Twitter mood predicts the stock market (Bollen et al. 2011)



3 Sentiment Analysis

Sentiment Analysis: Sentiment viz



3

Sentiment Analysis

Sentiment Analysis Tasks

- ***Movie:*** *Is this review positive or negative?*
- ***Products:*** *what do people think about the new phone?*
- ***Public sentiment:*** *how is consumer confidence? Is despair increasing?*
- ***Politics:*** *what do people think about this candidate or issue?*
- ***Prediction:*** *predict election outcomes or market trends from sentiment*

3

Sentiment Analysis

What will be considered to analyse sentiment

Sentiment analysis = the detection of Attitudes

Enduring, affectively colored beliefs, dispositions towards objects/persons

Main Factors

- **Target Object:** *an entity that can be a product, person, event, organisation, or topic (e.g. iPhone)*
- **Attribute:** *an object usually has two types of attributes*
 - *Components (e.g. touch screen, battery)*
 - *Properties (e.g. size, weight, colour, voice quality)*
 - *Explicit and implicit attributes:*
 - *Explicit attributes: appearing in the attitude (e.g. “the battery life of this phone was not long”)*
 - *Implicit attributes: not appearing in the attitude (e.g. “this phone is too expensive” – the property price)*
- **Attitude Holder:** *the person or organisation that expresses the opinion (e.g. my mother was mad with me)*
- **Type of attitude:** *positive, negative, or neutral or set of types (e.g. happy)*
- **Time:** *the time that expresses the opinion*

3

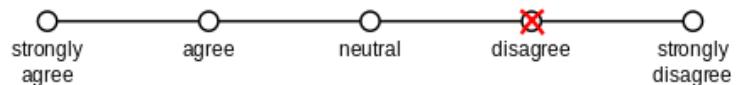
Sentiment Analysis

What is Sentiment Analysis?

- *Basic Task: Is the attitude of this text positive or negative?*



- *More complex task: Rank the attitude of this text from 1 to 5 Likert Scale (1 to 5)*



- *Advanced task: Detect the target, source, or complex attitude types*

3

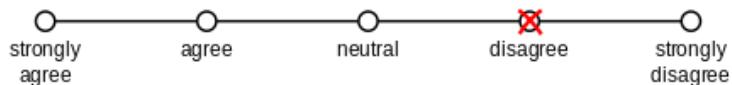
Sentiment Analysis

What is Sentiment Analysis?

- *Basic Task: Is the attitude of this text positive or negative?*



- *More complex task: Rank the attitude of this text from 1 to 5 Likert Scale (1 to 5)*



- *Advanced task: Detect the target, source, or complex attitude types*

Finding aspect/attribute/target of sentiment

Title: Sharp, Solid, but Harder to Hold than iPhone 7

- By Tristan on March 13, 2017

"my thoughts on the iPhone 7 are:

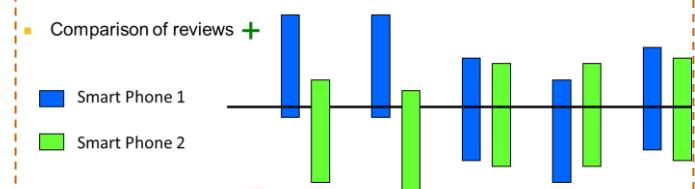
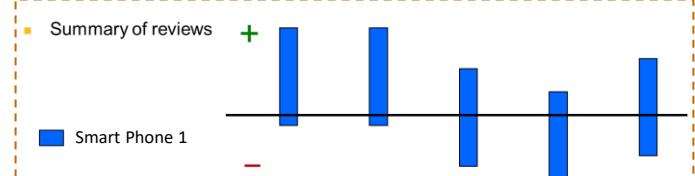
1) Retina display is awesome. Everything looks more defined and sharper. There is much color and clarity out there... or should I say, in those digital images and videos... needless to say, the camera as well captures great images.

....."

Attribute based Summary

- Attribute 1: display
 - Positive
 1. Retina display is awesome
 2. There is much color and clarity out there
 3. ...
- Attribute 2: camera
 - Positive
 1. the camera as well captures great images.
 2.

Attribute based Visualisation



3

Sentiment Analysis

Features Vectors: a bird's eye view

- Word ngrams (up to 4), skip ngrams w/ 1 missing word
- Character ngrams up to 5
- All caps: number of words in capitals
- Number of continuous punctuation marks, either exclamation or question or mixed. Also whether last char contains one of these.
- Presence of emoticons

Classify your Sentiment is a classification problem

- *Typically people have used **Naïve Bayes** or **Support Vector Machines (SVM)** in the past [Mohammad et al. 2013]*
- ***Artificial Neural Nets** are also becoming more popular now [Nogueira dos Santos & Gatti, 2014]*

Useful Sentiment Lexicons

Name	Details
The General Inquirer http://www.wjh.harvard.edu/~inquirer http://www.wjh.harvard.edu/~inquirer/homecat.htm http://www.wjh.harvard.edu/~inquirer/inquirerbasic.xls	<p>Categories</p> <ul style="list-style-type: none"> Positive (1915 words) and Negative (2291 words) Strong vs Weak, Active vs Passive, Overstated versus Understated Pleasure, Pain, Virtue, Vice, Motivation, Cognitive Orientation, etc <p>Free to use</p>
LIWC Linguistic Inquiry and Word Count http://www.liwc.net/	<p>2300 words and less than 70 classes</p> <p>Affective Processes</p> <ul style="list-style-type: none"> negative emotion (bad, weird, hate, problem, tough) positive emotion (love, nice, sweet) <p>Cognitive Processes</p> <ul style="list-style-type: none"> Tentative (maybe, perhaps, guess), Inhibition (block, constraint) Pronouns, Negation (no, never), Quantifiers (few, many) <p>\$30 or \$90 fee</p>
MPQA Subjectivity Cues Lexicon http://www.cs.pitt.edu/mpqa/subj_lexicon.html	<p>Each word annotated for intensity (strong, weak)</p> <p>6885 words from 8221 lemmas</p> <ul style="list-style-type: none"> 2718 positive 4912 negative <p>GNU GPL (widely-used free software license)</p>
Opinion Lexicon http://www.cs.uic.edu/~liub/FBS/opinion--lexicon--English.rar	<p>6786 words</p> <ul style="list-style-type: none"> 2006 positive/ 4783 negative <p>Free to use</p>
SentiWordNet http://swn.isti.cnr.it/	<p>All WordNet synsets automatically annotated for degrees of positivity, negativity, and neutrality/objectiveness</p> <ul style="list-style-type: none"> [estimable(J,3)] "may be computed or estimated" Pos 0 Neg 0 Obj 1 [estimable(J,1)] "deserving of respect or high regard" Pos .75 Neg 0 Obj .25 <p>Free to use</p>

3 Sentiment Analysis

Can you build the sentiment lexicon by yourself?

Bootstrap style: Semi-supervised learning of lexicons

- *Use a small amount of information*
- *A few labeled examples*
- *A few hand-built patterns*
- *Bootstrapping a lexicon*



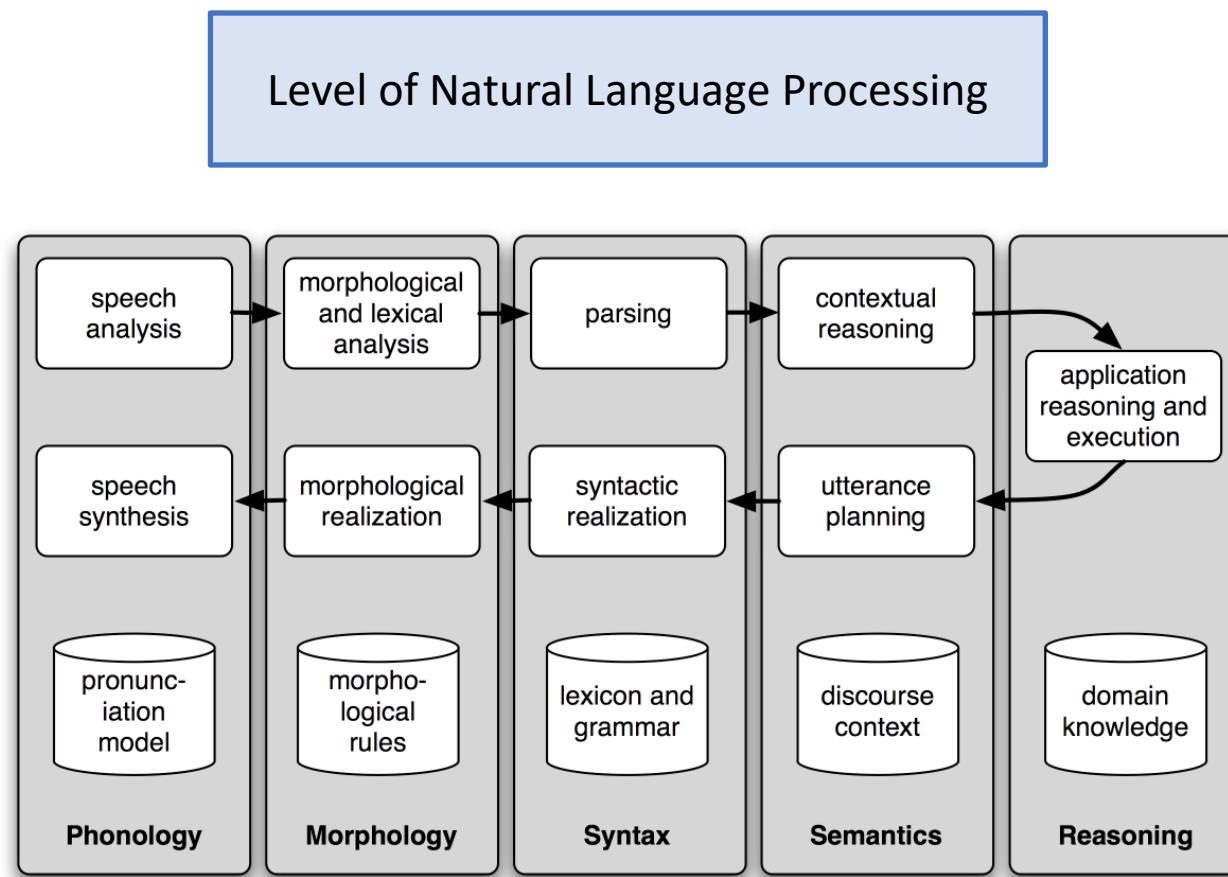
0 LECTURE PLAN

Lecture 5: Assignment1 and Language Fundamental

1. RNN/LSTM, Dealing Context Review
2. Assignment 1 Discussion
3. Sentiment Analysis
 1. Sentiment Analysis
 2. Sentiment Analysis: Examples
 3. Sentiment Analysis: Lexicons
4. **Language Fundamental**
 - Phonology, Morphology, Syntax, Semantics, Pragmatics
5. Text Preprocessing
 1. Tokenization
 2. Cleaning and Normalisation
 3. Stemming and Lemmatisation
 4. Stopword
 5. Regular Expression

4 Language Fundamental

Level of Natural Language Processing



Phonology
All sound system

Pragmatics
Language use

We know the sounds of our language

Which sounds are in our language and which sounds are not

- For example, English speakers know the [ŋ] sound (in sing) does not appear at the beginning of a word
- Does this mean that [ŋ] cannot appear at the beginning of words in all human languages?



NO! — Nguyen Tran



NO! — Andrew Ng

We know how sounds can combine

Often shown when a word from one language is borrowed into another:



- McDonalds — in English consonant clusters allowed ([mk] and [ldz]) becomes...

マクドナルド

Makudonarudo

麦当劳

Màidāngláo

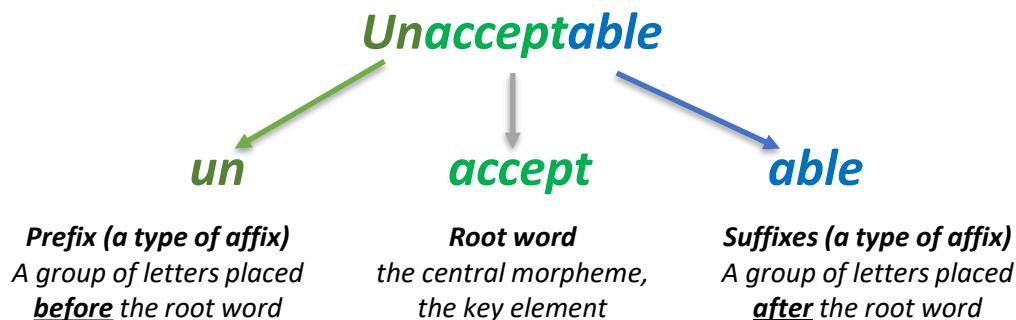
맥도날드

Maegdonaldeu

in other language — consonant clusters are not allowed

Morphology: Pieces of words

- A field of linguistics focused on the study of the ***forms and formation of words in a language***
- Words in a language consist of one element or elements of meaning which are ***morphemes***
 - ***Morphemes*** are the pieces of words: bases, roots and affixes (pre-fix, suffix).



Morphology: Pieces of words

- A field of linguistics focused on the study of the ***forms and formation of words in a language***
- Words in a language consist of one element or elements of meaning which are **morphemes**
 - **Morphemes** are the pieces of words: bases, roots and affixes.
- walk walked walking walks walk walk -ed walk -ing walk -s

Natural Language Processing Level

- **Phonology/Morphology: the structure of words**
 - *Unusually* is composed of a prefix *un-*, a stem *usual*, and an affix *-ly*. *Learned* is *learn* plus the inflectional affix *-ed*
- **Syntax: the way words are used to form phrases**
 - It is part of English syntax that a determiner such as *the* will come before a noun, and also that determiners are obligatory with certain singular noun.
- **Semantics: Compositional and lexical semantics**
 - Compositional semantics: the construction of meaning based on syntax
 - Lexical semantics: the meaning of individual words
- **Pragmatics: meaning in context**
 - *Do you have the time?* – means ‘can you tell me what time is it now?’

0 LECTURE PLAN

Lecture 5: Assignment1 and Language Fundamental

1. RNN/LSTM, Dealing Context Review
2. Assignment 1 Discussion
3. Sentiment Analysis
 1. Sentiment Analysis
 2. Sentiment Analysis: Examples
 3. Sentiment Analysis: Lexicons
4. Language Fundamental
 - Phonology, Morphology, Syntax, Semantics, Pragmatics
5. Text Preprocessing
 1. Tokenization
 2. Cleaning and Normalisation
 3. Stemming and Lemmatisation
 4. Stopword
 5. Regular Expression

5 Text Preprocessing

Text Preprocessing

- Every NLP task needs to do text pre-processing
 - Segmenting/tokenizing words in running text
 - Normalizing word formats
 - Segmenting sentences in running text

5 Text Preprocessing

How many words?

- Type: an element of the vocabulary.
- Token: an instance of that type in running text.
- How many of them in the sentence?
 - 14 tokens
 - 13 types (or 12) (or 11?)

they lay back on the Sydney grass and looked at the stars and their

- **Token** = number of tokens
- **Type** = vocabulary = set of types
 - $|V|$ is the size of the vocabulary

5

Text Preprocessing

How many words?

- N = number of tokens
- V = vocabulary = set of types
 - $|V|$ is the size of the vocabulary

	Tokens = N	Types = $ V $
Switchboard phone conversations	2.4 million	20 thousand
Shakespeare	884,000	31 thousand
Google N-grams	1 trillion	13 million

5 Text Preprocessing

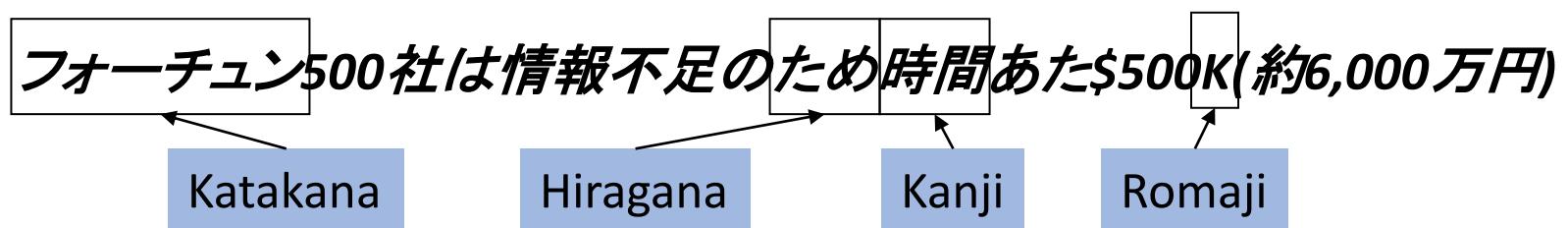
Tokenization: language issues

- French
 - L'ensemble → one token or two?
 - L ? L' ? Le ?
 - Want l'ensemble to match with un ensemble
 - Until 2003, Google cannot make this work
- German noun compounds are not segmented
 - *Lebensversicherungsgesellschaftsangestellter*
 - ‘life insurance company employee’
 - German information retrieval needs ***compound splitter***

5 Text Preprocessing

Tokenization: language issues

- Chinese has no spaces between words:
 - 悉尼大学位于澳大利亚悉尼
 - 悉尼大学 位于 澳大利亚 悉尼
 - University of Sydney is located in Sydney, Australia
- Further complicated in Japanese, with multiple alphabets intermingled
 - Dates/amounts in multiple formats



5 Text Preprocessing

Tokenization: language issues

- Arabic (or Hebrew) is basically written right to left, but with certain items like numbers written left to right
- Words are separated, but letter forms within a word form complex ligatures

← → ← → ← start

اسْتَقْلَتُ الْجَزَائِرُ فِي سَنَةِ 1962 بَعْدَ 132 عَامًا مِنْ الْاحْتِلَالِ الْفَرْنَسِيِّ.

- ‘Algeria achieved its independence in 1962 after 132 years of French occupation.’
- With Unicode, the order of characters in files matches the conceptual order, and the reversal of displayed characters is handled by the rendering system.

5 Text Preprocessing

Normalization

- Need to “normalize” terms
 - Information Retrieval: indexed text & query terms must have same form.
 - We want to match U.S.A. and USA
- We implicitly define equivalence classes of terms
 - e.g., deleting periods in a term
- Alternative: asymmetric expansion:
 - Enter: window Search: window, windows
 - Enter: windows Search: Windows, windows, window
 - Enter: Windows Search: Windows
- Potentially more powerful, but less efficient

5 Text Preprocessing

Case Folding

- Applications like IR: ***convert all letters to lower case***
 - Since users tend to use lower case
 - Possible exception: upper case in mid-sentence?
 - e.g., General Motors
 - Fed vs. fed
 - SAIL vs. sail
- For sentiment analysis, Machine Translation, Information extraction
 - Case is helpful (US versus us is important)

5 Text Preprocessing

Lemmatization

- Reduce inflections or variant forms to **base form**
 - am, are, is → be
 - car, cars, car's, cars' → car
- *the boy's cars are different colors* → *the boy car be different color*
- Lemmatization: have to find correct dictionary headword form
 - *Machine translation*
 - Spanish quiero ('I want'), quieres ('you want') same lemma as querer 'want'

5 Text Preprocessing

Morphology

- Morphemes:
 - The small meaningful units that make up words
 - **Stems:** The core meaning-bearing units
 - **Affixes:** Bits and pieces that adhere to stems
 - Often with grammatical functions

5 Text Preprocessing

Stemming

- Reduce terms to their stems in information retrieval
- Stemming is crude chopping of affixes
 - language dependent
 - e.g., *automate(s)*, *automatic*, *automation* all reduced to *automat.*

for example compressed and compression are both accepted as equivalent to compress.

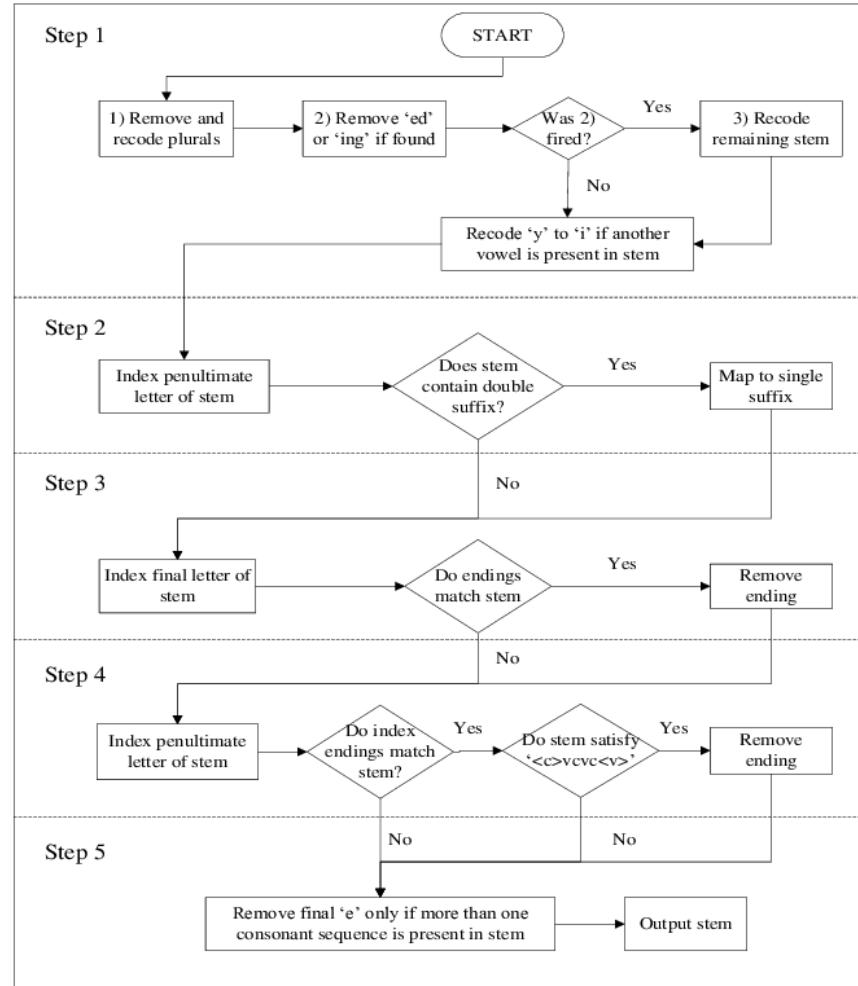


for exampl compress and compress ar both accept as equival to compress

5 Text Preprocessing

Porter's algorithm: The most common English stemmer

Porter Stemming Algorithm



5 Text Preprocessing

Dealing with complex morphology is sometimes necessary

- Some languages require complex morpheme segmentation
 - Turkish
 - Uygar *lastiramidaklarimizdanmissinizcasina*
 - `(behaving) as if you are among those whom we could not civilize'
 - Uygar 'civilized' + *las* 'become'
 - + *tir* 'cause' + *ama* 'not able'
 - + *dik* 'past' + *lar* 'plural'
 - + *imiz* 'p1pl' + *dan* 'abl'
 - + *mis* 'past' + *siniz* '2pl' + *casina* 'as if'

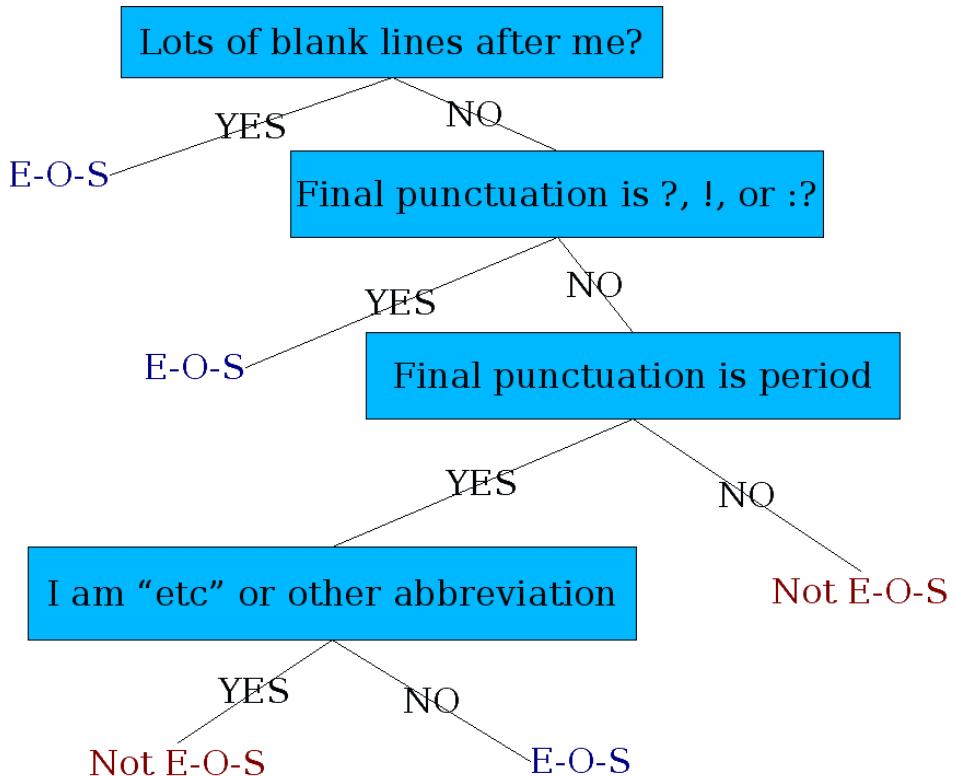
5 Text Preprocessing

Sentence Segmentation

- !, ? are relatively unambiguous
- Period “.” is quite ambiguous
 - Sentence boundary
 - Abbreviations like Inc. or Dr.
 - Numbers like .02% or 4.3
- Build a binary classifier
 - Looks at a “.”
 - Decides EndOfSentence/NotEndOfSentence
 - Classifiers: hand-written rules, regular expressions, or machine-learning

5 Text Preprocessing

Sentence Segmentation using a Decision Tree



5 Text Preprocessing

Implementing Decision Trees or other classifiers

- A decision tree is just an if-then-else statement
- The interesting research is choosing the features
- Setting up the structure is often too hard to do by hand
 - Hand-building only possible for very simple features, domains
 - For numeric features, it's too hard to pick each threshold
 - Instead, structure usually learned by machine learning from a training corpus
- As features that could be exploited by any kind of classifier
 - Logistic regression
 - SVM
 - Neural Nets
 - etc.

5 Text Preprocessing

Regular expressions

- A formal language for specifying text strings
- How can we search for any of these?
 1. woodchuck
 2. woodchucks
 3. Woodchuck
 4. Woodchucks



5 Text Preprocessing

Regular Expressions: Disjunctions

- Letters inside square brackets []

Pattern	Matches
[wW]oodchuck	Woodchuck, woodchuck
[1234567890]	Any digit

- Ranges [A-Z]

Pattern	Matches	
[A-Z]	An upper case letter	Drenched Blossoms
[a-z]	A lower case letter	my beans were impatient
[0-9]	A single digit	Chapter 1: Down the Rabbit Hole

5 Text Preprocessing

Regular Expressions: Negation in Disjunction

- Negations [^Ss]
 - Carat means negation only when first in []

Pattern	Matches	
[^A-Z]	Not an upper case letter	O <u>y</u> fn pripetchik
[^Ss]	Neither 'S' nor 's'	<u>I</u> have no exquisite reason"
a^b	The pattern a carat b	Look up <u>a^b</u> now

5 Text Preprocessing

Regular Expressions: More Disjunction

- Woodchucks is another name for groundhog!
- The pipe | for disjunction

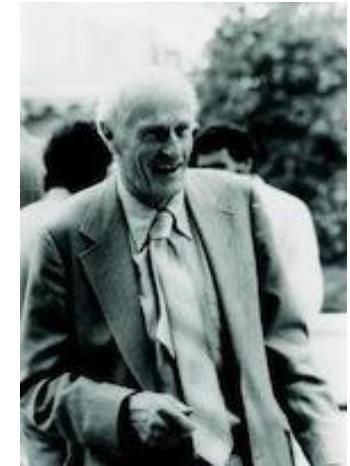
Pattern	Matches
groundhog woodchuck	
yours mine	yours mine
a b c	= [abc]
[gG] roundhog [Ww]oodchuck	



5 Text Preprocessing

Regular Expressions: ? * + .

Pattern	Matches	
colou?r	Optional previous char	<u>color</u> <u>colour</u>
oo*h!	0 or more of previous char	<u>oh!</u> <u>ooh!</u> <u>oooh!</u> <u>ooooh!</u>
o+h!	1 or more of previous char	<u>oh!</u> <u>ooh!</u> <u>oooh!</u> <u>ooooh!</u>
baa+		<u>baa</u> <u>baaa</u> <u>baaaa</u> <u>baaaaa</u>
beg.n		<u>begin</u> <u>begun</u> <u>begun</u> <u>beg3n</u>



Stephen C Kleene

Kleene *, Kleene +

5 Text Preprocessing

Regular Expressions: Anchors ^ \$

Pattern	Matches
<code>^ [A-Z]</code>	<u>P</u> alo Alto
<code>^ [^A-Za-z]</code>	<u>1</u> <u>"Hello"</u>
<code>\. \$</code>	The end <u>.</u>
<code>. \$</code>	The end <u>?</u> The end <u>!</u>

5 Text Preprocessing

Summary

- Regular expressions play a surprisingly large role
 - Sophisticated sequences of regular expressions are often the first model for any text processing task
- For many hard tasks, we use machine learning classifiers
 - But regular expressions are used as features in the classifiers
 - Can be very useful in capturing generalizations

/ Reference

Reference

- Serban, Iulian V., Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015. "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models.

COMP5046

Natural Language Processing

Lecture 6: Part of Speech Tagging

Dr. Caren Han

Semester 1, 2022

School of Computer Science,
University of Sydney



1 The course topics

What will you learn in this course?

Week 1: Introduction to Natural Language Processing (NLP)

Week 2: Word Embeddings (Word Vector for Meaning)

Week 3: Word Classification with Machine Learning I

Week 4: Word Classification with Machine Learning II

NLP and
Machine
Learning

Week 5: Language Fundamental

Week 6: Part of Speech Tagging

Week 7: Dependency Parsing

Week 8: Language Model and Natural Language Generation

NLP
Techniques

Week 9: Information Extraction: Named Entity Recognition

Week 10: Advanced NLP: Attention and Reading Comprehension

Week 11: Advanced NLP: Transformer and Machine Translation

Week 12: Advanced NLP: Pretrained Model in NLP

Advanced
Topic

Week 13: Future of NLP and Exam Review

0 LECTURE PLAN

Lecture 6: Part of Speech Tagging

1. **Part-of-Speech Tagging**
2. Baseline Approaches
 1. Rule-based Model
 2. Look-up Table Model
 3. N-Gram Model
3. Probabilistic Approaches
 1. Hidden Markov Model
 2. Conditional Random Field
4. Deep Learning Approaches

Part-of-Speech Tagging

Parts of Speech (or word classes)

A class of words based on the word's function, the way it works in a sentence

8 parts of speech are commonly listed

2000 years ago (starting with Aristotle)

<i>Nouns</i>	<i>Verbs</i>	<i>Pronouns</i>	<i>Prepositions</i>
<i>Adverbs</i>	<i>Conjunctions</i>	<i>Participles</i>	<i>Articles</i>

Dionysius Thrax of Alexandria (c. 100 BCE)

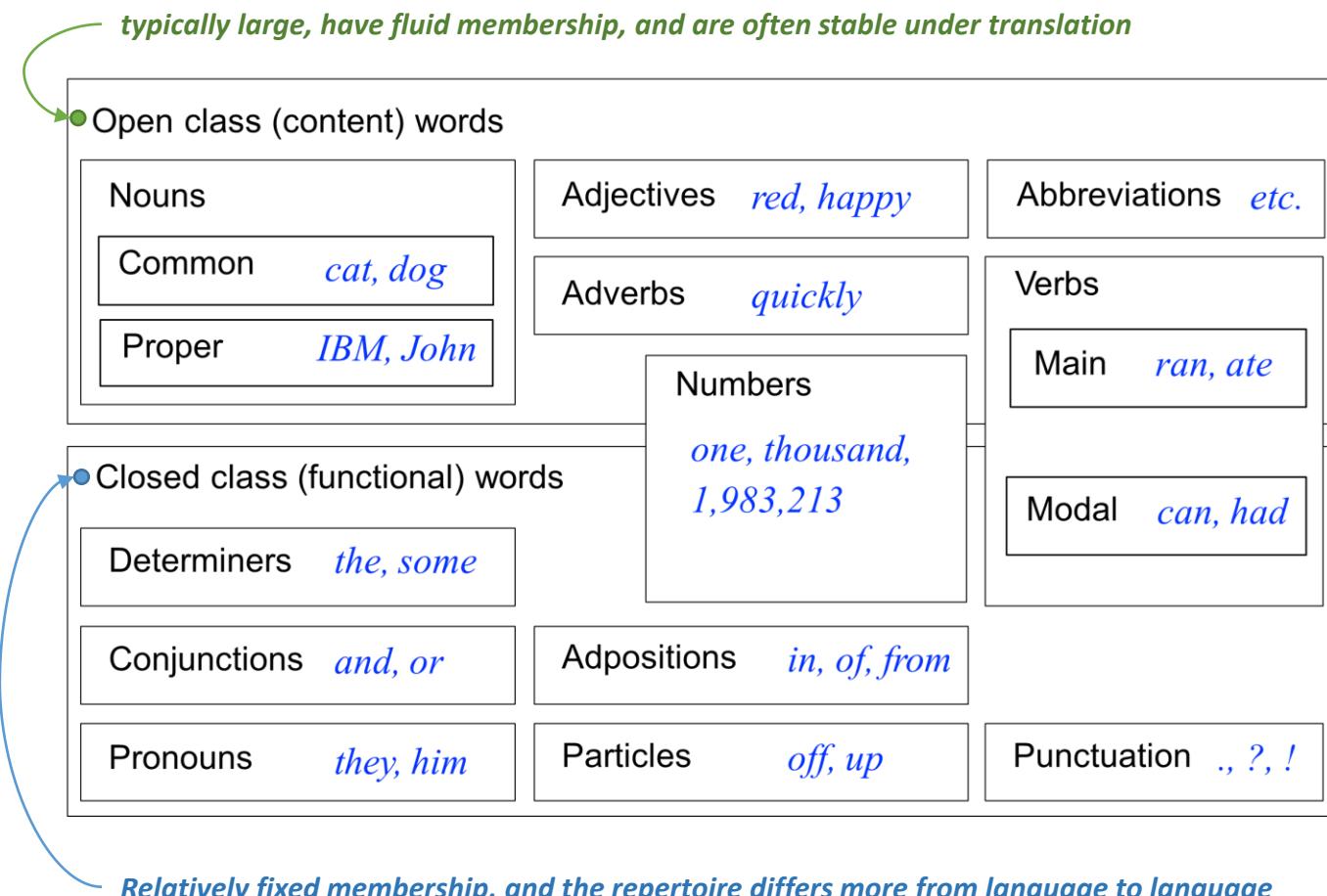
Now (School Grammar) + articles or determiner

<i>Nouns</i>	<i>Verbs</i>	<i>Pronouns</i>	<i>Prepositions</i>
<i>Adverbs</i>	<i>Conjunctions</i>	<i>Adjectives</i>	<i>Interjections</i>

1 Part-of-Speech Tagging

Part-of-Speech (English)

One basic kind of linguistic structure: syntactic word classes



Part-of-Speech Tagging

Part-of-Speech Tag sets – Modern English

In modern (English) NLP, **larger** and **more fine-grained** tag sets are preferred.

Example

Penn Treebank 45 tags <http://bit.ly/1gwbird>

Brown Corpus 87 tags <https://bit.ly/2FGtdLd>

C7 Tagset 146 tags <http://bit.ly/1Mh36KX>

Trade-off between complexity and precision and whatever tag-set we use,
there will be some words that are hard to classify.

1 Part-of-Speech Tagging

POS tags in Penn Treebank

The Penn Treebank POS tagset.

1. CC	Coordinating conjunction	25. TO	<i>to</i>
2. CD	Cardinal number	26. UH	Interjection
3. DT	Determiner	27. VB	Verb, base form
4. EX	Existential <i>there</i>	28. VBD	Verb, past tense
5. FW	Foreign word	29. VBG	Verb, gerund/present participle
6. IN	Preposition/subordinating conjunction	30. VBN	Verb, past participle
7. JJ	Adjective	31. VBP	Verb, non-3rd ps. sing. present
8. JJR	Adjective, comparative	32. VBZ	Verb, 3rd ps. sing. present
9. JJS	Adjective, superlative	33. WDT	<i>wh</i> -determiner
10. LS	List item marker	34. WP	<i>wh</i> -pronoun
11. MD	Modal	35. WP\$	Possessive <i>wh</i> -pronoun
12. NN	Noun, singular or mass	36. WRB	<i>wh</i> -adverb
13. NNS	Noun, plural	37. #	Pound sign
14. NNP	Proper noun, singular	38. \$	Dollar sign
15. NNPS	Proper noun, plural	39. .	Sentence-final punctuation
16. PDT	Predeterminer	40. ,	Comma
17. POS	Possessive ending	41. :	Colon, semi-colon
18. PRP	Personal pronoun	42. (Left bracket character
19. PP\$	Possessive pronoun	43.)	Right bracket character
20. RB	Adverb	44. "	Straight double quote
21. RBR	Adverb, comparative	45. '	Left open single quote
22. RBS	Adverb, superlative	46. "	Left open double quote
23. RP	Particle	47. '	Right close single quote
24. SYM	Symbol (mathematical or scientific)	48. "	Right close double quote

Part-of-Speech Tagging

Criteria for part-of-speech tagging

Three different criteria might be considered.

- ***Distributional*** criteria: Where can the words occur?
- ***Morphological*** criteria: What form does the word have? (E.g. -tion, -ize). What affixes can it take? (E.g. -s, -ing, -est).
- ***Notional(or semantic)*** criteria: What sort of concept does the word refer to? (E.g. nouns often refer to ‘people, places or things’). More problematic: less useful for us

1 Part-of-Speech Tagging

Criteria for part-of-speech tagging: Nouns

Three different criteria might be considered.

- **Distributional** criteria: Where can the nouns appear?

For example, nouns can appear with possession: "his car", "her idea".

- **Morphological** criteria: What form does the word have? (E.g. -tion, -ize). What affixes can it take? (E.g. -s, -ing, -est).

ness, -tion, -ity, and -ance tend to indicate nouns. (happiness, exertion, levity, significance).

- **Notional(or semantic)** criteria: What sort of concept does the word refer to?

Nouns generally refer to living things (mouse), places (Sydney), non-living things (computer), or concepts (marriage).

Part-of-Speech Tagging

Criteria for part-of-speech tagging: **Verbs**

Three different criteria might be considered.

- **Distributional** criteria: Where can the verbs appear?

Different types of verbs have different distributional properties. For example, base form verbs can appear as infinitives: "to jump", "to learn".

- **Morphological** criteria: What form does the word have? (E.g. -tion, -ize). What affixes can it take? (E.g. -s, -ing, -est).

words that end in -ate or -ize tend to be verbs, and ones that end in -ing are often the present participle of a verb (automate, equalize; rising, washing)

- **Notional(or semantic)** criteria: What sort of concept does the word refer to?

Verbs refer to actions (observe, think, give).

1 Part-of-Speech Tagging

Example of POS inference

Emma has a beautiful flower

NNP VBZ DT JJ NN



Parts-of-speech.Info

POS tagging [about Parts-of-speech.info](#)

Enter a **complete sentence** (no single words!) and click at "POS-tag!". The tagging works better when grammar and orthography are correct.

Text:

Emma has a beautiful flower

English

Adjective
Adverb
Conjunction
Determiner
Noun
Number
Preposition
Pronoun
Verb

Part-of-Speech Tagging

POS Tagging: Issue

Given an input text, tag each word correctly:

*There/ was/ still/ lemonade/ in/ the/ **bottle/***

- (Tag sets are quite counterintuitive!)
 - In the above, the **bottle** is a noun not a verb
 - *but how does our tagger tell?*
 - The *still* could be an adjective or an adverb
 - *which seems more likely?*

1 Part-of-Speech Tagging

POS Tagging: Issue

Given an input text, tag each word correctly:

*There/ was/ **still/** lemonade/ in/ the/ **bottle/***

- (Tag sets are quite counterintuitive!)
 - In the above, the **bottle** is a noun not a verb
 - *but how does our tagger tell?*
 - The **still** could be an adjective or an adverb
 - *which seems more likely?*

adjective, still-er, still-est.

- 1 remaining in place or at rest; motionless; stationary:
to stand still.
- 2 free from sound or noise, as a place or persons; silent:
to keep still about a matter.
- 3 subdued or low in sound; hushed:
a still, small voice.
- 4 free from turbulence or commotion; peaceful; tranquil; calm:
the still air.
- 5 without waves or perceptible current; not flowing, as water.
- 6 not effervescent or sparkling, as wine.
- 7 *Photography.* noting, pertaining to, or used for making single photographs, as opposed to a motion picture

adverb

- 10 at this or that time; as previously:
Are you still here?
- 11 up to this or that time; as yet:
A day before departure we were still lacking an itinerary.
- 12 in the future as in the past:
Objections will still be made.
- 13 even; in addition; yet (used to emphasize a comparative):
still more complaints; still greater riches.
- 14 even then; yet; nevertheless:
to be rich and still crave more.

1 Part-of-Speech Tagging

The purpose of POS Tagging

Essential ingredient in natural language applications

- Useful in and of itself (more than you'd think)
 - Text-to-speech: record, lead
 - Lemmatization: saw[v] see, saw[n] saw
 - Linguistically motivated word clustering
- Useful as a pre-processing step for parsing
- Useful as features to downstream systems.

0 LECTURE PLAN

Lecture 6: Part of Speech Tagging

1. Part-of-Speech Tagging
2. **Baseline Approaches**
 1. Rule-based Model
 2. Look-up Table Model
 3. N-Gram Model
3. Probabilistic Approaches
 1. Hidden Markov Model
 2. Conditional Random Field
4. Deep Learning Approaches

2 Baseline Approaches

Part of Speech Tagging

N

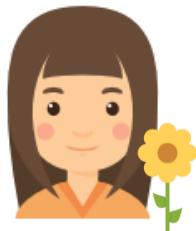
V

D

A

N

Emma has a beautiful flower



2 Baseline Approaches

Part of Speech Tagging

N

V

D

A

N

Emma has a beautiful flower



Open class (content) words		
Nouns	Adjectives	Abbreviations
Common <i>cat, dog</i>	<i>red, happy</i>	<i>etc.</i>
Proper <i>IBM, John</i>		
Closed class (functional) words		
Determiners <i>the, some</i>		Verbs
Conjunctions <i>and, or</i>		Main <i>ran, ate</i>
Adpositions <i>in, of, from</i>		Modal <i>can, had</i>
Pronouns <i>they, him</i>	Particles <i>off, up</i>	Punctuation <i>, ? , !</i>

NOTE: The example includes the high level of classes from the PoS tagset

2 Baseline Approaches

Rule-based POS Tagging

Basic idea:

Old POS taggers used to work in two stages, based on hand-written rules:

- the first stage identifies a set of possible POS for each word in the sentence (based on a lexicon), and
- the second uses a set of hand-crafted rules in order to select a POS from each of the lists for each word

**IF Condition,
Then Conclusion**

2 Baseline Approaches

Rule-based POS Tagging

Basic idea:

- Assign each token all its possible tags.
- Apply rules that eliminate all tags for a token that are inconsistent with its context.

Example

the	DT (determiner)		the	DT (determiner)	
can	MD (modal)	⇒	can	MD (modal)	X
	NN (sg noun)			NN (sg noun)	✓
	VB (base verb)			VB (base verb)	X

- Assign any unknown word tokens a tag that is consistent with its context (eg, the most frequent tag).

2 Baseline Approaches

Rule-based POS Tagging

- Rule-based tagging often used a large set of hand-crafted context-sensitive rules.

Example (schematic):

Example

the	DT (determiner)		the	DT (determiner)	
can	MD (modal)	⇒	can	MD (modal)	X
	NN (sg noun)			NN (sg noun)	✓
	VB (base verb)			VB (base verb)	X

“Cannot eliminate all POS ambiguity.”

2 Baseline Approaches

Part of Speech Tagging



Emma



John



Will

Emma likes John

2 Baseline Approaches

Part of Speech Tagging



Emma



John



Will

Emma likes John

Database

John likes Will

N V N

Will likes Emma

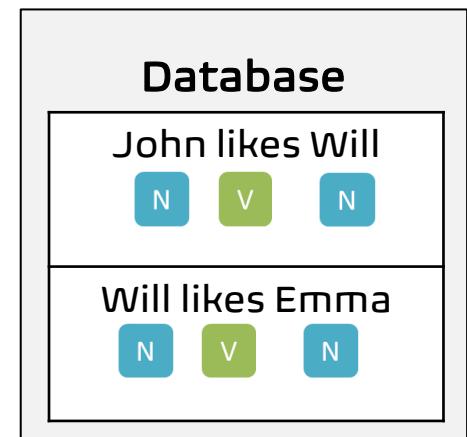
N V N

2 Baseline Approaches

Part of Speech Tagging: Lookup Table

	N	V
John	1	0
likes	0	2
Will	2	0
Emma	1	0

Emma likes John

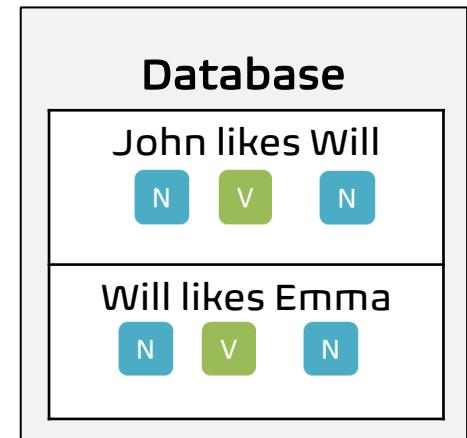


2 Baseline Approaches

Part of Speech Tagging: Lookup Table

	N	V
John	1	0
likes	0	2
Will	2	0
Emma	1	0

Emma likes John



2 Baseline Approaches

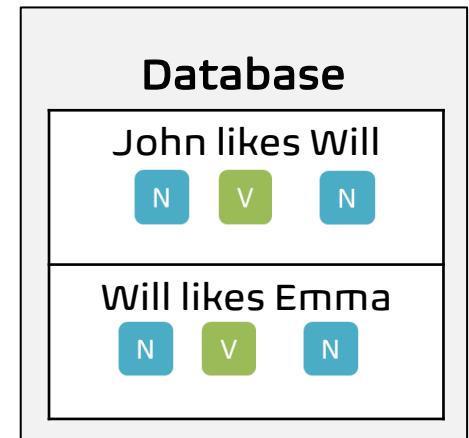
Part of Speech Tagging: Lookup Table

Pick the largest number of the corresponding row

	N	V
John	1	0
likes	0	2
Will	2	0
Emma	1	0



Emma likes John



What about more complicated sentences?

2 Baseline Approaches

Part of Speech Tagging



Emma



John



Will

Emma will meet Will

Database

John will meet Will

N M V N

Emma will meet John

N M V N

Will will meet Emma

N M V N

2 Baseline Approaches

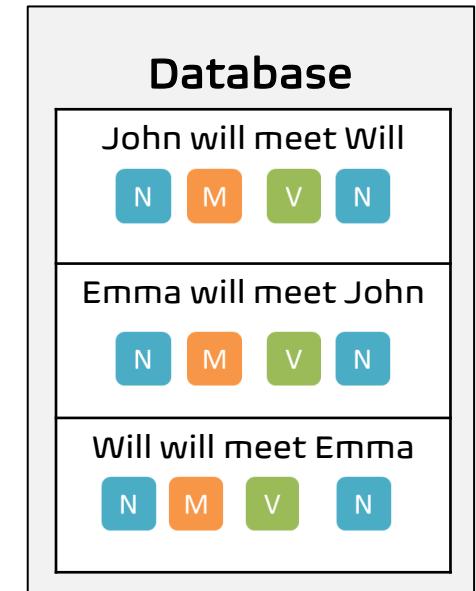
Part of Speech Tagging

Pick the largest number of the corresponding row

	N	V	M
John	2	0	0
meet	0	3	0
Will	2	0	3
Emma	2	0	0



Emma will meet Will



2 Baseline Approaches

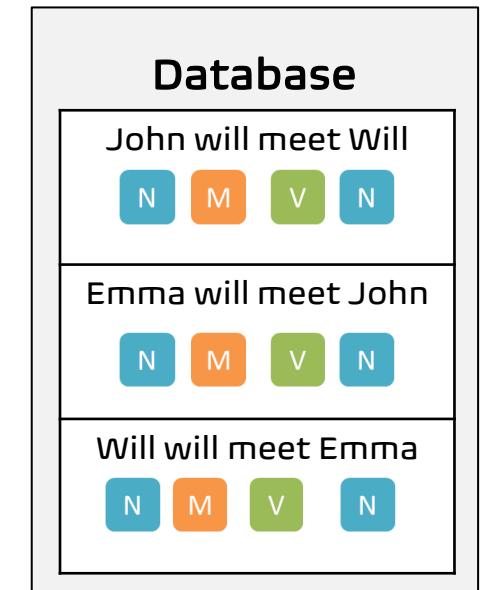
Part of Speech Tagging

Pick the largest number of the corresponding row

	N	V	M
John	2	0	0
meet	0	3	0
Will	2	0	3
Emma	2	0	0



Emma will meet Will



Better Solution? What about considering the Neighbors?

2 Baseline Approaches

Part of Speech Tagging: N-gram

A contiguous sequence of N items from a given sample of text

$N=1$

Emma will meet Will *unigram*

$N=2$

Emma will meet Will *bigram*

$N=3$

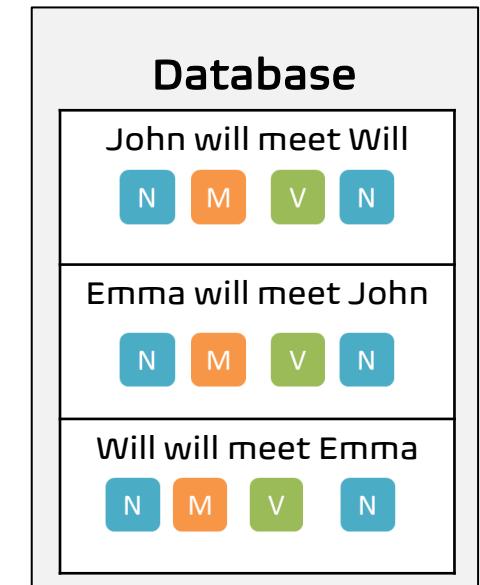
Emma will meet Will *trigram*

2 Baseline Approaches

Part of Speech Tagging: N-gram

	N - M	M - V	V - N
john-will	1	0	0
will-meet	0	3	0
meet-will	0	0	1
emma-will	1	0	0
meet-john	0	0	1
will-will	1	0	0
meet-emma	0	0	1

Emma will meet Will

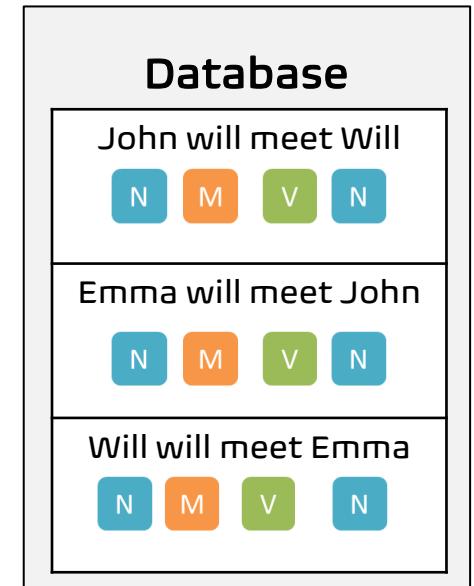


Part of Speech Tagging: N-gram

	N - M	M - V	V - N
john-will	1	0	0
will-meet	0	3	0
meet-will	0	0	1
emma-will	1	0	0
meet-john	0	0	1
will-will	1	0	0
meet-emma	0	0	1

N M V N

Emma will meet Will

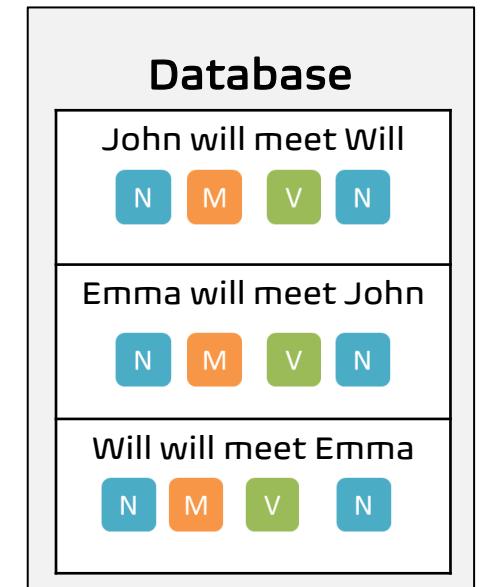


Part of Speech Tagging: N-gram

	N - M	M - V	V - N
john-will	1	0	0
will-meet	0	3	0
meet-will	0	0	1
emma-will	1	0	0
meet-john	0	0	1
will-will	1	0	0
meet-emma	0	0	1

N M V N

Emma will meet Will



What if we don't have in the database?

2 Baseline Approaches

Part of Speech Tagging: N-gram

	N - M	M - V	V - N
john-will	1	0	0
will-meet	0	3	0
meet-will	0	0	1
emma-will	1	0	0
meet-john	0	0	1
will-will	1	0	0
meet-emma	0	0	1



SORRY !!!
No Results Found

Emma will see Will

Database

John will meet Will
N M V N
Emma will meet John
N M V N
Will will meet Emma
N M V N

What if we don't have in the database?

0 LECTURE PLAN

Lecture 6: Part of Speech Tagging

1. Part-of-Speech Tagging
2. Baseline Approaches
 1. Rule-based Model
 2. Look-up Table Model
 3. N-Gram Model
3. Probabilistic Approaches
 1. Hidden Markov Model
 2. Conditional Random Field
4. Deep Learning Approaches

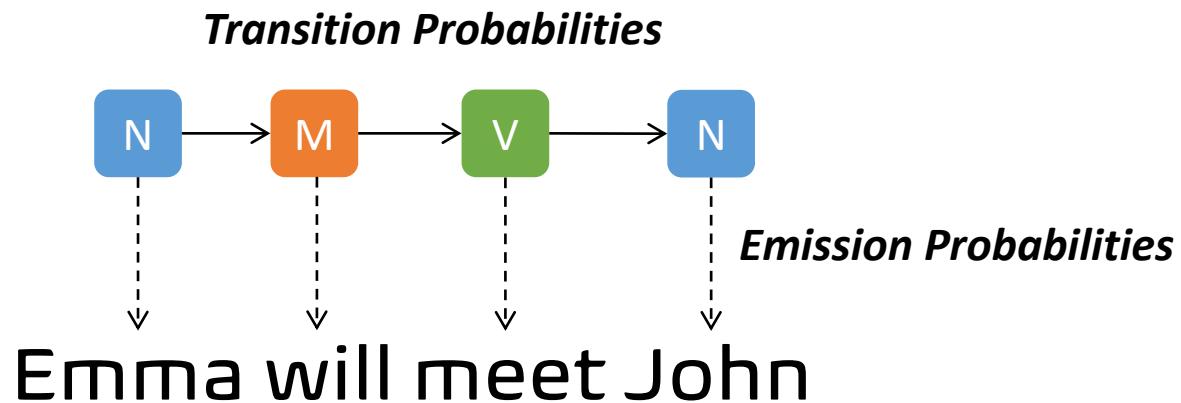
Hidden Markov Model: Idea



Emma will meet John

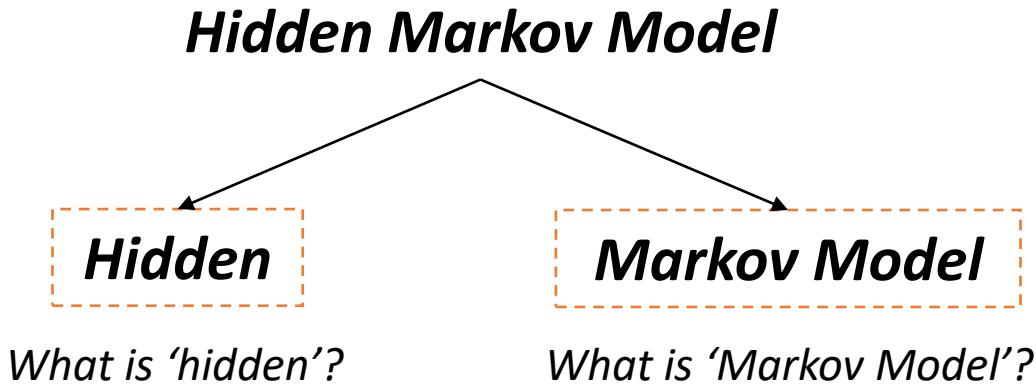
3 Probabilistic Approaches

Hidden Markov Model: Idea



3 Probabilistic Approaches

Hidden Markov Model (HMM)



3 Probabilistic Approaches

Markov Model



Andrei Andreyevich Markov

The purpose of introducing Markov Chain
*An example of statistical investigation in the text of
'Eugene Onyegin' illustrating coupling of 'tests' in chains.*

- A stochastic model used to model randomly changing system
- Has the Markov property if the ***conditional probability distribution of future states*** of the process ***depends only upon the present state***, not on the events that occurred before it.

3 Probabilistic Approaches

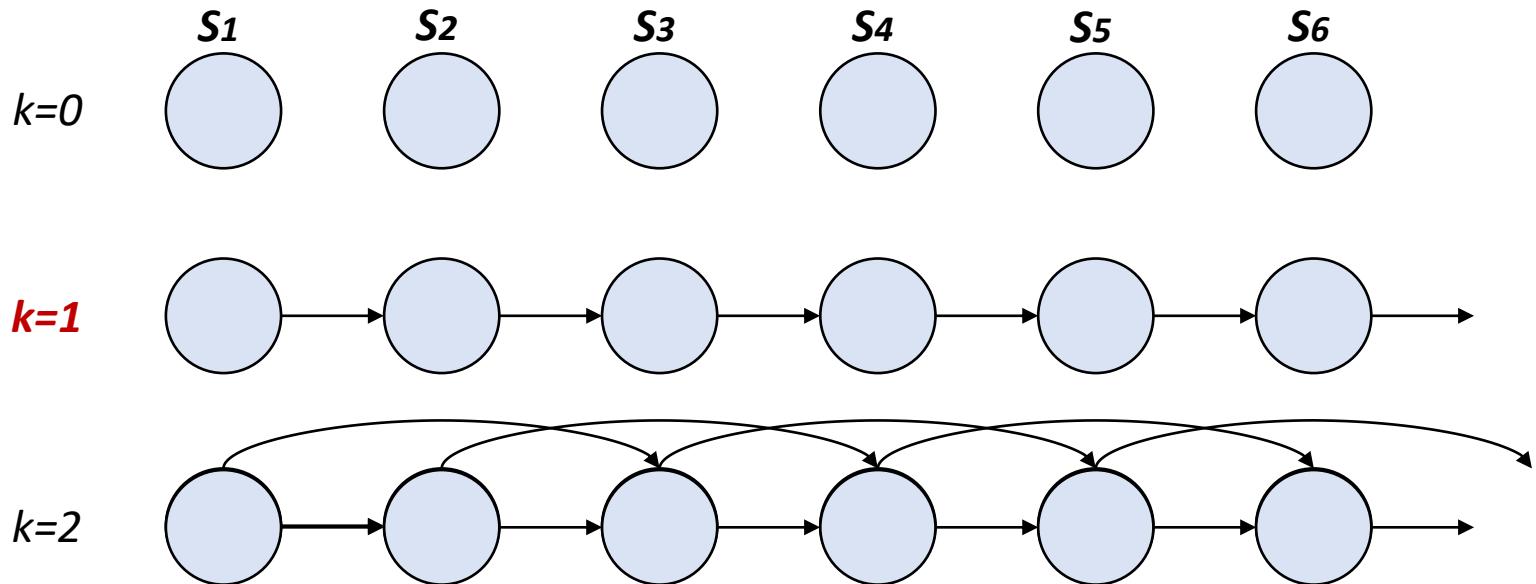
Markov Model (MM): K-Order Markov Property

- Assumption: last k states are sufficient
 - ($k=1$)** First-order Markov Process (*Most Commonly used*)

$$P(S_t | S_{t-1})$$

- ($k=2$)** Second-order Markov Process

$$P(S_t | S_{t-1}, S_{t-2})$$



3 Probabilistic Approaches

Markov Model (MM): Example

Let's predict tomorrow weather in Sydney. Assume we have three classes

Class 1: Rainy



Class 2: Cloudy



Class 3: Sunny



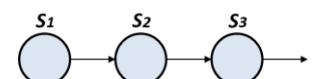
NOTE: Tomorrow weather depends only on today's!

First order Markov Model

We found the weather change pattern based on the 1-year data.

		Tomorrow		
		Rainy	Cloudy	Sunny
Today	Rainy	0.4	0.3	0.3
	Cloudy	0.2	0.6	0.2
	Sunny	0.1	0.1	0.8

$$S_{ij} = P(S_t = j | S_{t-1} = i)$$



3 Probabilistic Approaches

Markov Model (MM): Example

Let's predict tomorrow weather in Sydney. Assume we have three classes

Class 1: Rainy



Class 2: Cloudy



Class 3: Sunny



NOTE: Tomorrow weather depends only on today's!

First order Markov Model



If it is raining today,
how will be the weather
tomorrow?

We found the weather change pattern based on the 1-year data.

		Tomorrow		
		Rainy	Cloudy	Sunny
Today	Rainy	0.4	0.3	0.3
	Cloudy	0.2	0.6	0.2
	Sunny	0.1	0.1	0.8

$$S_{ij} = P(S_t = j | S_{t-1} = i)$$

$$S_{\text{rainyrainy}} = 0.4$$

$$S_{\text{rainycloudy}} = 0.3$$

$$S_{\text{rainysunny}} = 0.3$$

3 Probabilistic Approaches

Markov Model (MM): Example

Let's predict tomorrow weather in Sydney. Assume we have three classes

Class 1: Rainy



Class 2: Cloudy



Class 3: Sunny



NOTE: Tomorrow weather depends only on today's!

First order Markov Model



If it is raining today,
how will be the weather
tomorrow? **Rainy!**

We found the weather change pattern based on the 1-year data.

		Tomorrow		
		Rainy	Cloudy	Sunny
Today	Rainy	0.4	0.3	0.3
	Cloudy	0.2	0.6	0.2
	Sunny	0.1	0.1	0.8

$$S_{ij} = P(S_t = j | S_{t-1} = i)$$

$$S_{\text{rainyrainy}} = 0.4$$

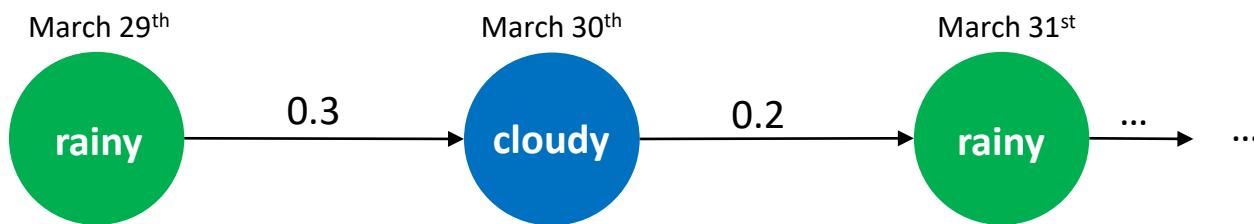
$$S_{\text{rainycloudy}} = 0.3$$

$$S_{\text{rainysunny}} = 0.3$$

3 Probabilistic Approaches

Markov Model (MM): Example

Transition Probabilities



We found the weather change pattern based on the 1-year data.

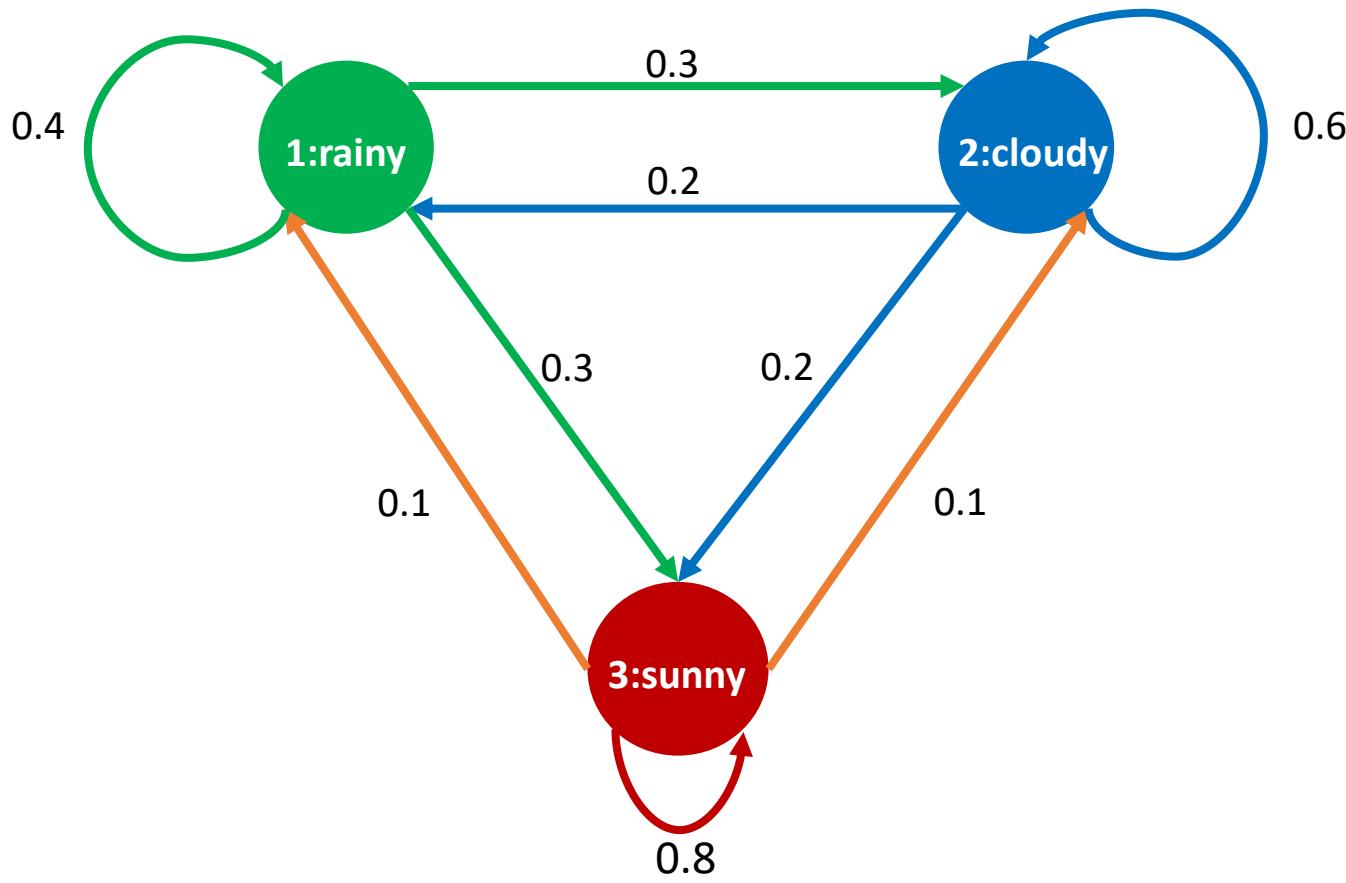
		Tomorrow		
		Rainy	Cloudy	Sunny
Today	Rainy	0.4	0.3	0.3
	Cloudy	0.2	0.6	0.2
	Sunny	0.1	0.1	0.8

Markov Model (MM): Example

Visual illustration with diagram

- Each state corresponds to one observation
- Sum of outgoing edge weights is one

		Tomorrow		
		Rainy	Cloudy	Sunny
Today	Rainy	0.4	0.3	0.3
	Cloudy	0.2	0.6	0.2
	Sunny	0.1	0.1	0.8



3 Probabilistic Approaches

Markov Model (MM): Example

State Transition Matrix

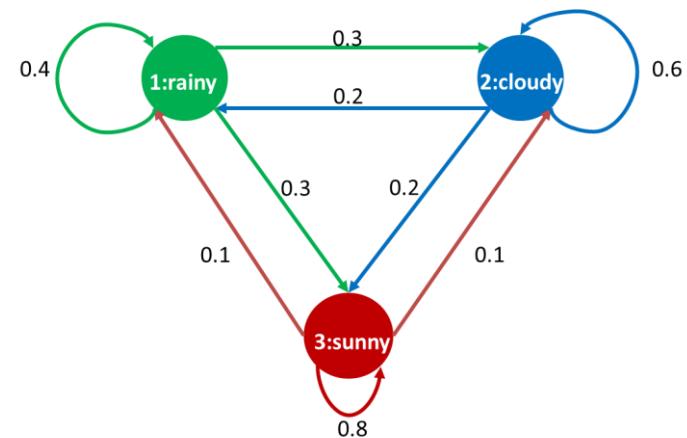
$$S_{ij} = P(S_t = j | S_{t-1} = i) \quad 1 \leq i, j \leq N$$

$$S_{ij} \geq 0$$

$$S = \begin{bmatrix} S_{11} & S_{12} & \dots & S_{1N} \\ S_{21} & S_{22} & \dots & S_{2N} \\ S_{31} & S_{32} & \dots & S_{3N} \\ \vdots & \vdots & \vdots & \vdots \\ S_{N1} & S_{N2} & \dots & S_{NN} \end{bmatrix}$$

		Tomorrow		
		Rainy	Cloudy	Sunny
Today	Rainy	0.4	0.3	0.3
	Cloudy	0.2	0.6	0.2
	Sunny	0.1	0.1	0.8

		Time $t+1$		
		S1	S2	S3
Time t	S1	0.4	0.3	0.3
	S2	0.2	0.6	0.2
	S3	0.1	0.1	0.8



3 Probabilistic Approaches

Markov Model (MM): Example

Sequence Probability

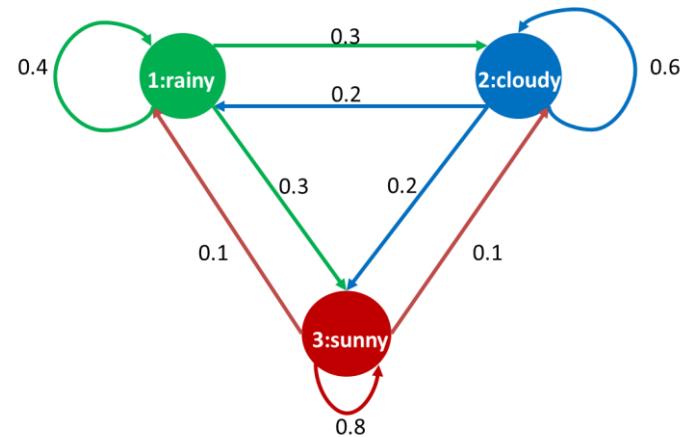
Q: What is the probability that the weather for the next 7 days will be “sun-sun-rain-rain-sun-cloudy-sun” if it is sunny today?

$$P(S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3 \mid \text{model})$$

$$= P(S_3) \cdot P(S_3 \mid S_3) \cdot P(S_3 \mid S_3) \cdot P(S_1 \mid S_3) \cdot P(S_1 \mid S_1) P(S_3 \mid S_1) P(S_2 \mid S_3) P(S_3 \mid S_2)$$

$$= 1 \cdot (0.8)(0.8)(0.1)(0.4)(0.3)(0.1)(0.2)$$

$$= 1.536 \times 10^{-4}$$



$$S_{ij} = P(S_t = j \mid S_{t-1} = i)$$

3 Probabilistic Approaches

Hidden Markov Model (HMM)

Hidden Markov Model

Hidden

What is ‘hidden’?

Markov Model

What is ‘Markov Model’?

oy you there wakeup!



ARE YOU STILL SLEEPING?

3 Probabilistic Approaches

Hidden Markov Model (HMM)

Hidden Markov Model

Hidden

What is 'hidden'?

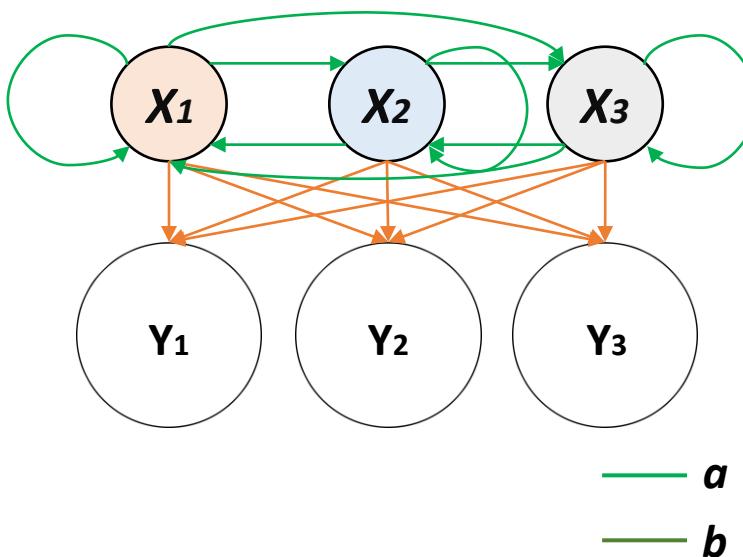
Markov Model

What is 'Markov Model'?

3 Probabilistic Approaches

Hidden Markov Model (HMM)

Hidden Markov Models (HMMs) are a class of probabilistic graphical model that allow us to ***predict a sequence of unknown (hidden) variables*** from a set of observed variables.



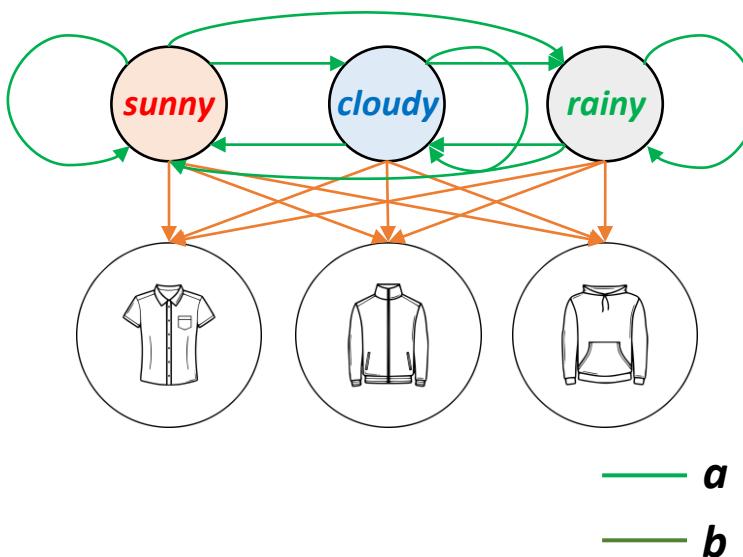
x *hidden states* ←
 y possible observations
 a state transition probabilities
 b output probabilities

- States are ***hidden***
- ***Observable outcome*** linked to states
- Each state has ***observation probabilities*** to determine the observable event

3 Probabilistic Approaches

Hidden Markov Model (HMM)

Hidden Markov Models (HMMs) are a class of probabilistic graphical model that allow us to ***predict a sequence of unknown (hidden) variables*** from a set of observed variables.



x *hidden states* ←
 y possible observations
 a state transition probabilities
 b output probabilities

- States are **hidden**
- **Observable outcome** linked to states
- Each state has **observation probabilities** to determine the observable event

3 Probabilistic Approaches

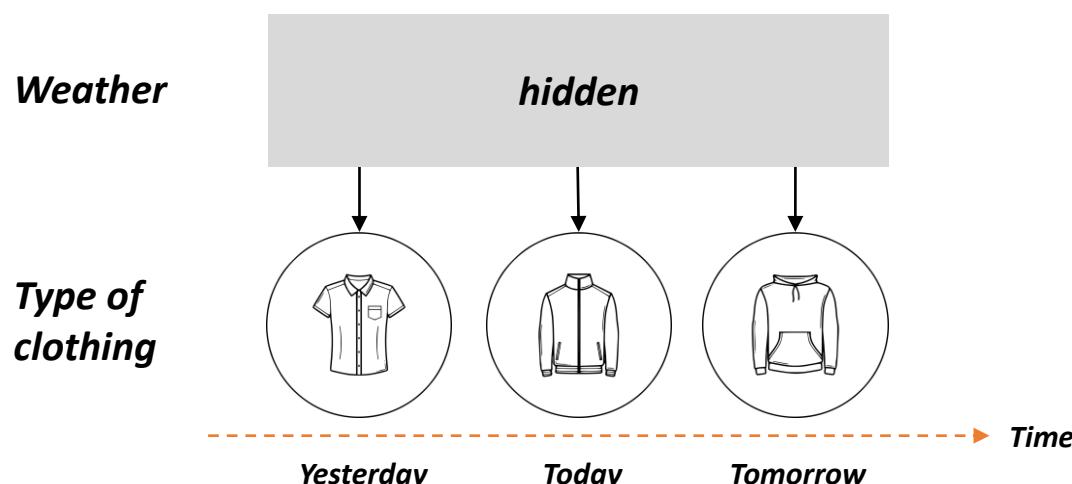
Hidden Markov Model (HMM)

Predicting the **weather (state: hidden variable)** based on the type of **clothes that the person wears (observed event)**

- **Weather (hidden variable): sunny, cloudy, rainy**
- **Observed variables are the type of clothing the person worn**

The arrows represent:

- *Transition Probabilities: from a hidden state to another hidden state*
- *Emission Probabilities: from a hidden state to an observed variable*

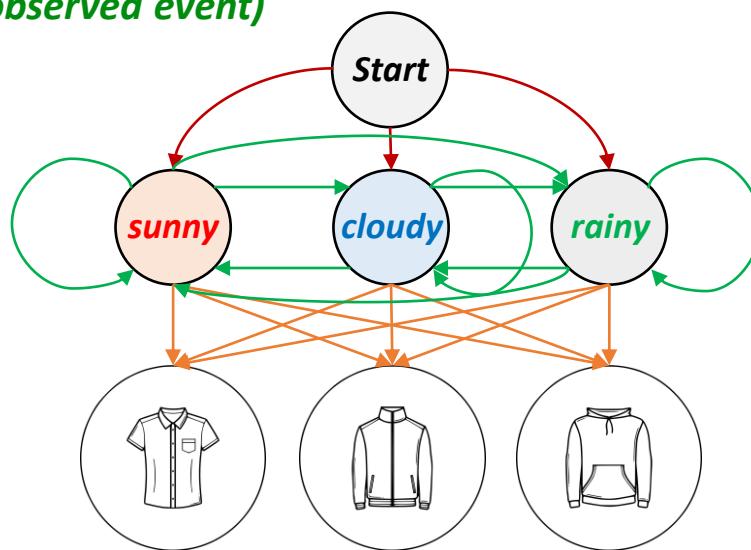


One or more observations allow us to make an inference about a sequence of hidden states

3 Probabilistic Approaches

Hidden Markov Model (HMM)

Predicting the **weather (state: hidden variable)** based on the type of **clothes that the person wears (observed event)**



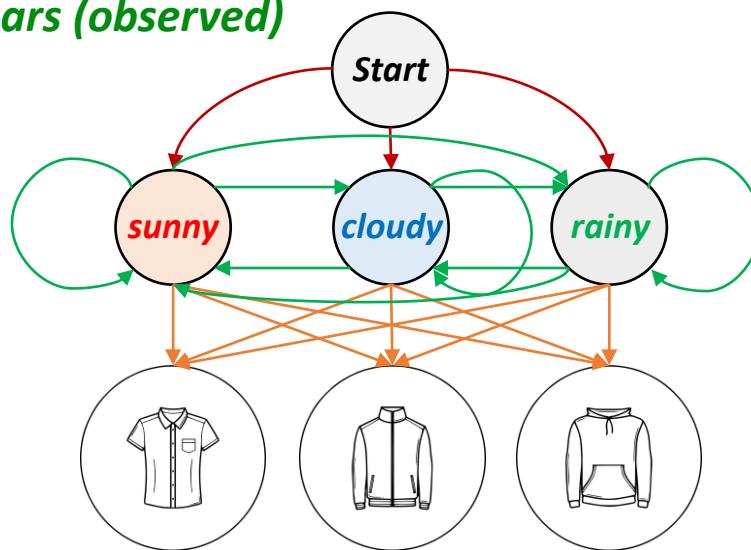
In order to compute the joint probability of a sequence of hidden states, we need to assemble three types of information:

1. **Initial state information (a.k.a. prior probability)** - The initial probability of transitioning to a hidden state.
2. **Transition probabilities** — the probability of transitioning to a new state conditioned on a present state
3. **Emission probabilities** – the probability of transitioning to an observed state conditioned on a hidden state

3 Probabilistic Approaches

Hidden Markov Model (HMM)

Predicting the **weather (hidden variable)** based on the type of **clothes that someone wears (observed)**



Priors

Rainy	0.6
Cloudy	0.3
Sunny	0.1

Transitions

		Tomorrow		
		Rainy	Cloudy	Sunny
Today	Rainy	0.6	0.3	0.1
	Cloudy	0.4	0.3	0.3
	Sunny	0.1	0.4	0.5

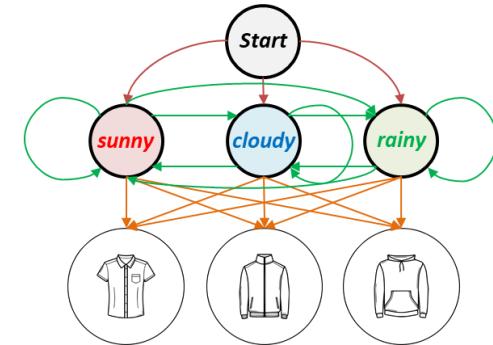
Emissions

	Shirts	Jacket	Hoodies
Rainy	0.8	0.19	0.01
Cloudy	0.5	0.4	0.1
Sunny	0.01	0.2	0.79

3 Probabilistic Approaches

Hidden Markov Model (HMM)

We had the list of clothes that Caren wears for three days



Firstly, just calculate the weather condition 'cloudy-cloudy-sunny' (Random Selection)

1. Calculate the probability that Caren could wear that clothing (with the weather condition 'cloudy – cloudy – sunny')

$$P(\text{shirts}/\text{cloudy}) * P(\text{hoodie}/\text{cloudy}) * P(\text{hoodie}/\text{sunny})$$

2. Calculate the probability that weathers were 'cloudy – cloudy – sunny'

$$P(\text{prior_cloudy}) * P(\text{cloudy}/\text{cloudy}) * P(\text{sunny}/\text{cloudy})$$

$$P(\text{shirts}/\text{cloudy}) * P(\text{hoodie}/\text{cloudy}) * P(\text{hoodie}/\text{sunny}) * P(\text{prior_cloudy}) * P(\text{cloudy}/\text{cloudy}) * P(\text{sunny}/\text{cloudy})$$

This is the probability when we assume the weather (cloudy – cloudy – sunny)

Hidden Markov Model (HMM)

The previous was the only probability when we assume the weather (cloudy – cloudy – sunny)

This is a complete set of $3^3=27$ cases of weather states for three days:

$\{x_1=s_1=\text{sunny}, x_2=s_1=\text{sunny}, x_3=s_1=\text{sunny}\}$, $\{x_1=s_1=\text{sunny}, x_2=s_1=\text{sunny}, x_3=s_2=\text{cloudy}\}$,
 $\{x_1=s_1=\text{sunny}, x_2=s_1=\text{sunny}, x_3=s_3=\text{rainy}\}$, $\{x_1=s_1=\text{sunny}, x_2=s_2=\text{cloudy}, x_3=s_1=\text{sunny}\}$,
 $\{x_1=s_1=\text{sunny}, x_2=s_2=\text{cloudy}, x_3=s_2=\text{cloudy}\}$, $\{x_1=s_1=\text{sunny}, x_2=s_2=\text{cloudy}, x_3=s_3=\text{rainy}\}$,
 $\{x_1=s_1=\text{sunny}, x_2=s_3=\text{rainy}, x_3=s_1=\text{sunny}\}$, $\{x_1=s_1=\text{sunny}, x_2=s_3=\text{rainy}, x_3=s_2=\text{cloudy}\}$,
 $\{x_1=s_1=\text{sunny}, x_2=s_3=\text{rainy}, x_3=s_3=\text{rainy}\}$, $\{x_1=s_2=\text{cloudy}, x_2=s_1=\text{sunny}, x_3=s_1=\text{sunny}\}$,
 $\{x_1=s_2=\text{cloudy}, x_2=s_1=\text{sunny}, x_3=s_2=\text{cloudy}\}$, $\{x_1=s_2=\text{cloudy}, x_2=s_1=\text{sunny}, x_3=s_3=\text{rainy}\}$,
 $\{x_1=s_2=\text{cloudy}, x_2=s_2=\text{cloudy}, x_3=s_1=\text{sunny}\}$, $\{x_1=s_2=\text{cloudy}, x_2=s_2=\text{cloudy}, x_3=s_2=\text{cloudy}\}$,
 $\{x_1=s_2=\text{cloudy}, x_2=s_2=\text{cloudy}, x_3=s_3=\text{rainy}\}$, $\{x_1=s_2=\text{cloudy}, x_2=s_3=\text{rainy}, x_3=s_1=\text{sunny}\}$,
 $\{x_1=s_2=\text{cloudy}, x_2=s_3=\text{rainy}, x_3=s_2=\text{cloudy}\}$, $\{x_1=s_2=\text{cloudy}, x_2=s_3=\text{rainy}, x_3=s_3=\text{rainy}\}$,
 $\{x_1=s_3=\text{rainy}, x_2=s_1=\text{sunny}, x_3=s_1=\text{sunny}\}$, $\{x_1=s_3=\text{rainy}, x_2=s_1=\text{sunny}, x_3=s_2=\text{cloudy}\}$,
 $\{x_1=s_3=\text{rainy}, x_2=s_1=\text{sunny}, x_3=s_3=\text{rainy}\}$, $\{x_1=s_3=\text{rainy}, x_2=s_2=\text{cloudy}, x_3=s_1=\text{sunny}\}$,
 $\{x_1=s_3=\text{rainy}, x_2=s_2=\text{cloudy}, x_3=s_2=\text{cloudy}\}$, $\{x_1=s_3=\text{rainy}, x_2=s_2=\text{cloudy}, x_3=s_3=\text{rainy}\}$,
 $\{x_1=s_3=\text{rainy}, x_2=s_3=\text{rainy}, x_3=s_1=\text{sunny}\}$, $\{x_1=s_3=\text{rainy}, x_2=s_3=\text{rainy}, x_3=s_2=\text{cloudy}\}$,
 $\{x_1=s_3=\text{rainy}, x_2=s_3=\text{rainy}, x_3=s_3=\text{rainy}\}$.

Easy but slow solution: Exhaustive enumeration!

3 Probabilistic Approaches

Hidden Markov Model (HMM): Evaluation

Do we need to calculate this much all the time?

This is a complete set of $3^3=27$ cases of weather states for three days:

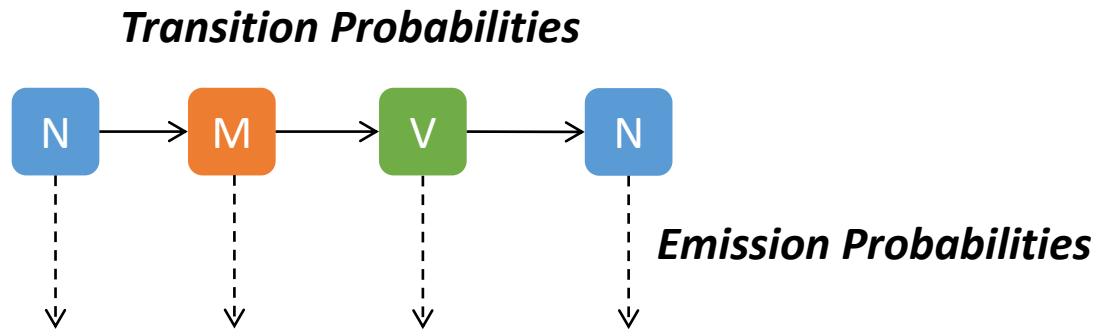
$\{x_1=s_1=\text{sunny}, x_2=s_1=\text{sunny}, x_3=s_1=\text{sunny}\}$, $\{x_1=s_1=\text{sunny}, x_2=s_1=\text{sunny}, x_3=s_2=\text{cloudy}\}$,
 $\{x_1=s_1=\text{sunny}, x_2=s_1=\text{sunny}, x_3=s_3=\text{rainy}\}$, $\{x_1=s_1=\text{sunny}, x_2=s_2=\text{cloudy}, x_3=s_1=\text{sunny}\}$,
 $\{x_1=s_1=\text{sunny}, x_2=s_2=\text{cloudy}, x_3=s_2=\text{cloudy}\}$, $\{x_1=s_1=\text{sunny}, x_2=s_2=\text{cloudy}, x_3=s_3=\text{rainy}\}$,
 $\{x_1=s_1=\text{sunny}, x_2=s_3=\text{rainy}, x_3=s_1=\text{sunny}\}$, $\{x_1=s_1=\text{sunny}, x_2=s_3=\text{rainy}, x_3=s_2=\text{cloudy}\}$,
 $\{x_1=s_1=\text{sunny}, x_2=s_3=\text{rainy}, x_3=s_3=\text{rainy}\}$, $\{x_1=s_2=\text{cloudy}, x_2=s_1=\text{sunny}, x_3=s_1=\text{sunny}\}$,
 $\{x_1=s_2=\text{cloudy}, x_2=s_1=\text{sunny}, x_3=s_2=\text{cloudy}\}$, $\{x_1=s_2=\text{cloudy}, x_2=s_2=\text{cloudy}, x_3=s_1=\text{sunny}\}$,
 $\{x_1=s_2=\text{cloudy}, x_2=s_2=\text{cloudy}, x_3=s_2=\text{cloudy}\}$, $\{x_1=s_2=\text{cloudy}, x_2=s_3=\text{rainy}, x_3=s_1=\text{sunny}\}$,
 $\{x_1=s_2=\text{cloudy}, x_2=s_3=\text{rainy}, x_3=s_2=\text{cloudy}\}$, $\{x_1=s_2=\text{cloudy}, x_2=s_3=\text{rainy}, x_3=s_3=\text{rainy}\}$,
 $\{x_1=s_3=\text{rainy}, x_2=s_1=\text{sunny}, x_3=s_1=\text{sunny}\}$, $\{x_1=s_3=\text{rainy}, x_2=s_1=\text{sunny}, x_3=s_2=\text{cloudy}\}$,
 $\{x_1=s_3=\text{rainy}, x_2=s_1=\text{sunny}, x_3=s_3=\text{rainy}\}$, $\{x_1=s_3=\text{rainy}, x_2=s_2=\text{cloudy}, x_3=s_1=\text{sunny}\}$,
 $\{x_1=s_3=\text{rainy}, x_2=s_2=\text{cloudy}, x_3=s_2=\text{cloudy}\}$, $\{x_1=s_3=\text{rainy}, x_2=s_2=\text{cloudy}, x_3=s_3=\text{rainy}\}$,
 $\{x_1=s_3=\text{rainy}, x_2=s_3=\text{rainy}, x_3=s_1=\text{sunny}\}$, $\{x_1=s_3=\text{rainy}, x_2=s_3=\text{rainy}, x_3=s_2=\text{cloudy}\}$,
 $\{x_1=s_3=\text{rainy}, x_2=s_3=\text{rainy}, x_3=s_3=\text{rainy}\}$.

Any efficient solution?

Yes! Dynamic Programming technique

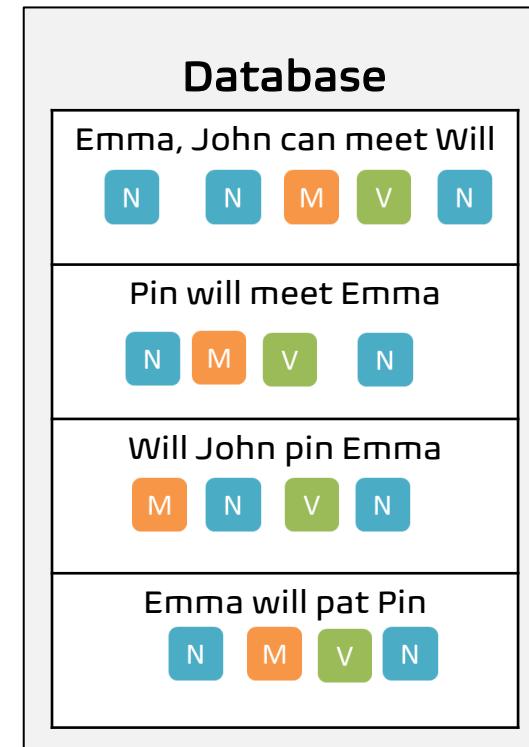
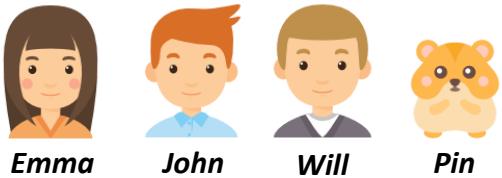
3 Probabilistic Approaches

Now, Let's Apply this HMM
to the Part of Speech Tagging Task!

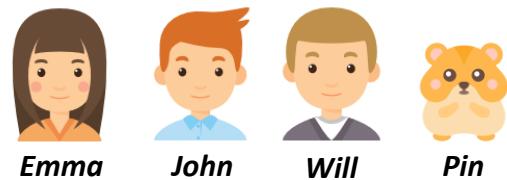


3 Probabilistic Approaches

Part of Speech Tagging: with HMM



3 Probabilistic Approaches



	N	V	M
Emma	4	0	0
John	2	0	0
Will	1	0	3
Pin	2	1	0
Can	0	0	1
Meet	0	2	0
Pat	0	1	0

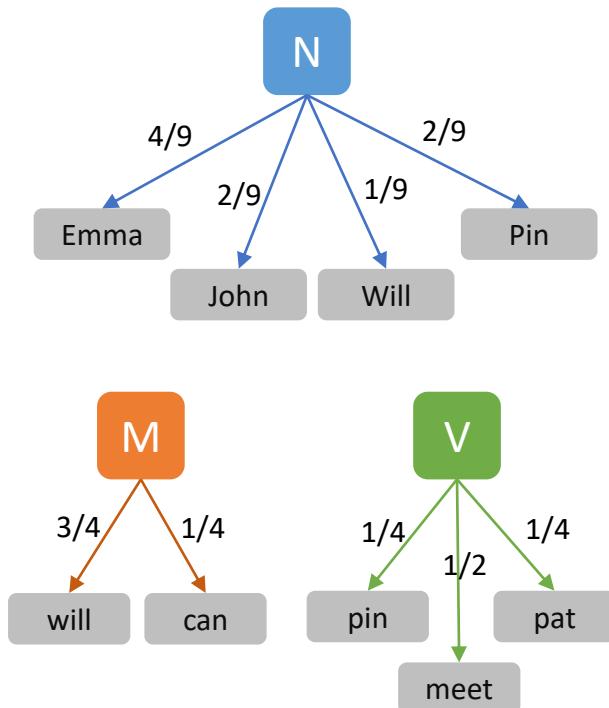
Database

Emma, John can meet Will
    
Pin will meet Emma
   
Will John pin Emma
   
Emma will pat Pin
   

Part of Speech Tagging: with HMM

Emission Probabilities

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

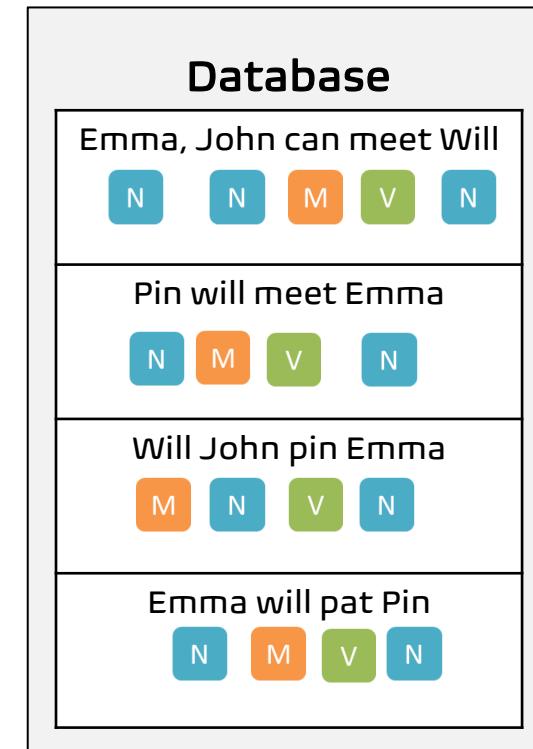


3 Probabilistic Approaches

Part of Speech Tagging: with HMM

Transition Probabilities

	N	V	M	<E>
<S>	3	0	1	0
N	1	1	3	4
V	4	0	0	0
M	1	3	0	0

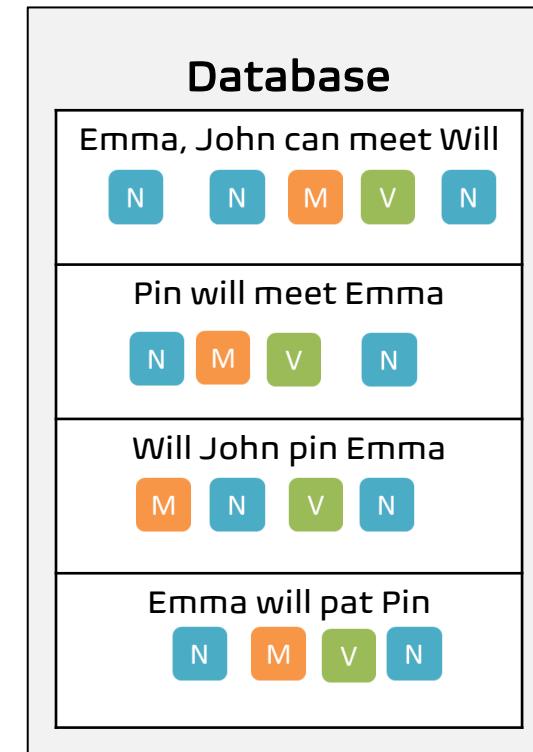


3 Probabilistic Approaches

Part of Speech Tagging: with HMM

Transition Probabilities

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0

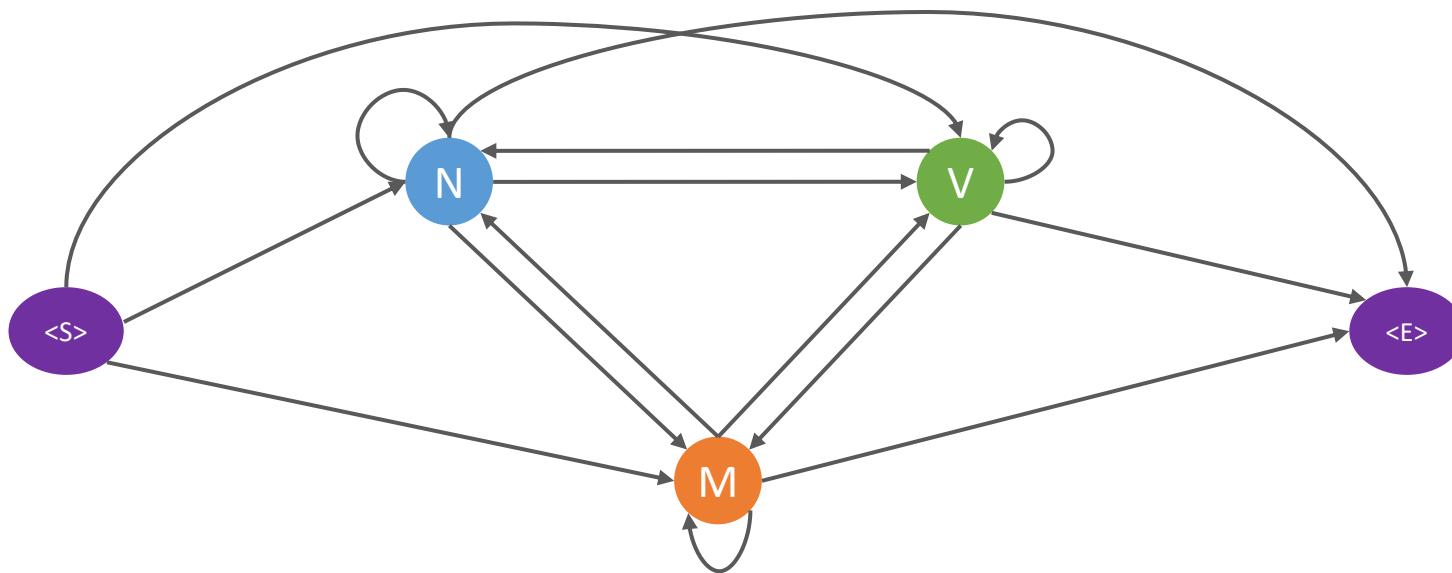


3 Probabilistic Approaches

Part of Speech Tagging: with HMM

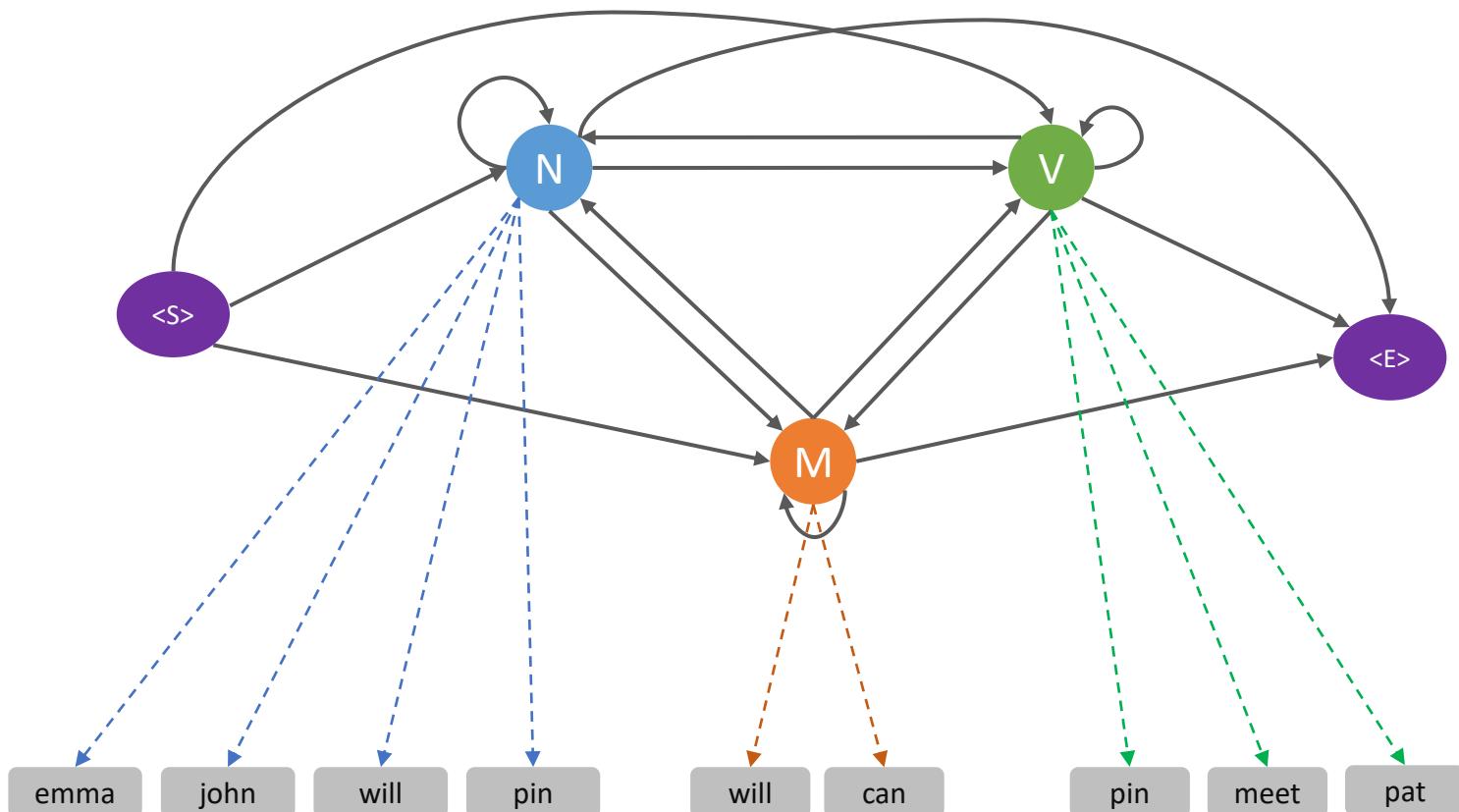
	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0

Transition Probabilities

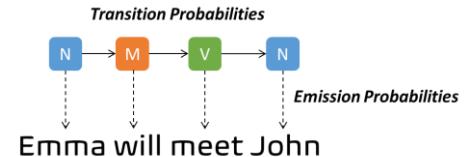


Part of Speech Tagging: with HMM

Let's combine this with emission probabilities!



3 Probabilistic Approaches

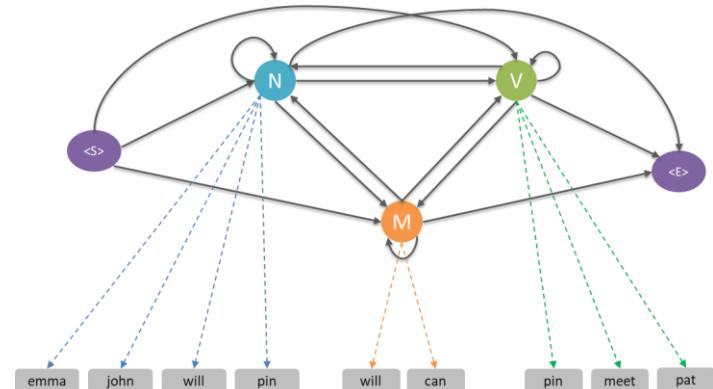


Part of Speech Tagging: with HMM

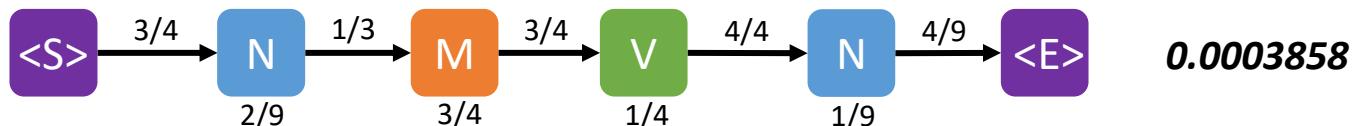
Let's combine this!

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0

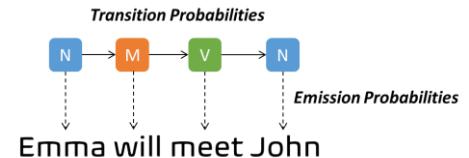


John will Pin Will



3

Probabilistic Approaches

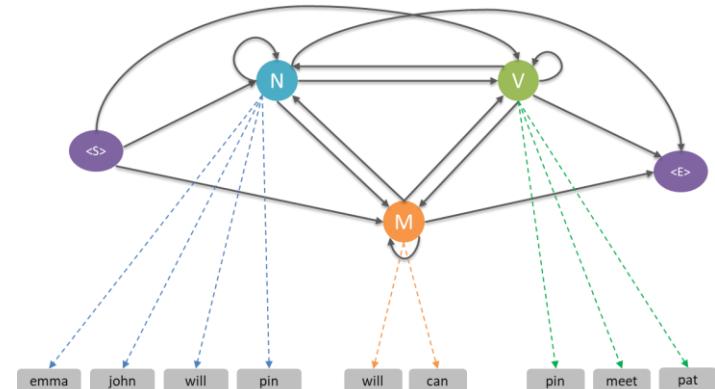


Part of Speech Tagging: with HMM

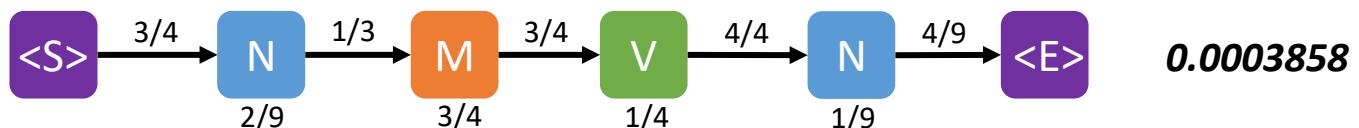
Let's combine this!

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

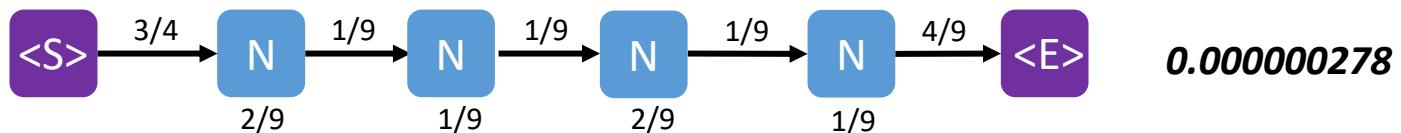
	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



John will Pin Will



John will Pin Will



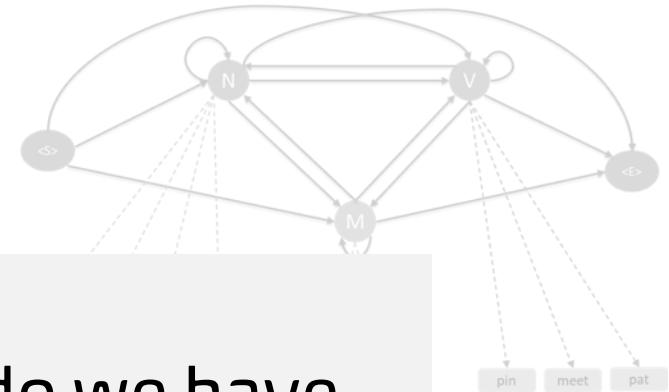
3 Probabilistic Approaches

Part of Speech Tagging: with HMM

Let's combine this!

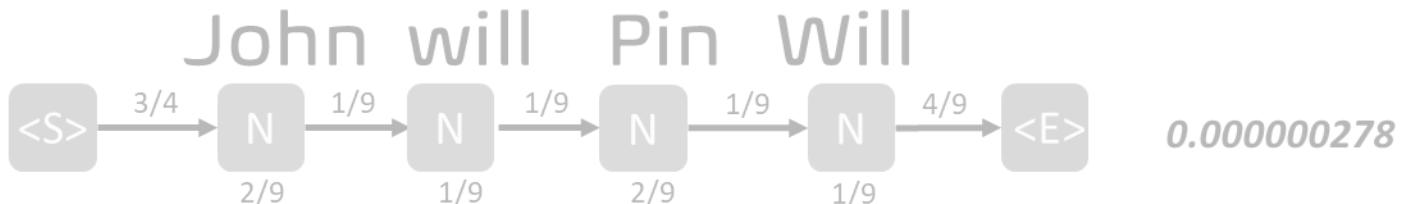
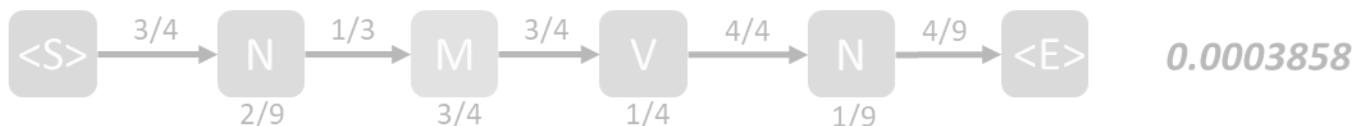
	N	V	M	
Emma	4/9	0	0	
John	2/9	0	0	
Will	1/9	0	3/4	
Pin	2/9	1/4	0	
Can				
Meet				
Pat				

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	1/4	0	0	0



Question!

How many possibilities do we have for the sentence 'John will Pin Will'?



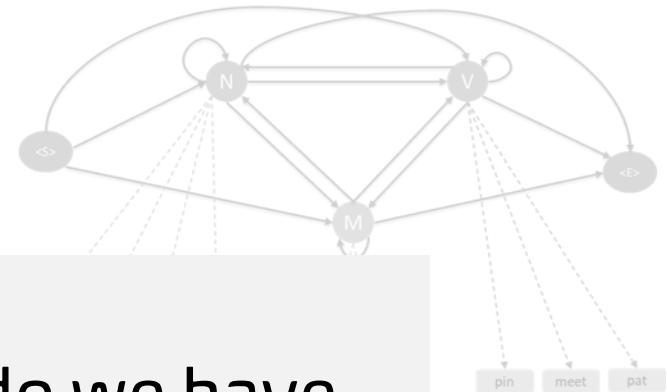
3 Probabilistic Approaches

Part of Speech Tagging: with HMM

Let's combine this!

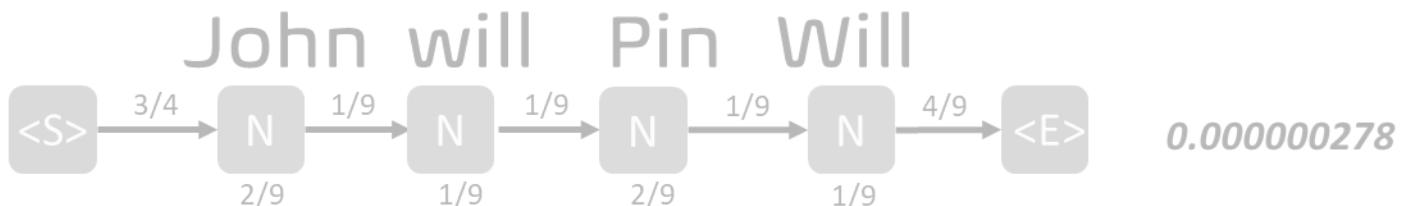
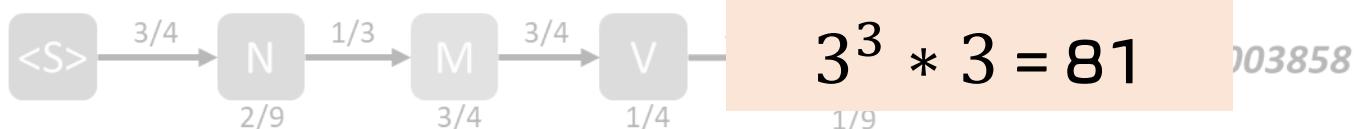
	N	V	M	
Emma	4/9	0	0	
John	2/9	0	0	
Will	1/9	0	3/4	
Pin	2/9	1/4	0	
Can				
Meet				
Pat				

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	1/4	0	0	0



Question!

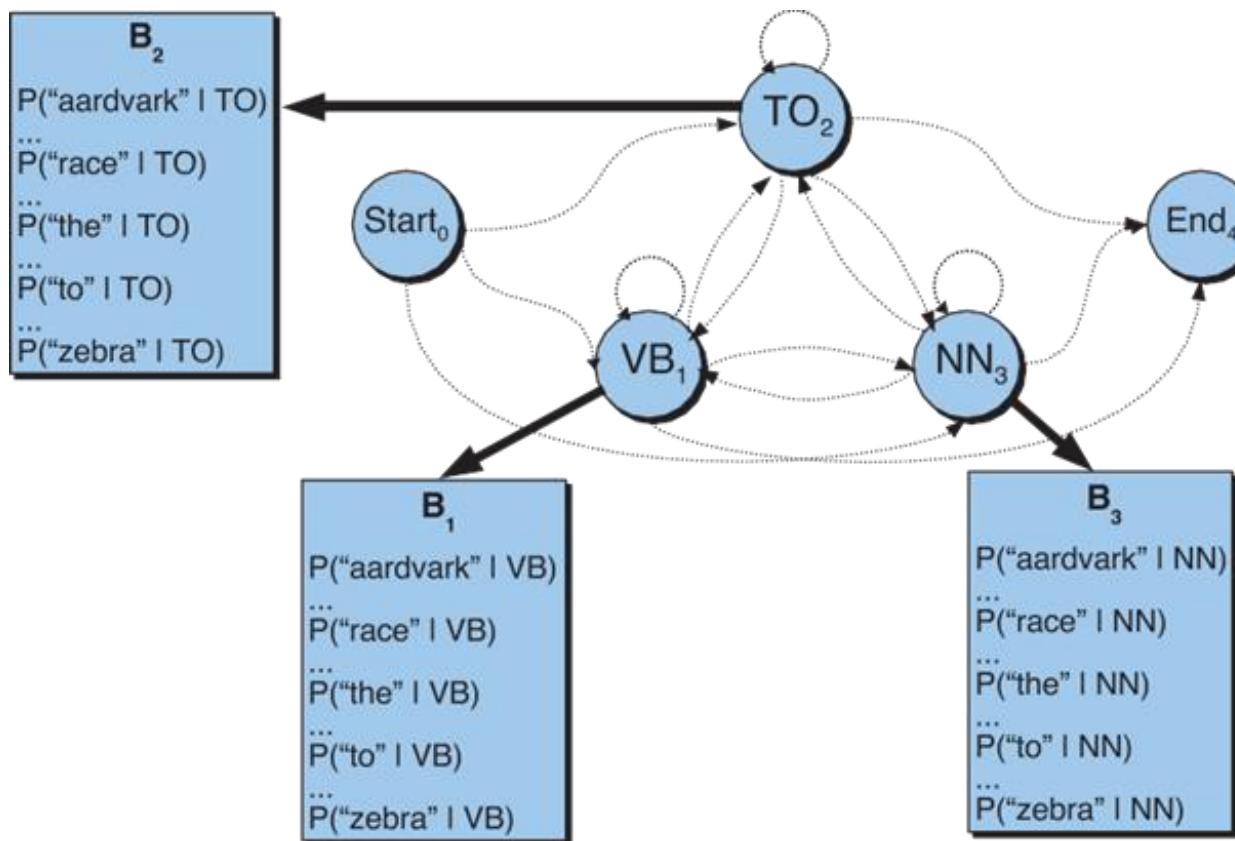
How many possibilities do we have
for the sentence 'John will Pin Will'?



3 Probabilistic Approaches

Hidden Markov Model (HMM) with POS Tagging

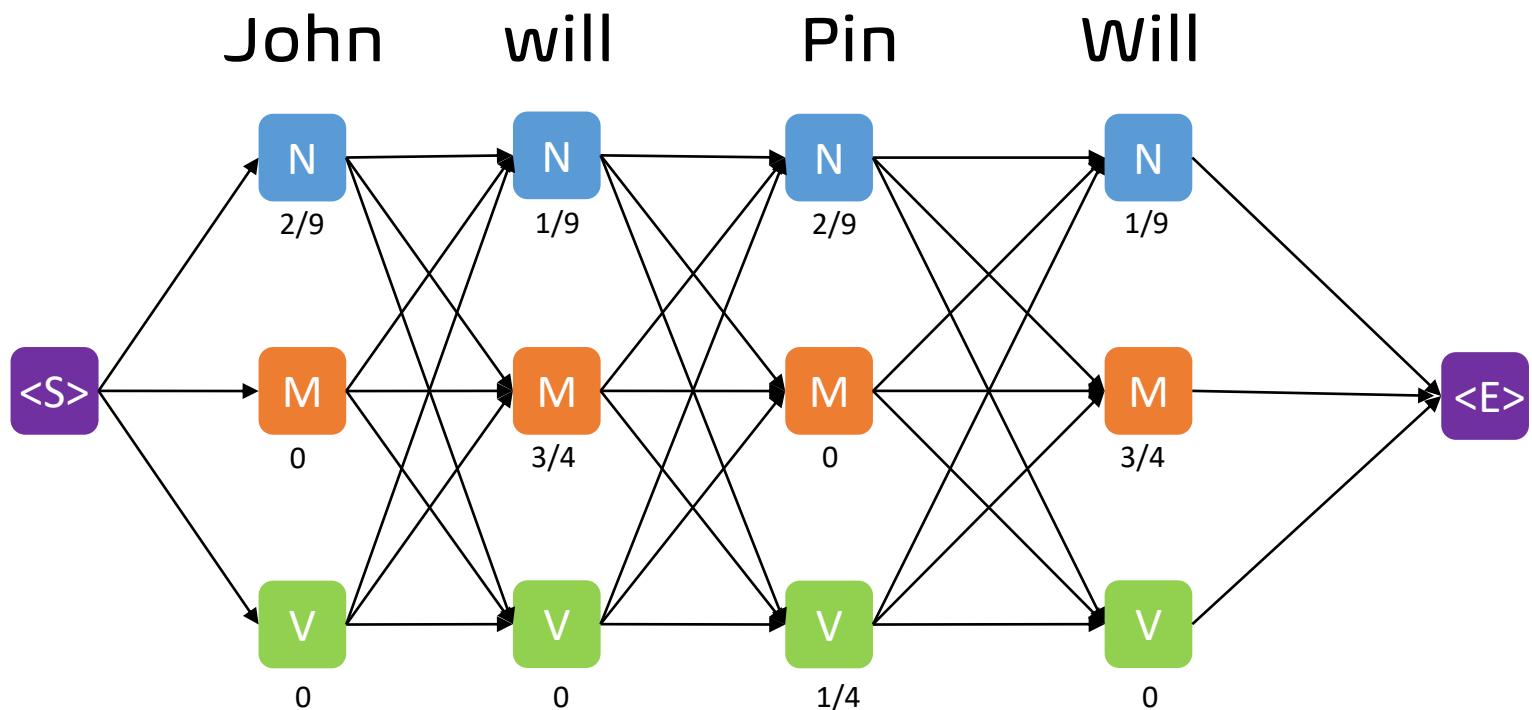
Emissions – Real World Example



3 Probabilistic Approaches

POS Tagging: with HMM

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

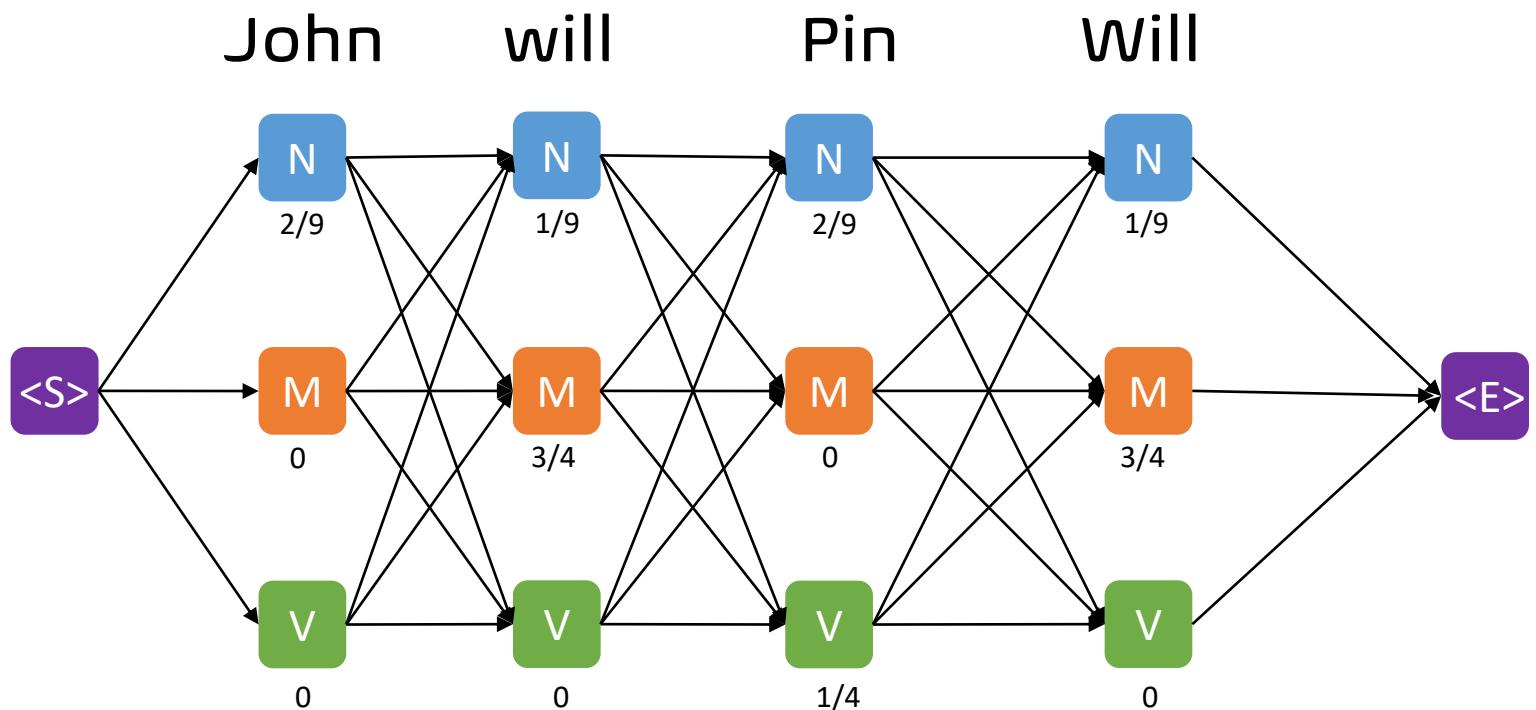


3 Probabilistic Approaches

POS Tagging: with HMM

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



Beam Search

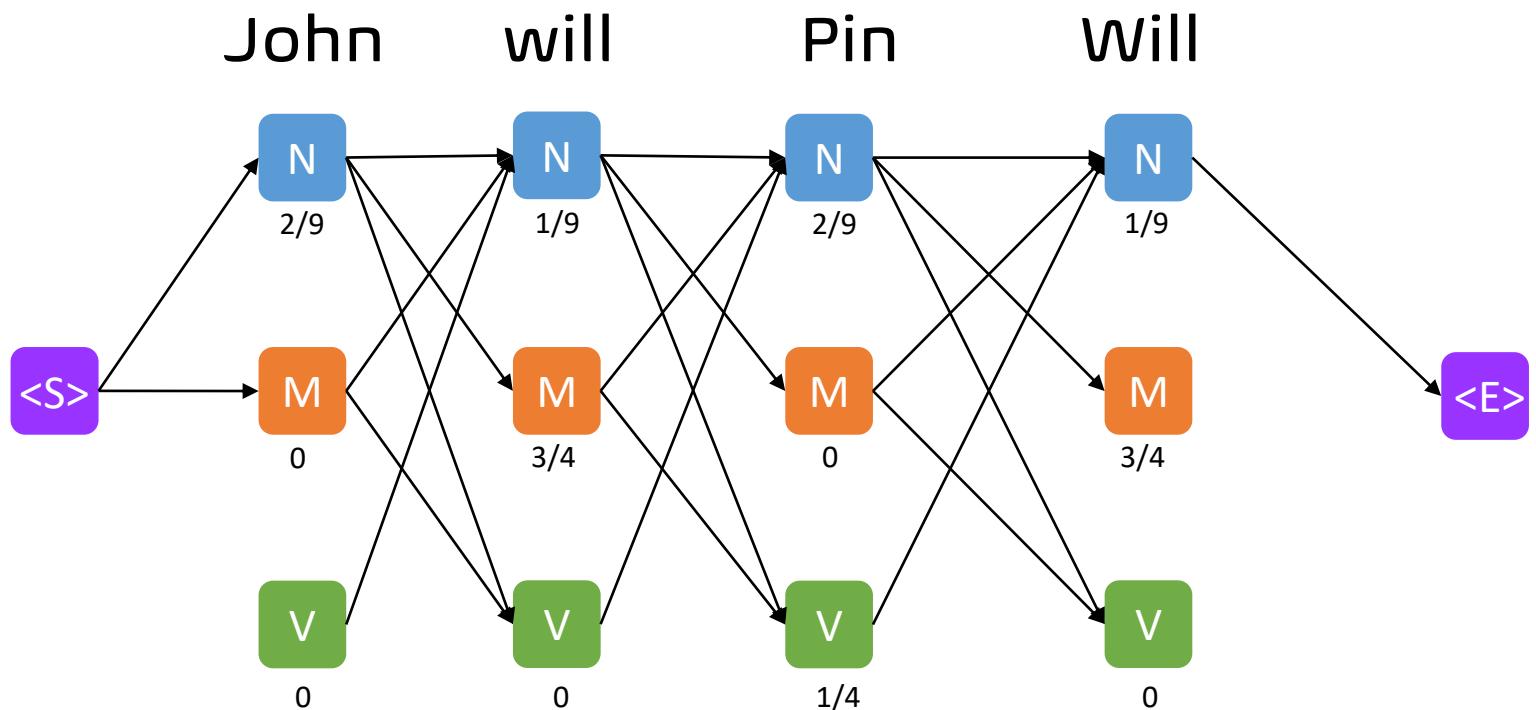
Get rid of unlikely candidates

3 Probabilistic Approaches

POS Tagging: with HMM

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



Beam Search

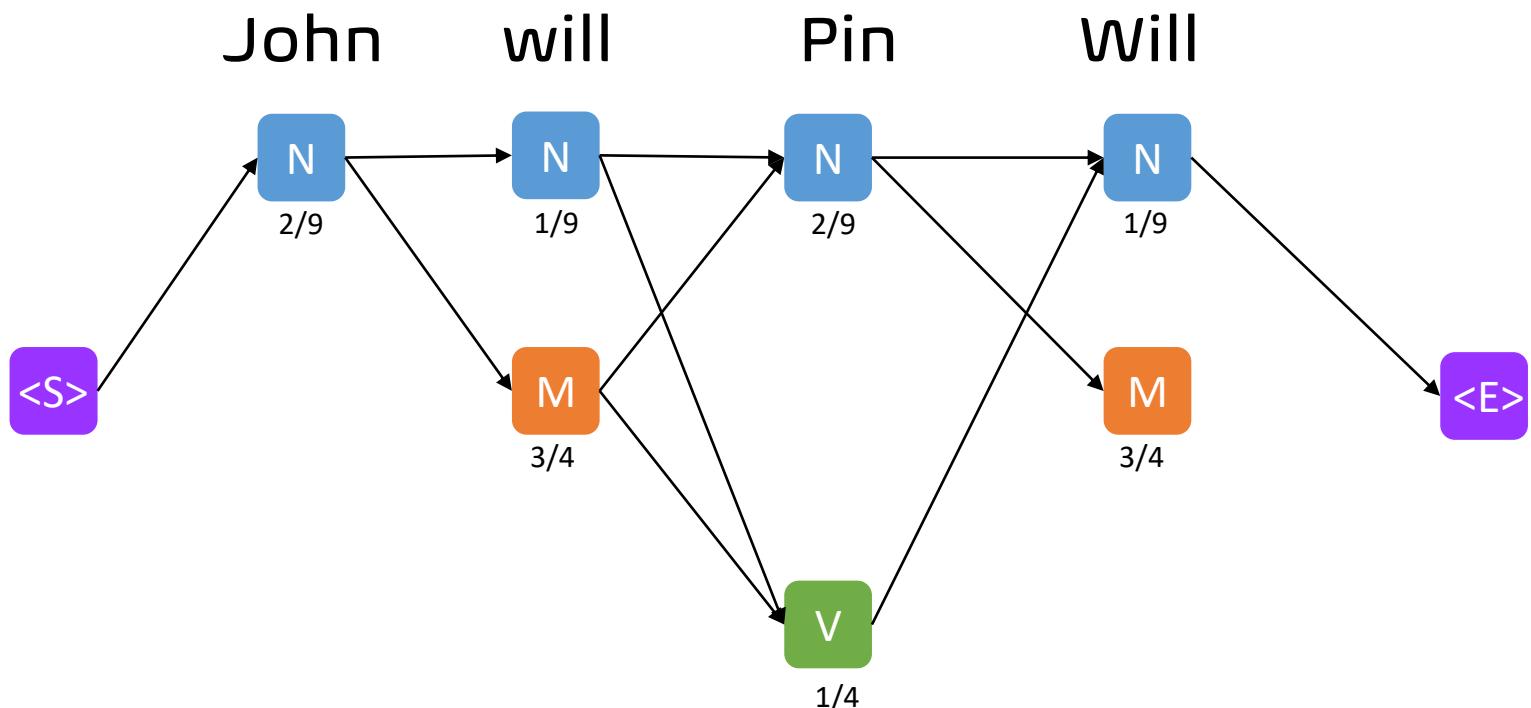
Get rid of unlikely candidates

3 Probabilistic Approaches

POS Tagging: with HMM

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



Beam Search

Get rid of unlikely candidates

3 Probabilistic Approaches

POS Tagging: with HMM

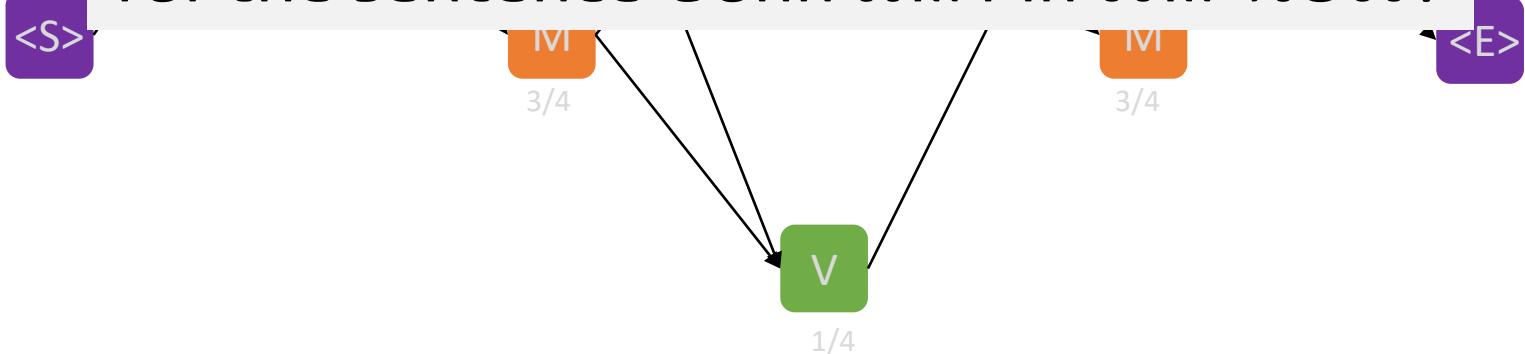
	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0

John will Pin will

Question!

How many possibilities do we have
for the sentence 'John will Pin will' NOW?

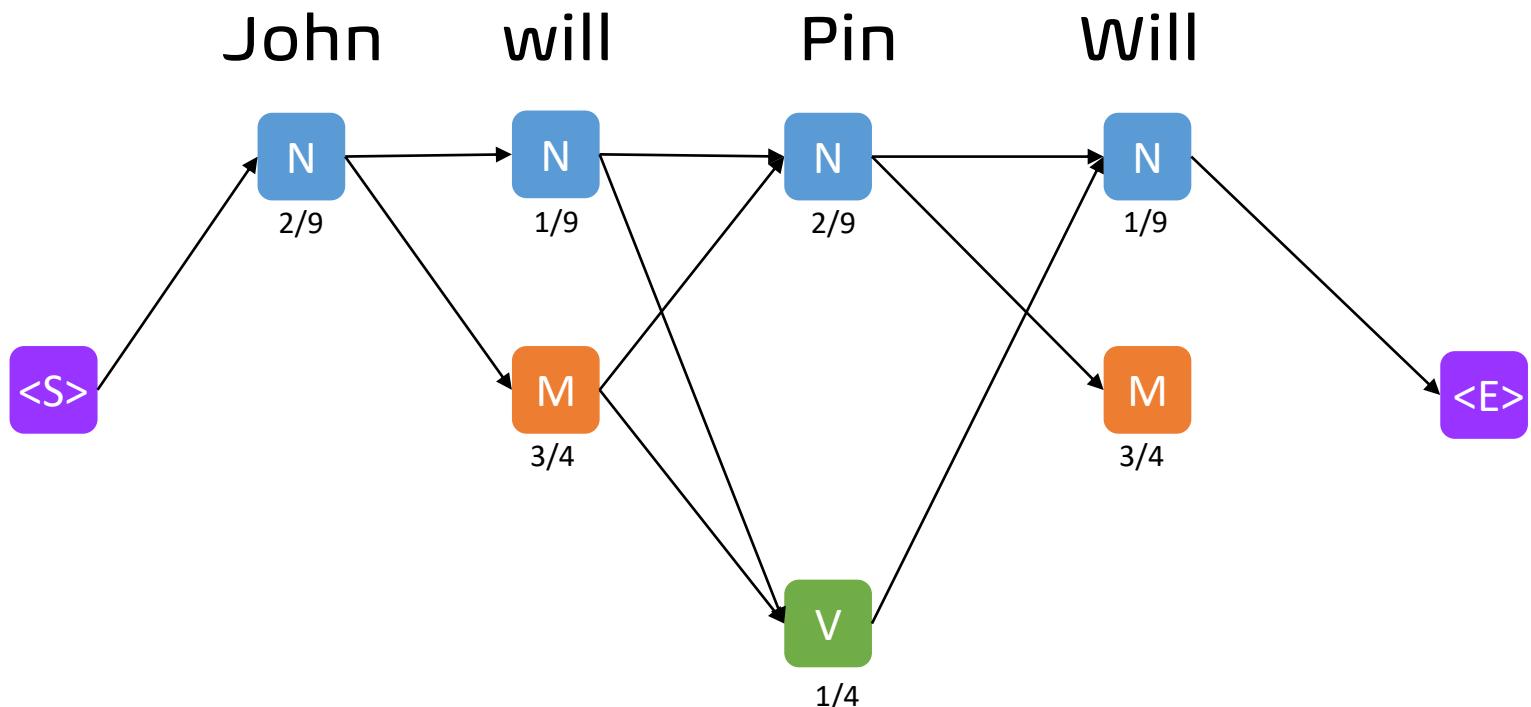


3 Probabilistic Approaches

POS Tagging: with HMM

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0

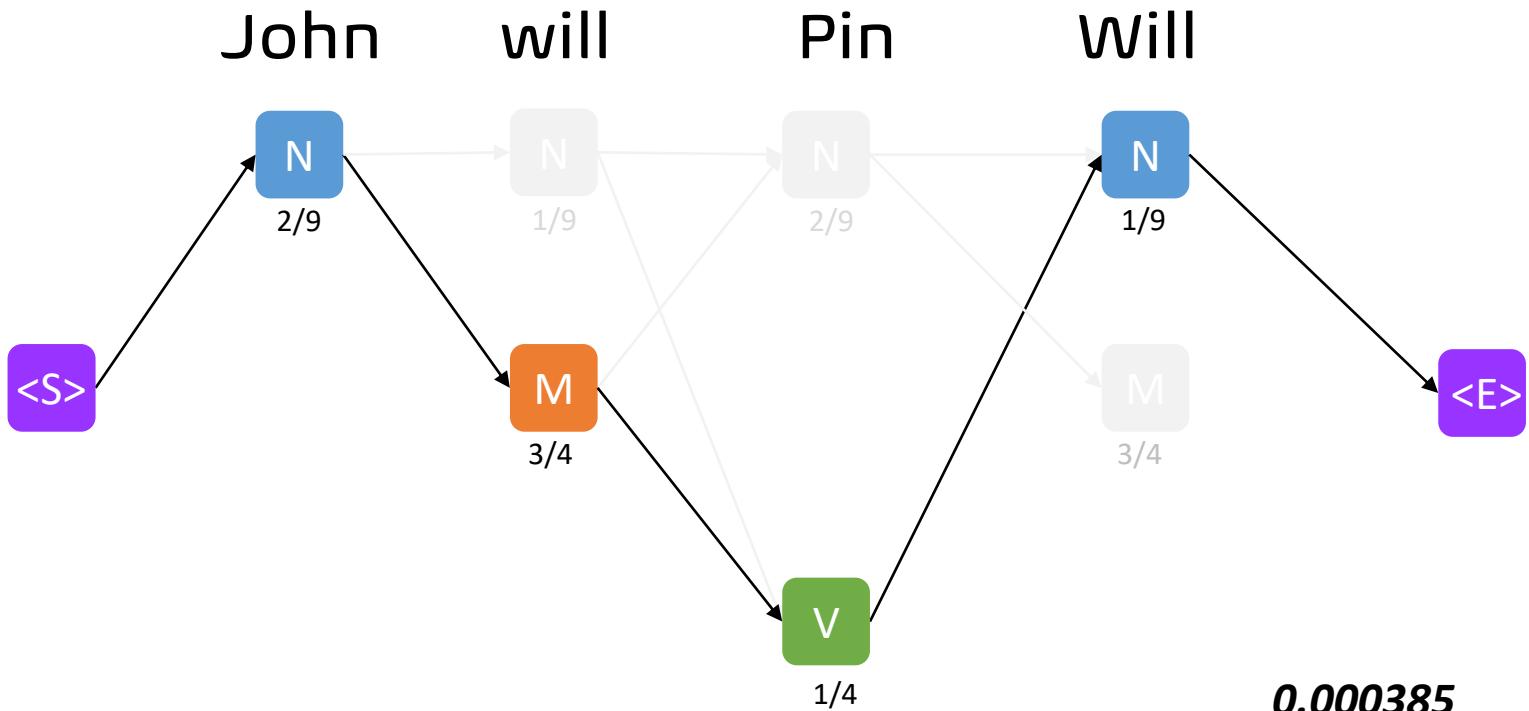


Which Path is the most likely?

POS Tagging: with HMM

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

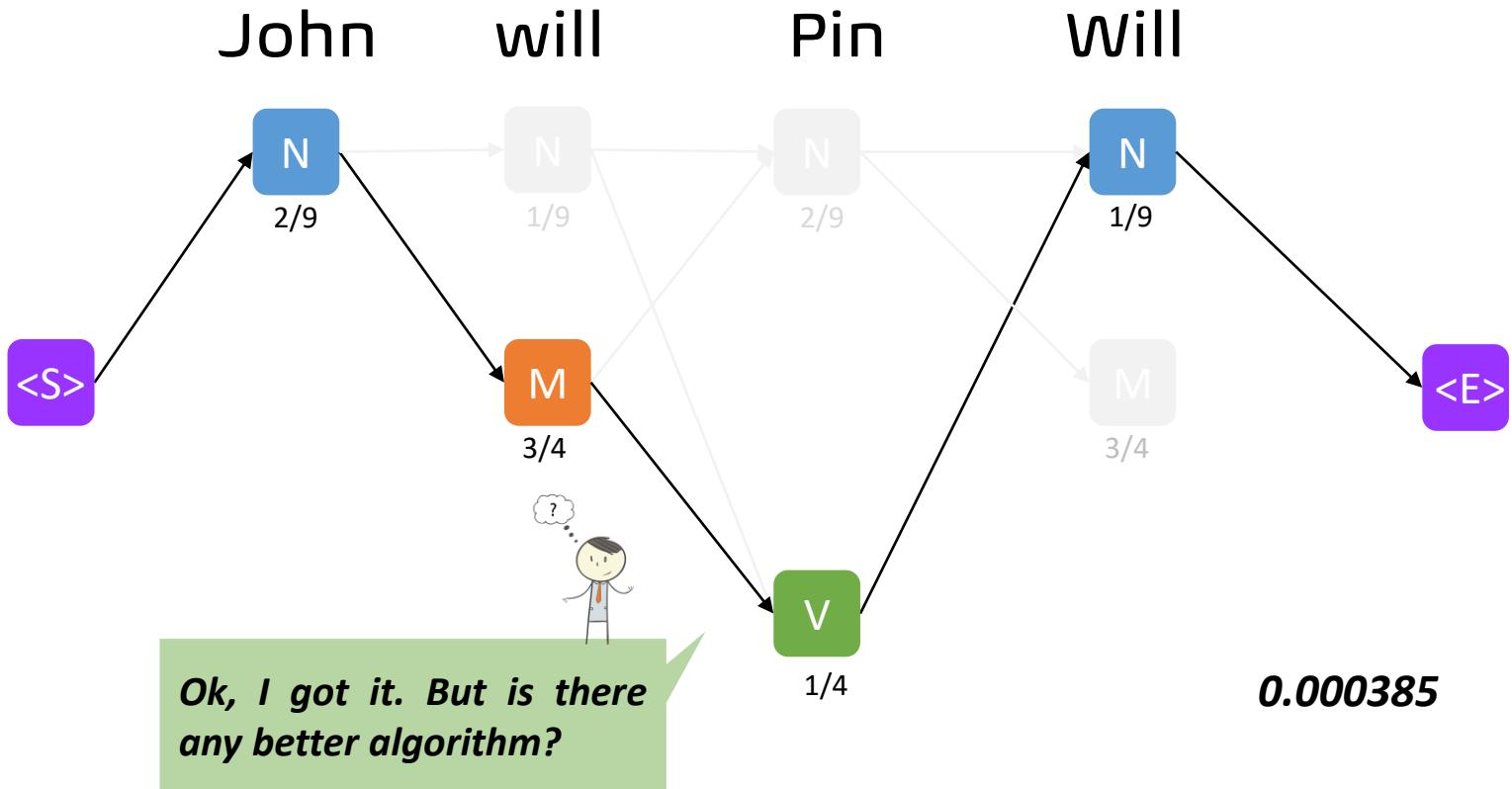
	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



POS Tagging: with HMM

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

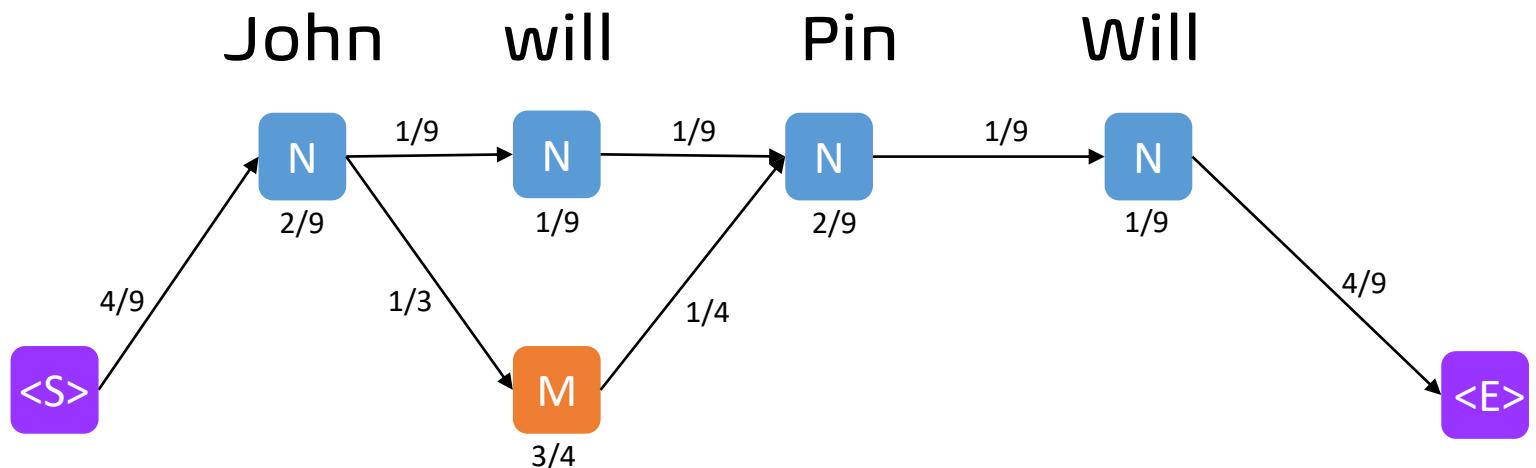
	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



3 Probabilistic Approaches

Viterbi Algorithm!

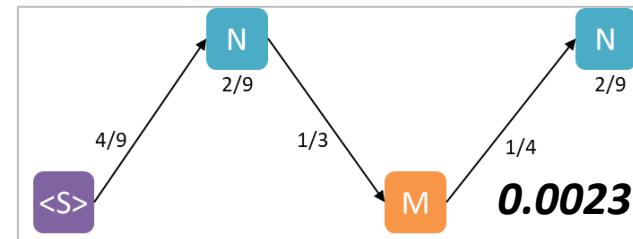
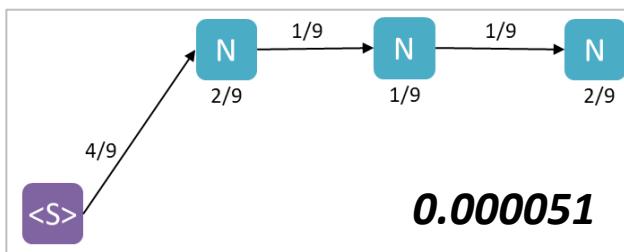
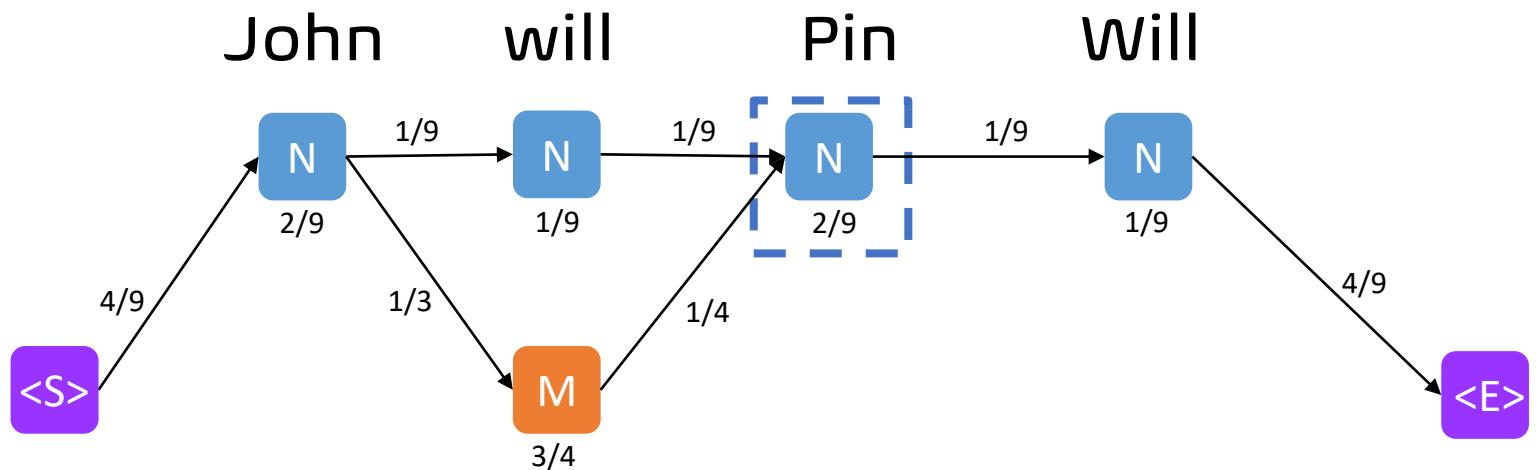
Assume we have only these options now



3 Probabilistic Approaches

Viterbi Algorithm!

Assume we have only these options now

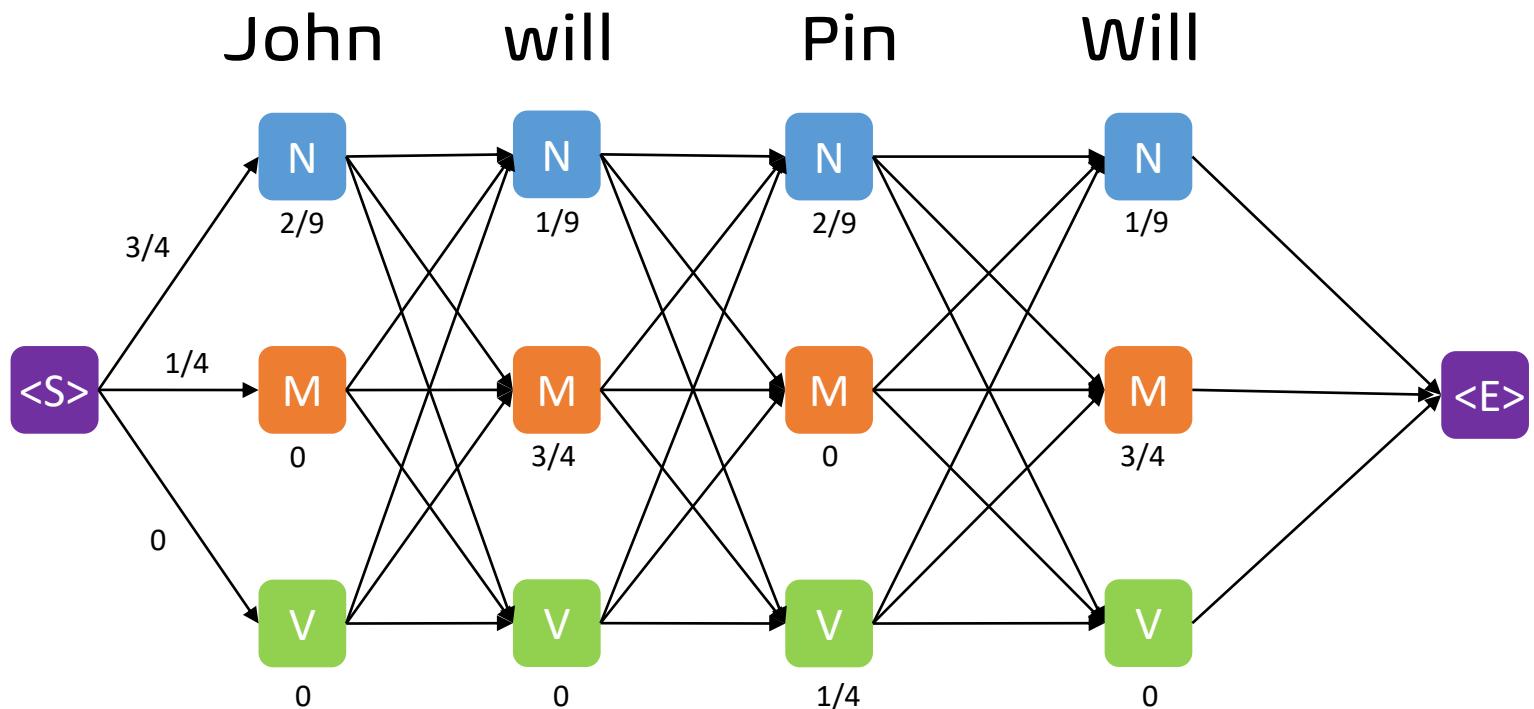


3 Probabilistic Approaches

Viterbi Algorithm

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



3 Probabilistic Approaches

Viterbi Algorithm

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

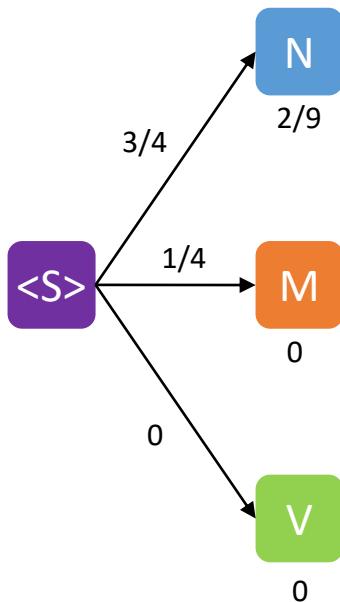
	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0

John

will

Pin

Will

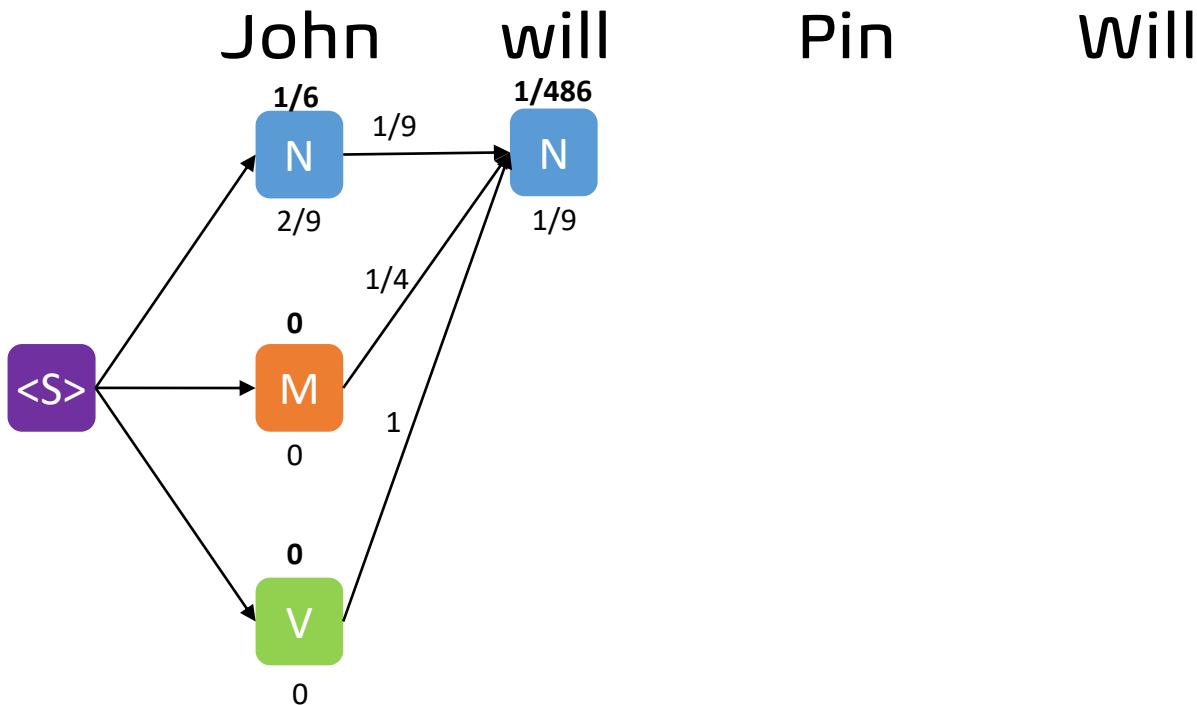


3 Probabilistic Approaches

Viterbi Algorithm

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0

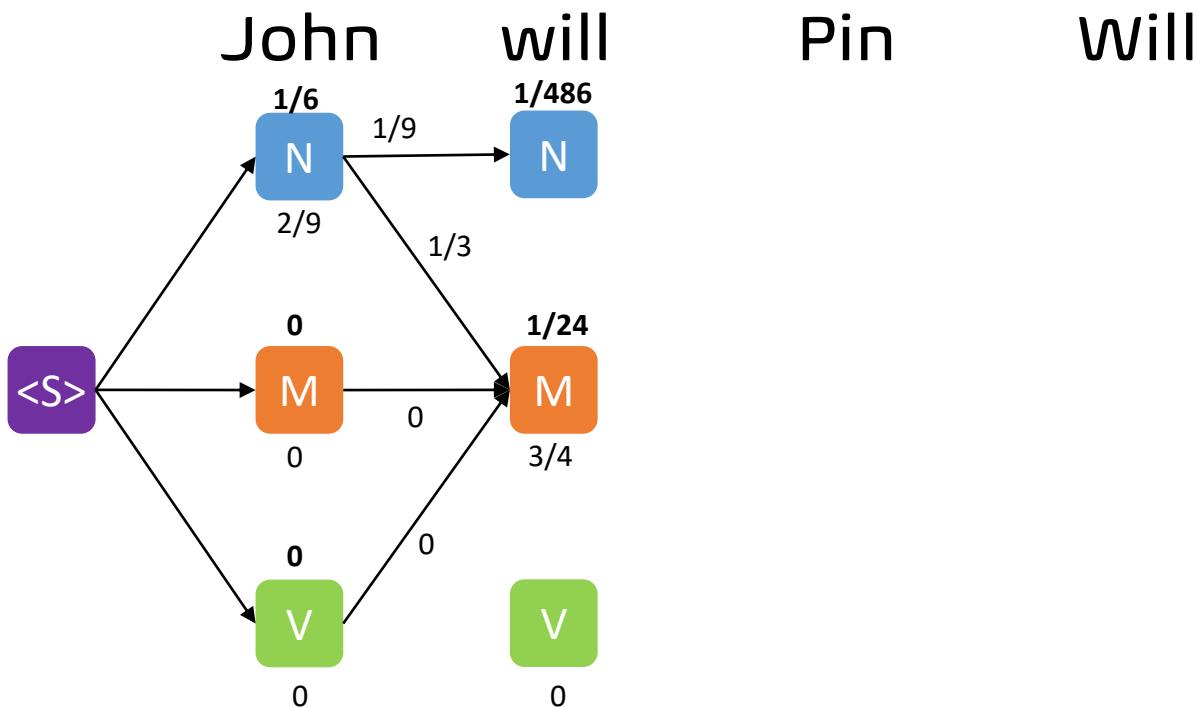


3 Probabilistic Approaches

Viterbi Algorithm

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

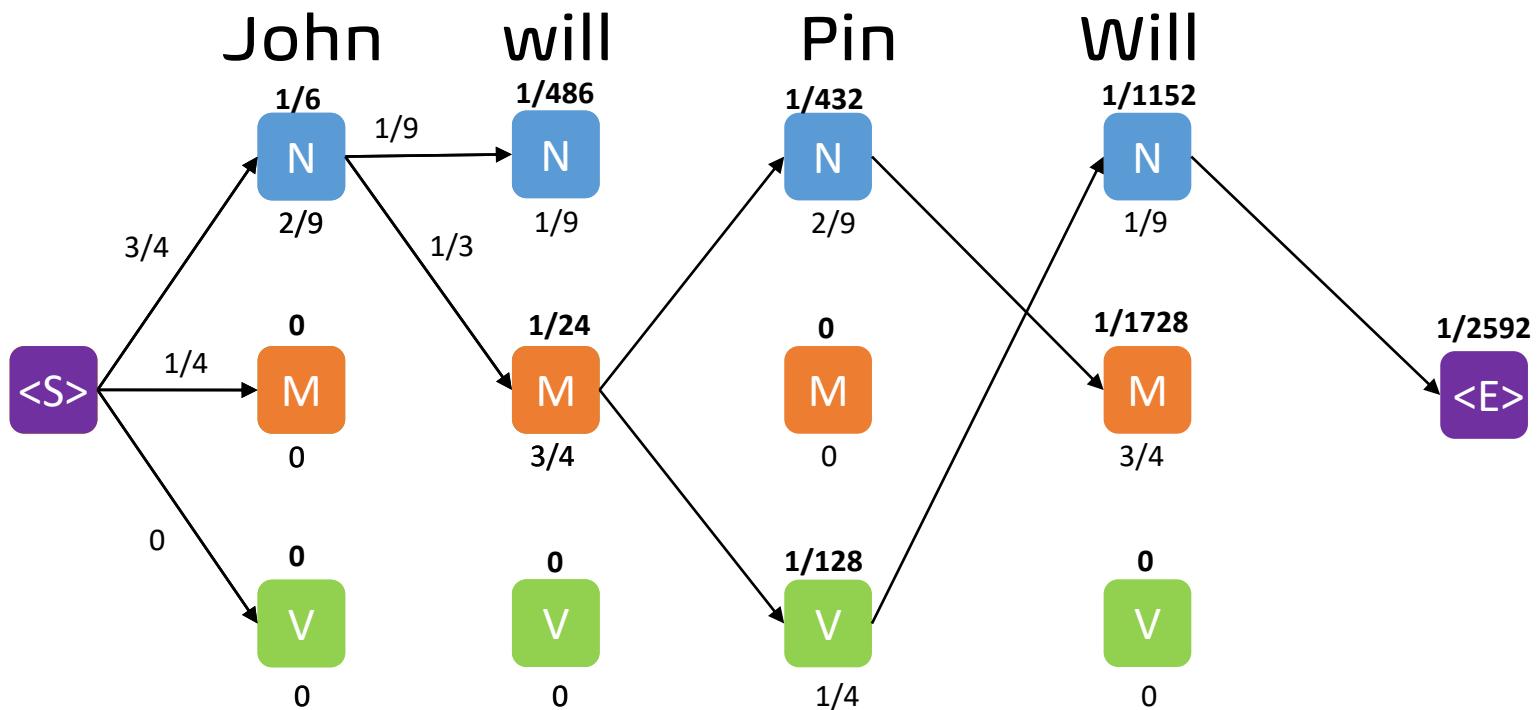
	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



Viterbi Algorithm

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

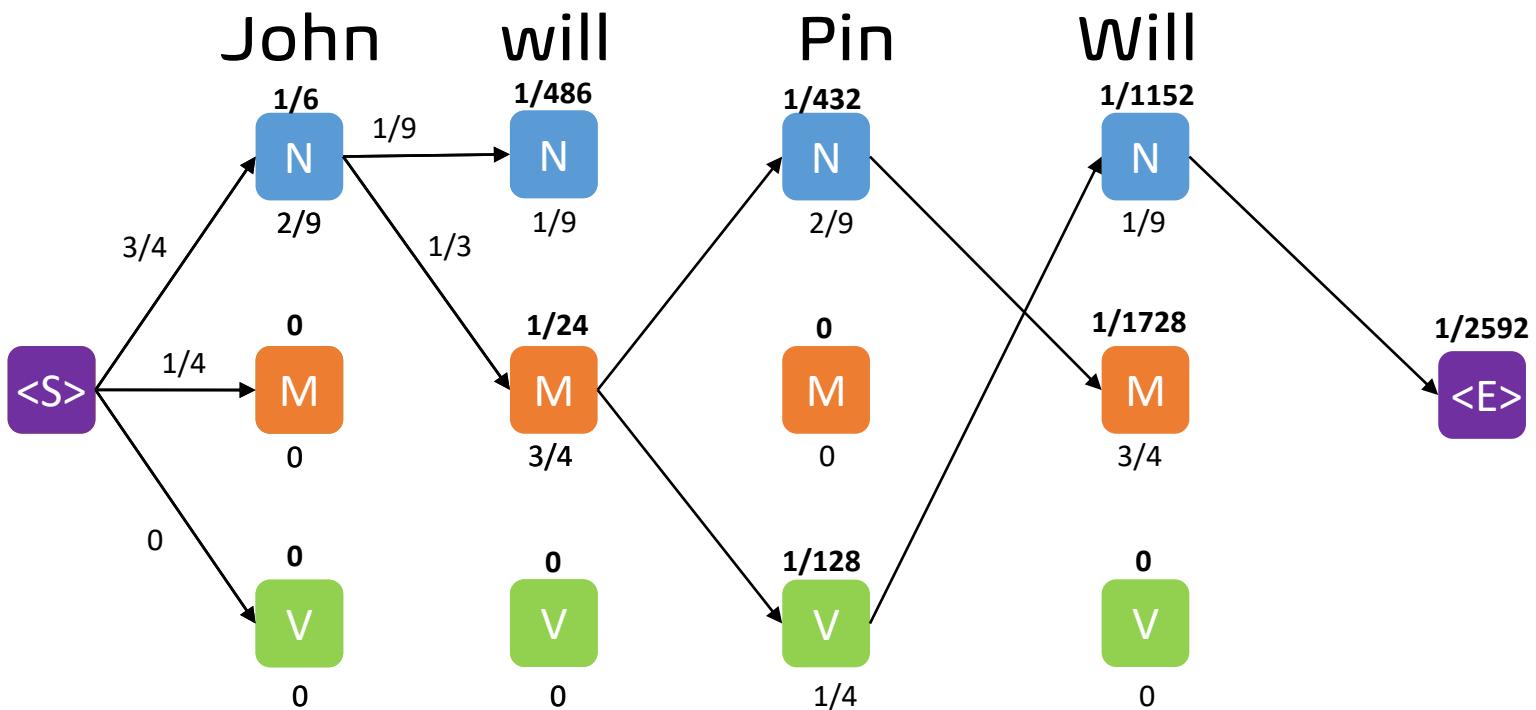
	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



Viterbi Algorithm

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

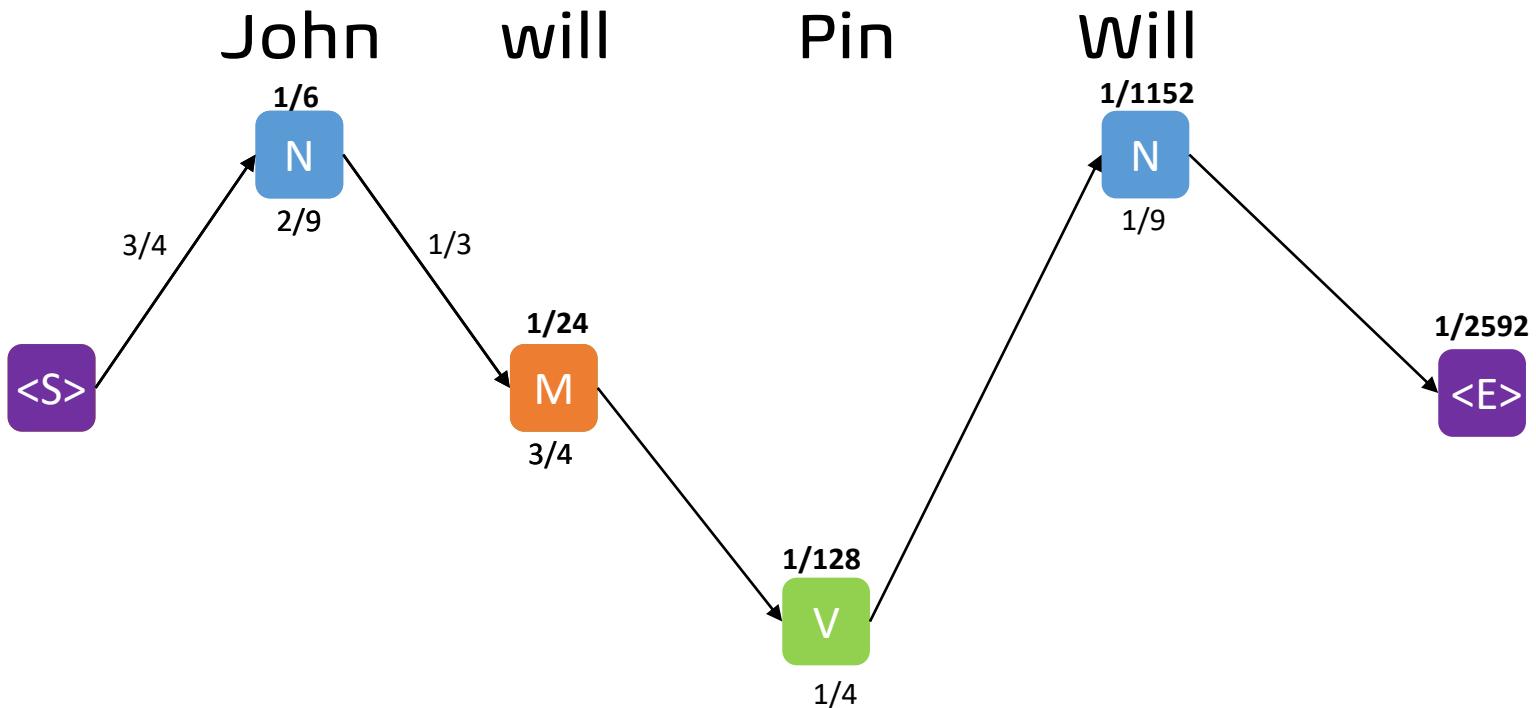
	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



Viterbi Algorithm

	N	V	M
Emma	4/9	0	0
John	2/9	0	0
Will	1/9	0	3/4
Pin	2/9	1/4	0
Can	0	0	1/4
Meet	0	2/4	0
Pat	0	1/4	0

	N	V	M	<E>
<S>	3/4	0	1/4	0
N	1/9	1/9	3/9	4/9
V	4/4	0	0	0
M	1/4	3/4	0	0



3 Probabilistic Approaches

Viterbi Algorithm

```

function VITERBI(observations of len  $T$ ,state-graph of len  $N$ ) returns best-path, path-prob
  create a path probability matrix viterbi[ $N, T$ ]
  for each state  $s$  from 1 to  $N$  do ; initialization step
     $viterbi[s, 1] \leftarrow \pi_s * b_s(o_1)$ 
     $backpointer[s, 1] \leftarrow 0$ 
  for each time step  $t$  from 2 to  $T$  do ; recursion step
    for each state  $s$  from 1 to  $N$  do
       $viterbi[s, t] \leftarrow \max_{s'=1}^N viterbi[s', t - 1] * a_{s', s} * b_s(o_t)$ 
       $backpointer[s, t] \leftarrow \operatorname{argmax}_{s'=1}^N viterbi[s', t - 1] * a_{s', s} * b_s(o_t)$ 
     $bestpathprob \leftarrow \max_{s=1}^N viterbi[s, T]$  ; termination step
     $bestpathpointer \leftarrow \operatorname{argmax}_{s=1}^N viterbi[s, T]$  ; termination step
    bestpath  $\leftarrow$  the path starting at state bestpathpointer, that follows backpointer[] to states back in time
  return bestpath, bestpathprob

```

3 Probabilistic Approaches

Out-of-Vocab

HMM Tagger Issue: #1. Unknown (OOV) Words

How to handle if there are any unknown words

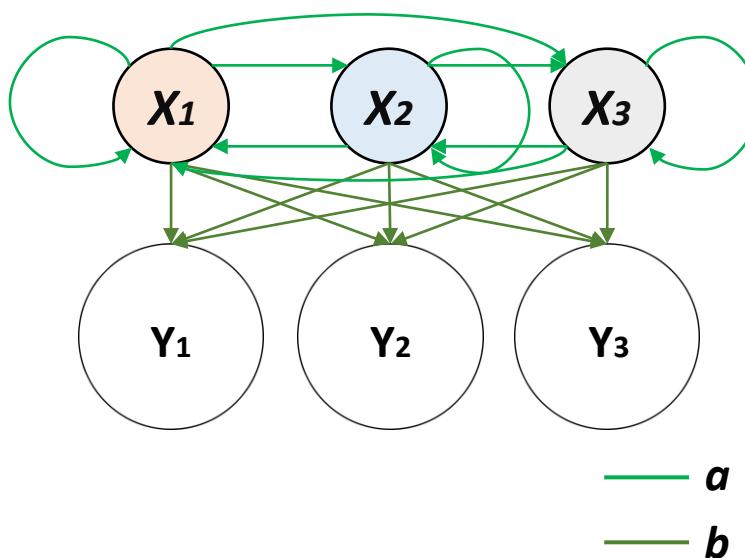
Solution 1: Use N-grams to predict the correct Tag

Solution 2: Use morphology (prefixes, suffixes) or hyphenation

3 Probabilistic Approaches

HMM Tagger Issue: #2. Independency Problem

HMM is only dependent on every state and its corresponding observed object. The sequence labeling, in addition to having a relationship with individual words, also relates to such aspects as the observed sequence length, word context and others.

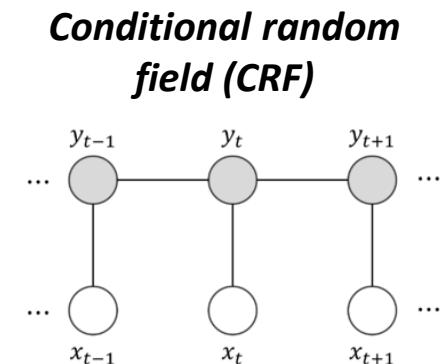
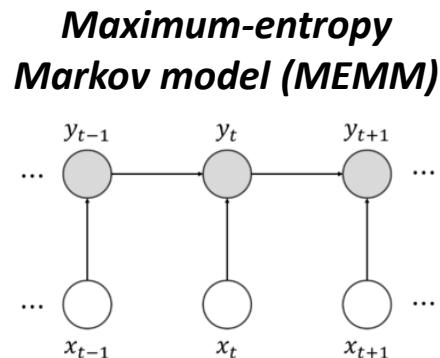
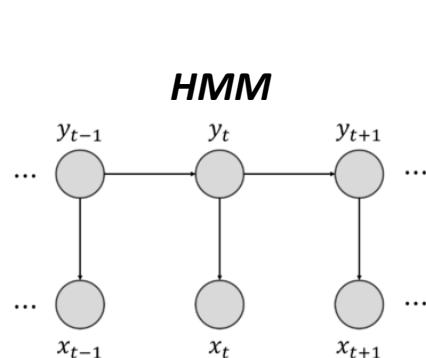


3 Probabilistic Approaches

Advanced HMM (MEMM or CRF)

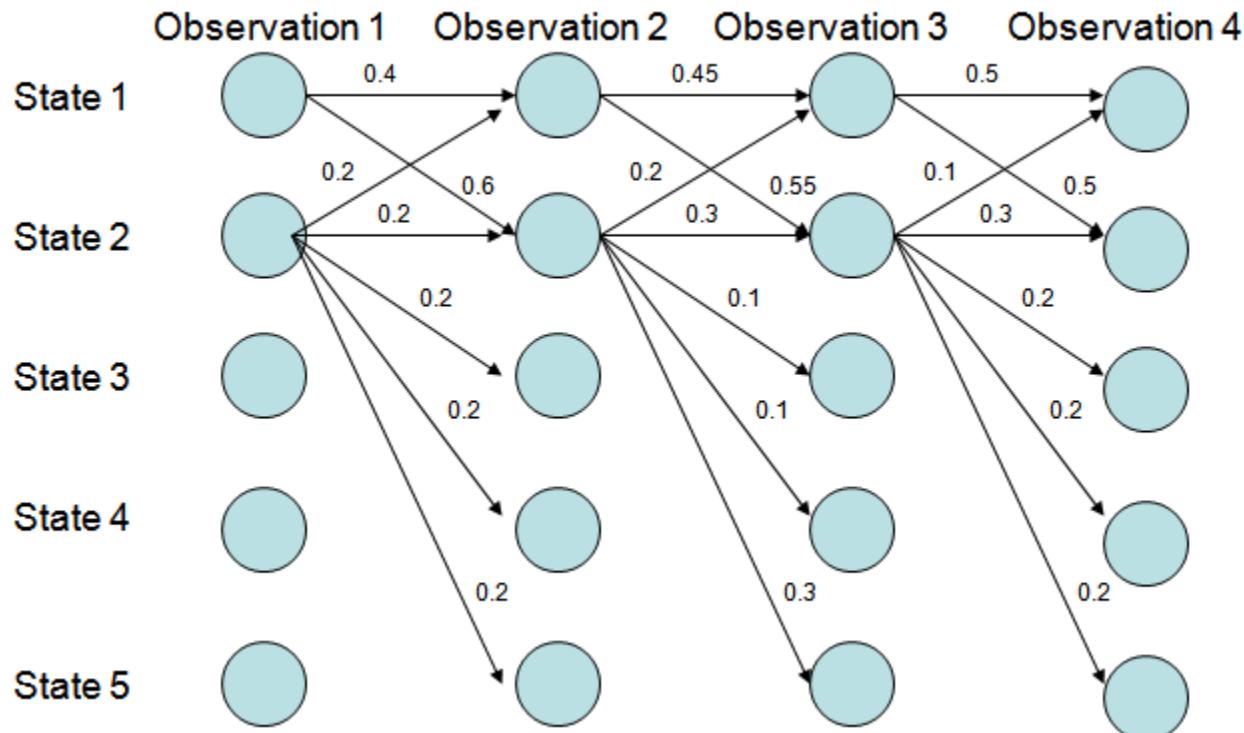
The CRF model has addressed the labeling bias issue and eliminated two unreasonable hypotheses in HMM.

MEMM adopts local variance normalization while CRF adopts global variance normalization.



3 Probabilistic Approaches

MEMM Labeling Bias



3 Probabilistic Approaches

Conditional Random Field: Advantages

- Compared with HMM: Since CRF does not have as strict independence assumptions as HMM does, it can accommodate any context information.
- Compared with MEMM: Since CRF computes the conditional probability of global optimal output nodes, it overcomes the drawbacks of label bias in MEMM.

MEMM suffers from Label Bias Problem, i.e., the transition probabilities of leaving a given state is normalized for only that state

However,

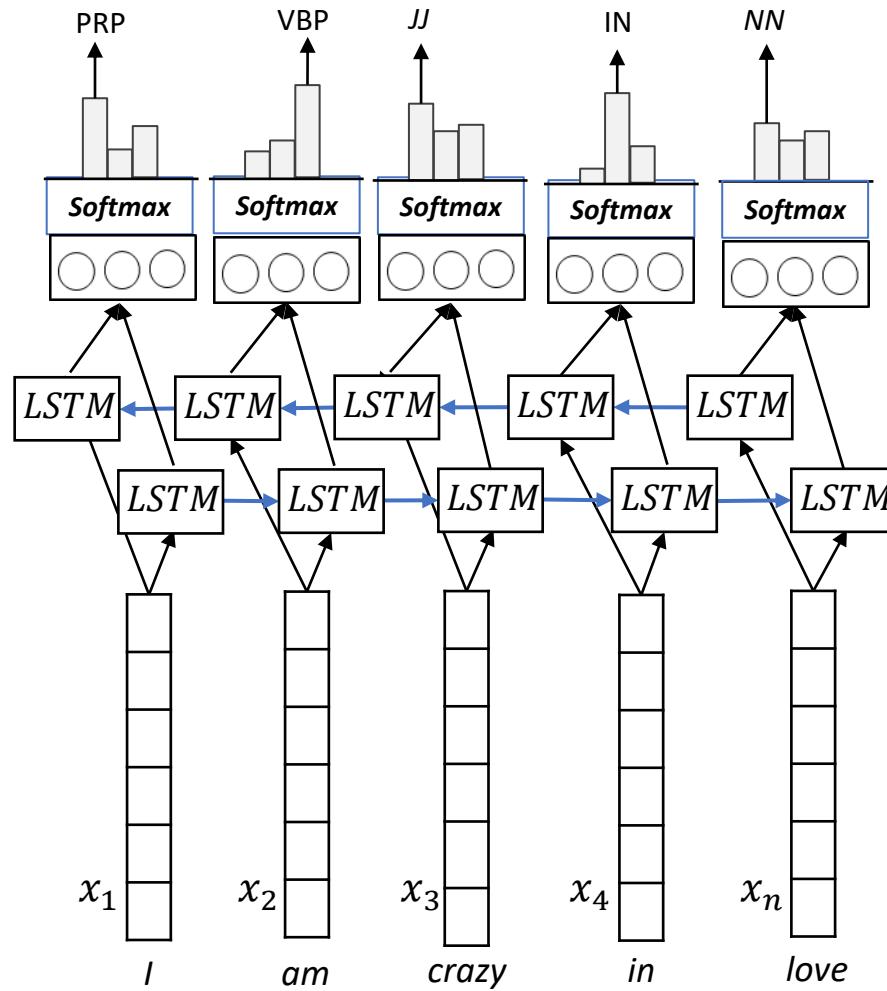
CRF is highly ***computationally complex at the training stage*** of the algorithm. It makes it ***very difficult to re-train the model*** when newer data becomes available.

0 LECTURE PLAN

Lecture 6: Part of Speech Tagging

1. Part-of-Speech Tagging
2. Baseline Approaches
 1. Rule-based Model
 2. Look-up Table Model
 3. N-Gram Model
3. Probabilistic Approaches
 1. Hidden Markov Model
 2. Conditional Random Field
4. Deep Learning Approaches

RNN/LSTM/GRU in Part of Speech Tagging



Do LSTMs really work so well for PoS tagging?

(*Horsmann and Zesch, 2017*)

Do LSTMs really work so well for PoS tagging? – A replication study

Tobias Horsmann and Torsten Zesch

Language Technology Lab

Department of Computer Science and Applied Cognitive Science

University of Duisburg-Essen, Germany

{tobias.horsmann,torsten.zesch}@uni-due.de

Abstract

A recent study by Plank et al. (2016) found that LSTM-based PoS taggers considerably improve over the current state-of-the-art when evaluated on the corpora of the Universal Dependencies project that use a *coarse-grained* tagset. We replicate this study using a fresh collection of 27 corpora of 21 languages that are annotated with *fine-grained* tagsets of varying size. Our replication confirms the result in general, and we additionally find that the advantage of LSTMs is even bigger for larger tagsets. However, we also find that for the very large tagsets of morphologically rich languages, hand-crafted morphological lexicons are still necessary to reach state-of-the-art performance.

ferty et al., 2001) and Hidden-Markov (HMM) implementations on corpora of various languages. Their evaluation concludes that the LSTM tagger reaches better results than the CRF and HMM tagger. The evaluation corpora were all annotated with a *coarse-grained* tagset with 17 tags. Thus, this LSTM tagger seems to be a well-performing, language-independent choice for learning models on coarse-grained tagsets. While for many tasks a coarse-grained tagset might be sufficient some tasks require more fine-grained tagsets.

We, thus, consider it worthwhile to explore if the results are reproducible using corpora with fine-grained tagsets. We use the LSTM tagger provided by Plank et al. (2016) and compare the results likewise to CRF and an off-the-shelf HMM tagger implementation. We compile a fresh set of 27 corpora of 21 languages which uses the commonly used *fine-grained* tagset of the respective

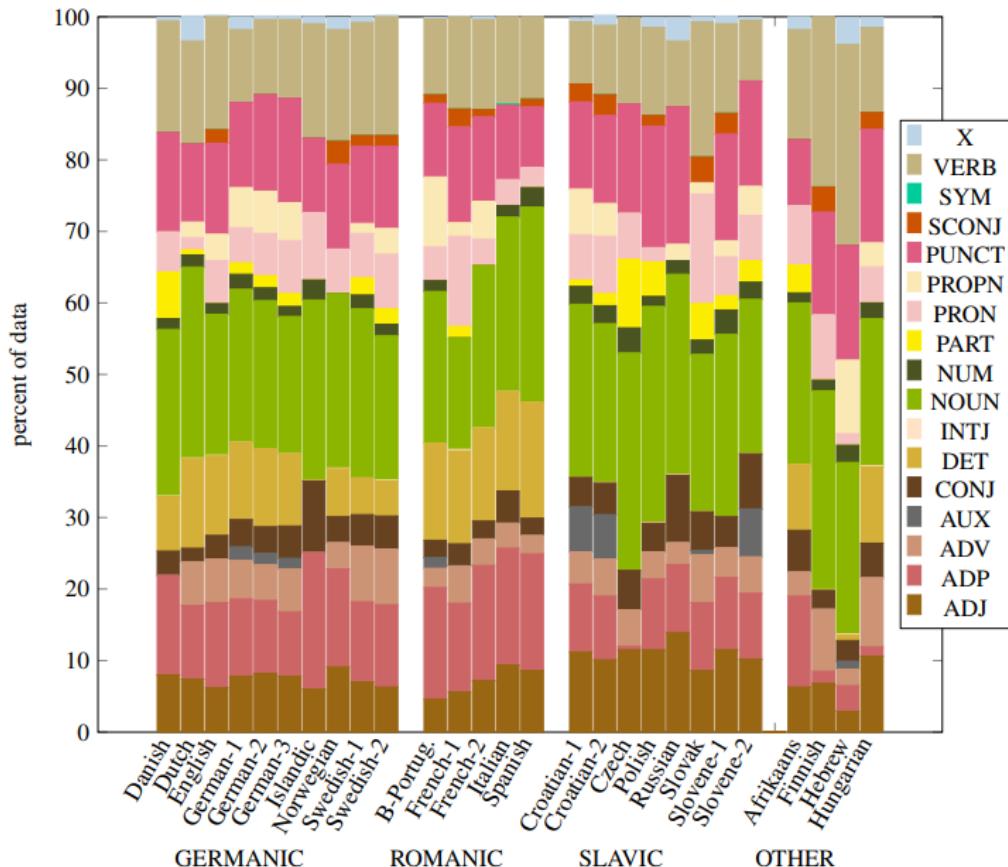
Do LSTMs really work so well for PoS tagging?

Corpora used in the experiments

Group	Corpus Id	Source	Tokens			Reference
			(10 ³)	# Tags	Annotation	
Germanic	Danish	Copenhagen DTB	255	36	manual	(Buch-Kromann and Korzen, 2010)
	Dutch	Alpino	200	20	manual	(Bouma et al., 2000)
	English	Brown	1,100	180	manual	(Nelson Francis and Kuçera, 1964)
	German-1	Hamburg DTB	4,800	54	manual	(Brants et al., 2004)
	German-2	Tiger	880	54	manual	(Telljohann et al., 2004)
	German-3	Tüba-D/Z	1,500	54	manual	(Foth et al., 2014)
	Icelandic	Mim	1,000	703	auto	(Helgadóttir et al., 2012)
	Norwegian	Norwegian DTB	1,300	19	manual	(Solberg et al., 2014)
	Swedish-1	Talbanken	96	25	manual	(Einarsson, 1976)
Romanic	Swedish-2	Stockholm-Umea	1,100	153	manual	(Ejerhed and Källgren, 1997)
	Braz.Portuguese	MAC-Morpho	1,000	82	manual	(Aluísio et al., 2003)
	French-1	Multitag	370	992	manual	(Paroubek, 2000)
	French-2	Sequoia	200	29	manual	(Candito et al., 2014)
	Italian	Turin Parallel	80	15	auto	(Bosco et al., 2012)
Slavic	Spanish	IULA DTB	550	241	manual	(Marimon et al., 2014)
	Croatian-1	Croatian DTB	200	692	manual	(Željko Agić and Ljubešić, 2014)
	Croatian-2	Hr500k	500	769	manual	(Ljubešić et al., 2016)
	Czech	Prague DTB	2,000	1,574	manual	(Bejček et al., 2013)
	Polish	Polish National Corpus	1,000	27	manual	(Przeźiókowski et al., 2008)
	Russian	Russian Open Corpus	1,700	22	manual	(Bocharov et al., 2013)
	Slovak	MULTEXT-East	84	956	manual	(Erjavec, 2010)
	Slovene-1	IJS-ELAN	540	1,181	auto	(Erjavec, 2002)
Others	Slovene-2	SSJ	590	1,304	manual	(Krek et al., 2013)
	Afrikaans	AfriBooms	50	12	manual	(Augustinus et al., 2016)
	Finnish	FinnTreebank	170	1573	manual	(Voutilainen, 2011)
	Hebrew	HaAretz Corpus	11,000	22	auto	(Itai and Wintner, 2008)
	Hungarian	The Szeged Treebank	1,200	1,085	manual	(Cséndes et al., 2005)

Do LSTMs really work so well for PoS tagging?

Coarse-grained PoS tag distribution of corpora by language group



Do LSTMs really work so well for PoS tagging?

(Horsmann and Zesch, 2017)

Lang. Group	Corpus Id	Word Ngrams ±1				Top 750 Char Ngrams				Clusters		Best CRF		HunPos	
		All	OOV	All	OOV	All	OOV	All	OOV	All	OOV	All	OOV	All	OOV
Germanic	Danish	90.9	53.3	90.3	69.3	89.5	67.6	96.1	82.4	94.9	74.2				
	Dutch	86.5	66.9	85.0	71.7	88.0	77.7	90.7	83.7	89.9	80.6				
	English	87.5	45.1	90.3	70.1	89.1	64.0	94.6	80.2	93.8	77.7				
	German-1	88.5	62.4	90.3	77.7	90.8	73.7	94.6	84.6	94.4	83.7				
	German-2	87.2	60.3	90.9	77.7	90.8	76.1	95.2	87.1	94.9	85.4				
	German-3	86.3	58.5	91.7	76.8	91.6	77.6	94.4	85.0	94.4	83.9				
	Icelandic	67.5	14.2	76.5	45.1	68.3	28.9	80.9	53.6	79.8	51.9				
	Norwegian	92.4	77.1	91.6	80.6	92.8	82.7	96.1	89.7	95.5	86.5				
Romance	Swedish-1	91.1	70.6	92.9	82.2	92.3	79.9	96.3	90.3	95.6	85.9				
	Swedish-2	78.7	29.7	87.2	67.3	81.4	48.8	91.0	74.6	91.4	77.6				
	B-Portug.	86.9	62.8	87.8	73.6	89.7	76.0	92.8	83.8	93.3	84.2				
	French-1	81.9	40.1	85.9	66.5	81.6	58.2	89.2	75.7	88.2	71.8				
	French-2	95.4	67.3	93.8	74.5	91.9	79.3	97.7	88.2	97.4	82.4				
Slavic	Italian	93.3	68.6	91.6	74.8	91.7	75.5	96.4	86.5	95.8	80.8				
	Spanish	88.5	45.5	94.5	78.2	88.1	58.8	96.4	83.5	96.6	83.6				
	Croatian-1	69.0	18.6	80.6	56.3	75.2	47.2	84.9	65.4	84.7	66.7				
	Croatian-2	66.3	15.9	78.5	54.4	73.5	44.8	83.4	63.9	82.6	63.9				
	Czech	64.1	14.4	79.2	56.0	75.2	39.2	83.1	62.9	81.7	60.9				
Other	Polish	82.9	58.1	92.5	86.9	86.5	72.5	95.5	91.5	93.6	85.4				
	Russian	83.7	53.7	93.0	83.5	88.2	70.9	95.5	87.5	94.6	83.6				
	Slovak	67.7	14.9	80.5	57.8	65.6	31.9	83.5	63.8	82.9	61.6				
	Slovene-1	72.6	17.4	83.5	55.6	72.4	39.4	86.4	62.5	82.6	59.6				
	Slovene-2	65.4	12.1	78.2	50.5	73.0	39.0	83.0	59.4	86.2	59.5				
Afrikaans	Afrikaans	95.7	75.0	95.3	80.3	95.8	81.9	97.8	89.6	97.3	85.5				
	Finnish	62.6	10.0	77.1	48.5	67.8	33.8	82.3	56.7	81.3	55.8				
	Hebrew	82.3	41.7	81.3	60.9	76.3	53.3	90.5	68.5	90.3	60.1				
	Hungarian	72.7	13.9	86.7	63.3	72.0	31.7	89.9	69.6	89.4	69.5				

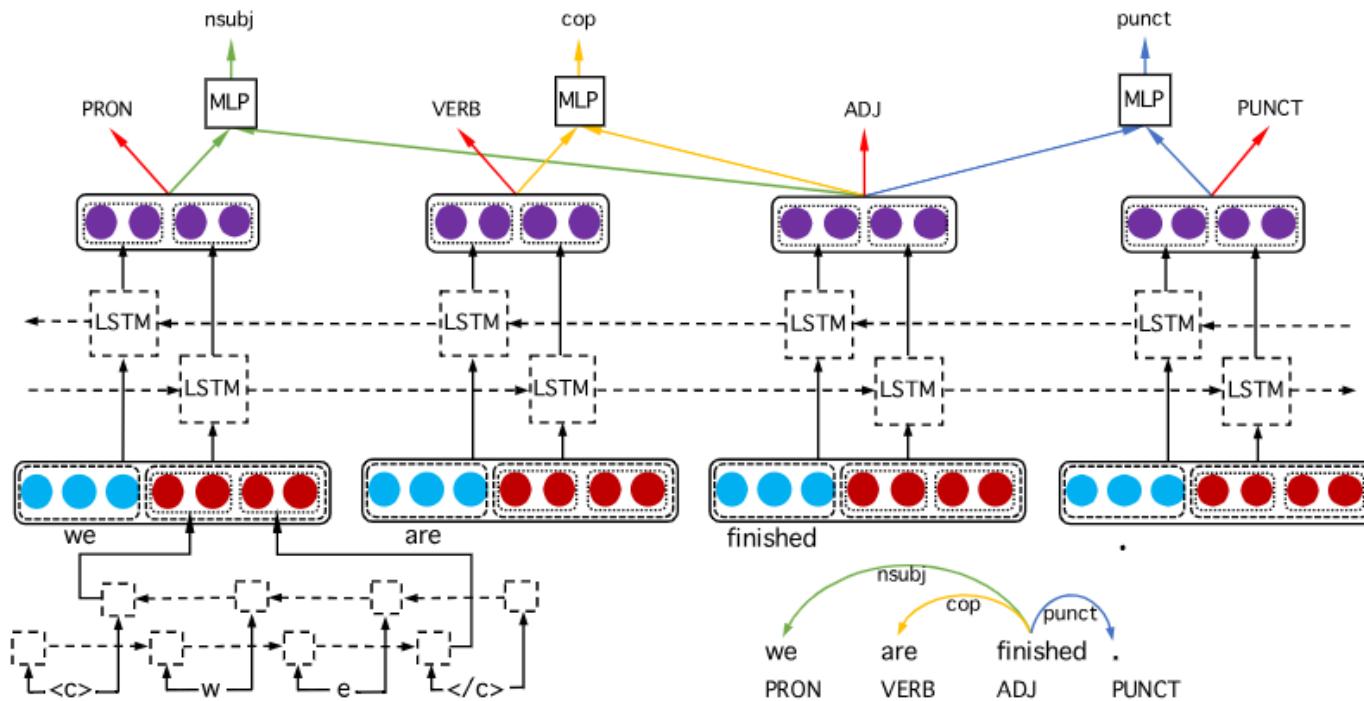
Table 2: Accuracy of CRF taggers (10fold CV)

Lang. Group	Corpus Id	Word		Char		Word-Char		Word-Char+		HunPos	
		All	OOV	All	OOV	All	OOV	All	OOV	All	OOV
Germanic	Danish	94.9	72.7	95.0	79.1	96.4	82.5	96.9	83.4	94.9	74.2
	Dutch	91.1	82.3	90.3	83.6	91.6	85.7	92.5	87.1	89.9	80.6
	English	91.9	65.9	92.3	77.4	94.1	79.6	94.9	80.9	93.8	77.7
	German-1	93.6	78.3	94.1	84.5	95.6	87.6	96.0	88.3	94.4	83.7
	German-2	94.5	82.4	94.6	87.1	96.4	90.1	96.8	91.5	94.4	85.4
	German-3	93.8	80.3	94.0	84.9	95.8	88.6	96.4	89.8	94.4	83.9
	Icelandic	76.0	34.8	76.5	49.3	81.8	56.2	84.1	60.6	79.8	51.9
	Norwegian	95.8	86.2	95.7	88.2	96.6	90.3	96.9	90.3	95.5	86.5
Romance	Swedish-1	94.9	81.4	95.3	86.7	96.2	89.0	96.7	89.8	95.6	85.9
	Swedish-2	86.5	54.3	88.9	74.3	91.8	78.5	92.5	80.4	91.4	77.6
	B-Portug.	93.3	82.4	93.9	87.4	95.0	90.3	95.1	90.8	93.3	84.2
	French-1	87.6	67.0	85.8	72.0	88.7	77.4	89.7	78.7	88.2	71.8
	French-2	97.5	80.4	97.4	83.4	98.1	87.7	98.3	88.7	97.4	82.4
Slavic	Italian	96.0	81.3	95.6	84.2	96.5	85.9	97.1	86.9	95.8	80.8
	Spanish	93.1	63.3	96.4	85.5	96.9	86.1	97.2	87.0	96.6	83.6
	Croatian-1	83.2	55.5	83.8	67.5	88.1	72.8	89.1	75.2	84.7	66.9
	Croatian-2	80.3	52.4	81.1	63.8	84.9	69.1	86.8	72.4	82.6	63.9
	Czech	79.4	49.1	81.0	62.7	85.8	68.7	87.7	72.4	81.7	60.9
Other	Polish	86.9	73.6	89.2	84.7	95.5	91.2	91.2	88.0	93.6	85.4
	Russian	91.3	73.2	94.6	85.8	95.3	86.9	96.0	88.4	94.6	83.6
	Slovak	78.7	44.9	80.6	65.0	85.3	69.7	86.6	71.4	82.9	61.6
	Slovene-1	81.9	44.5	83.9	61.1	86.0	62.6	87.9	65.7	82.6	59.6
	Slovene-2	79.9	47.9	82.0	63.4	85.8	67.4	87.5	70.1	86.2	59.5
Afrikaans	Afrikaans	97.3	82.8	97.1	85.8	97.8	88.4	98.0	90.0	97.3	85.5
	Finnish	76.7	42.7	78.0	57.6	82.0	58.9	83.6	61.2	81.3	55.8
	Hebrew	89.9	60.2	89.2	66.9	92.2	69.7	92.9	72.1	90.3	60.1
	Hungarian	84.7	53.3	88.0	73.1	91.2	76.9	92.0	79.0	89.4	69.5

Table 3: Accuracy of LSTM taggers (10fold CV)

LSTM-based POS Tagging

Illustration of LSTM-based joint POS tagging and graph-based dependency parsing.



/ Reference

Summary

- M. Marcus, B. Santorini and M.A. Marcinkiewicz (1993). Building a large annotated corpus of English: The Penn Treebank. In Computational Linguistics, volume 19, number 2, pp. 313–330.
- Nguyen, D. Q., Dras, M., & Johnson, M. (2017). A novel neural network model for joint pos tagging and graph-based dependency parsing. arXiv preprint arXiv:1705.05952.
- Ling, W., Luís, T., Marujo, L., Astudillo, R. F., Amir, S., Dyer, C., ... & Trancoso, I. (2015). Finding function in form: Compositional character models for open vocabulary word representation. arXiv preprint arXiv:1508.02096.
- Horsmann, T., & Zesch, T. (2017). Do LSTMs really work so well for PoS tagging?—A replication study. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (pp. 727-736).
- Speech and Language Processing. Daniel Jurafsky & James H. Martin. Copyright c 2019. All rights reserved. Draft of October 2, 2019.

The lecture will be started at 5:05PM sharply!

COMP5046

Natural Language Processing

Lecture 7: Dependency Parsing

Dr. Caren Han

Semester 1, 2022

*School of Computer Science,
University of Sydney*



0 The course topics

What will you learn in this course?

Week 1: Introduction to Natural Language Processing (NLP)

Week 2: Word Embeddings (Word Vector for Meaning)

Week 3: Word Classification with Machine Learning I

Week 4: Word Classification with Machine Learning II

NLP and
Machine
Learning

Week 5: Language Fundamental

Week 6: Part of Speech Tagging

Week 7: Dependency Parsing

Week 8: Language Model and Natural Language Generation

NLP
Techniques

Week 9: Information Extraction: Named Entity Recognition

Week 10: Advanced NLP: Attention and Reading Comprehension

Week 11: Advanced NLP: Transformer and Machine Translation

Week 12: Advanced NLP: Pretrained Model in NLP

Advanced
Topic

Week 13: Future of NLP and Exam Review

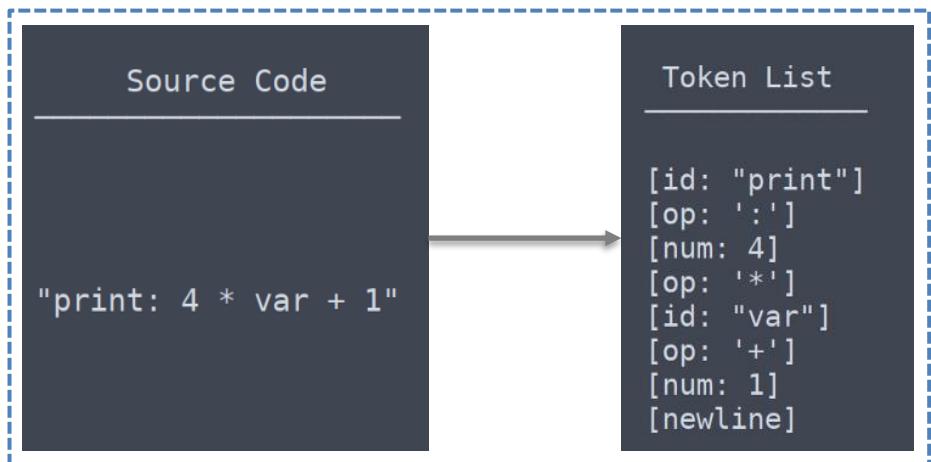
0 LECTURE PLAN

Lecture 7: Parsing

1. Linguistic Structure
2. Dependency Structure
3. Dependency Parsing Algorithms
4. Transition-based Dependency Parsing
5. Deep Learning-based Dependency Parsing

1 Linguistic Structure

Computer Language



Tokenisation

A token can be a variable or function name, an operator or a number.

1 Linguistic Structure

Parsing Computer Language



Parsing

The parser turns a list of tokens into a tree of nodes – logical order

Can we apply this in a human (natural) language?

1 Linguistic Structure

Parsing Natural Language (Human Language)

Q: Can we apply this in a human (natural) language?

A: Possible! But it is much more difficult than parsing computer language!

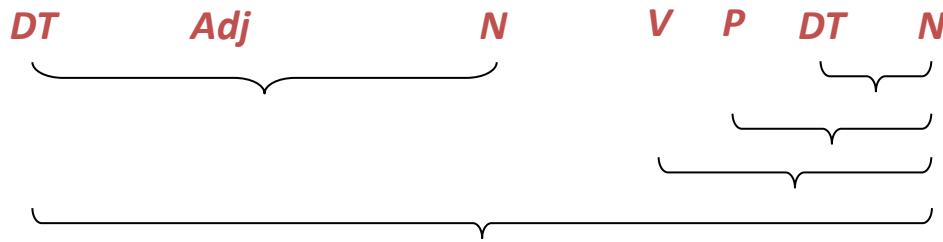
Why?

- *No types for words*
- *No brackets around phrases*
- *Ambiguity!*

Natural Language: Linguistic Structure

Let's try to categorise the given words (Part of Speech Tags)

The expensive computer is on the table



However, Language is **more than just a “bag of words”**.

Grammatical rules apply to combine:

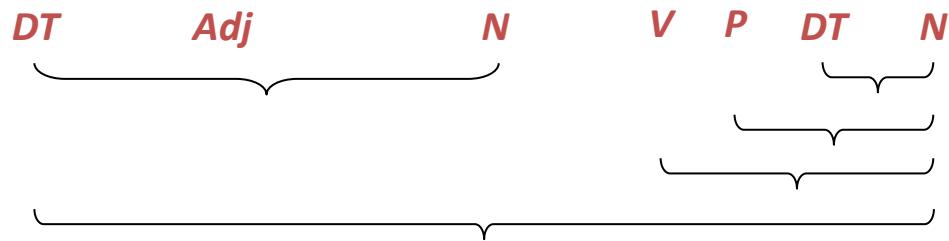
- words **into phrases**
- phrases into bigger phrases

1 Linguistic Structure

Natural Language: Linguistic Structure

Let's try to categorise the given words (Part of Speech Tags)

The expensive computer is on the table



However, Language is **more than just a “bag of words”**.

Grammatical rules apply to combine:

- words **into phrases**
- phrases into bigger phrases

Example: a sentence includes **a subject** and **a predicate**.

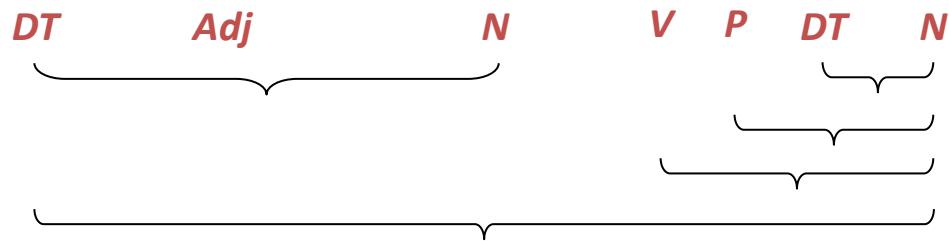
The **subject** is noun phrase and the **predicate** is a verb phrases.

1 Linguistic Structure

Natural Language: Linguistic Structure

Phrase Structure Grammar = Context-free Grammar (CFG)

The expensive computer is on the table



However, Language is **more than just a “bag of words”**.

Grammatical rules apply to combine:

- words **into phrases**
- phrases into bigger phrases

Example: a sentence includes a subject and a predicate.

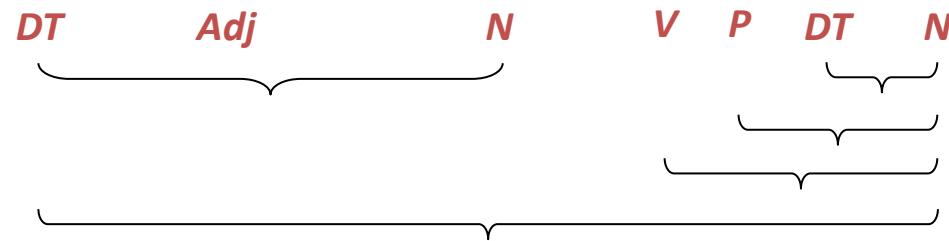
The subject is noun phrase and the predicate is a verb phrases.

1 Linguistic Structure

Natural Language: Linguistic Structure

Phrase Structure Grammar = Context-free Grammar (CFG)

The expensive computer is on the table



However, Language is **more than just a “bag of words”**.

Grammatical rules apply to combine:

- words **into phrases**
- phrases into bigger phrases

Parsing

- Associating tree structures to a sentence, given a grammar
(Context Free Grammar or Dependency Grammar)
will talk about this soon!

1 Linguistic Structure

Parsing Natural Language (Human Language)

Q: Can we apply this in a human (natural) language?

A: Possible! But it is much more difficult than parsing computer language!

Why?

- *No **types** for words*
- *No **brackets** around phrases*
- **Ambiguity!**

1 Linguistic Structure

Syntactic Ambiguities

Grammars are declarative

- *They don't specify how the parse tree will be constructed*

Ambiguity

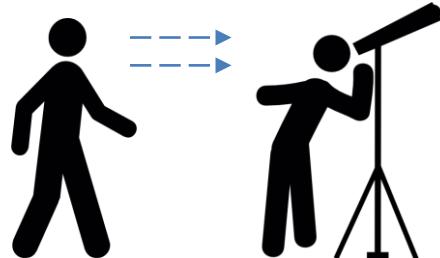
1. *Prepositional Phrase (PP) attachment ambiguity*
2. *Coordination Scope*
3. *Gaps*
4. *Particles or Prepositions*
5. *Gerund or adjective*

There are many more ambiguities...

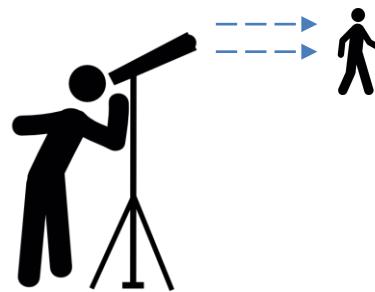
Syntactic Ambiguities – PP attachment Ambiguity

I saw the man with the telescope

I saw the man with the telescope



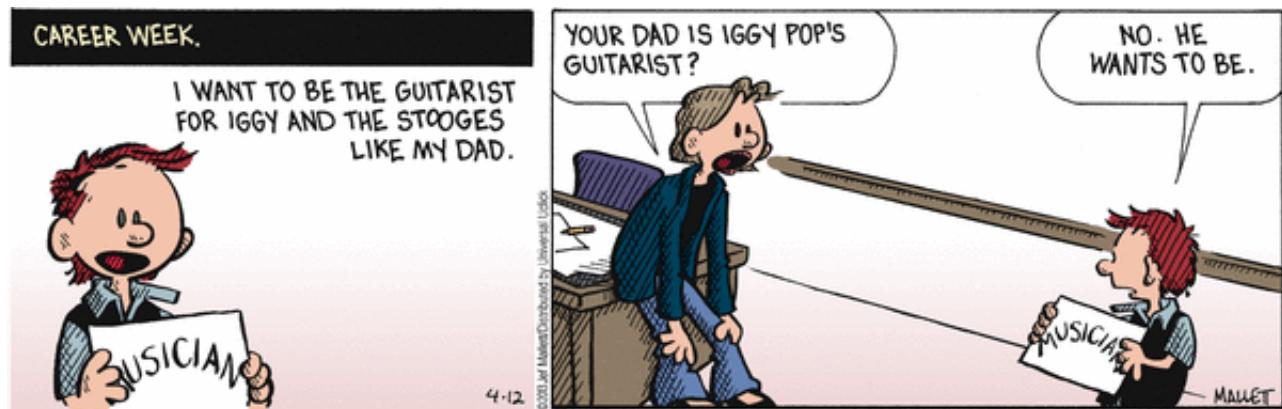
I saw the man with the telescope



1 Linguistic Structure

Syntactic Ambiguities – PP attachment Ambiguity Multiply

- A key parsing decision is how we ‘attach’ various constituents
 - *PPs, adverbial or participial phrases, infinitives, coordinations*



1 Linguistic Structure

Syntactic Ambiguities – Coordination Scope Ambiguity

I ate red apples and bananas



Syntactic Ambiguities - Gaps

She never saw a dog and did not smile



Syntactic Ambiguities – Particles or Prepositions

- Some verbs are followed by adverb particles.
(e.g. *put on*, *take off*, *give away*, *bring up*, *call in*)

*She **ran up** a large **bill***

*She ran **up** a large **hill***

Difference between an adverb particle and a preposition.

- the **particle** is closely tied to its verb to form idiomatic expressions
- the **preposition** is closely tied to the noun or pronoun it modifies.

1 Linguistic Structure

Syntactic Ambiguities – Gerund or Adjective

Dancing shoes can provide nice experience



Gerund



Adjective

1 Linguistic Structure

When and Where do we use Parsing?

Syntactic Analysis checks whether the generated tokens form a meaningful expression

- *Humans communicate complex ideas by composing words together into bigger units to convey complex meanings*
 - *We need to understand sentence structure in order to be able to interpret language correctly*
-
- *Grammar Checking*
 - *Question Answering*
 - *Machine Translation*
 - *Information Extraction*
 - *Text Classification*
 - *Chatbot*
- ... and many others*

1 Linguistic Structure

Two main views of linguistic structure

Constituency Grammar (a.k.a context-free grammar, phrase structure grammar)

- *Noam Chomsky (1928 -)*
- *Immediate constituent analysis*
- *Insists on classification and distribution*

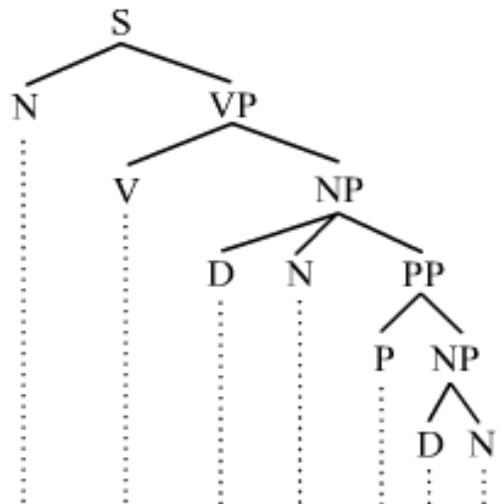
Dependency Grammar

- *Lucien Tesnière (1893 – 1954)*
- *Functional dependency relations*
- *Emphasises the relations between syntactic units,
thus adding meaningful links (semantics)*

1 Linguistic Structure

Two main views of linguistic structure

Constituency Parsing

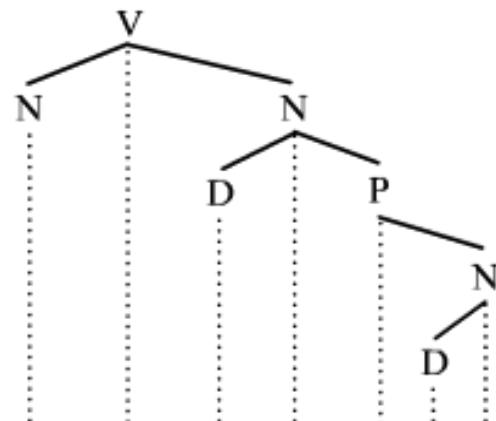


a. They killed the man with a gun.

Constituency grammars

One-to-one-or-more correspondence. For every word in a sentence, there is at least one node in the syntactic structure that corresponds to that word.

Dependency Parsing



b. They killed the man with a gun.

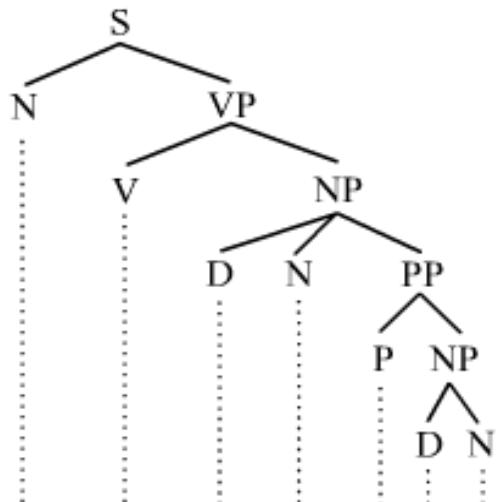
Dependency grammars

one-to-one relation; for every word in the sentence, there is exactly one node in the syntactic structure that corresponds to that word

1 Linguistic Structure

Two main views of linguistic structure

Constituency Parsing

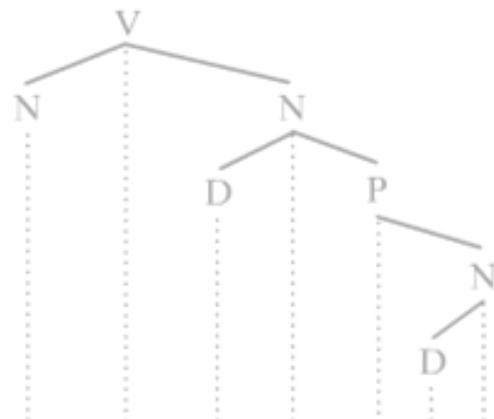


a. They killed the man with a gun.

Constituency grammars

One-to-one-or-more correspondence. For every word in a sentence, there is at least one node in the syntactic structure that corresponds to that word.

Dependency Parsing



b. They killed the man with a gun.

Dependency grammars

one-to-one relation; for every word in the sentence, there is exactly one node in the syntactic structure that corresponds to that word

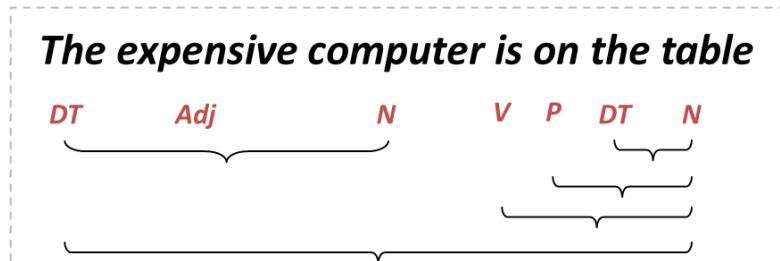
1 Linguistic Structure

Constituency Grammar

- A **basic observation about syntactic structure** is that groups of words can act as single units
- Such groups of words are called **constituents**
- Constituents tend to have **similar internal structure**, and behave similarly with respect to other units

Examples

- noun phrases (NP)
 - she, the house, Robin Hood and his merry men, etc.
- verb phrases (VP)
 - blushed, loves Mary, was told to sit down and be quiet, lived happily ever after
- prepositional phrases (PP)
 - on it, with the telescope, through the foggy dew, etc.



1 Linguistic Structure

A sample context-free grammar

I prefer a morning flight

1. Starting unit: words are *given a category (part-of-speech)*
2. Combining words into *phrases with categories*
3. Combining phrases into *bigger phrases recursively*

1 Linguistic Structure

A sample context-free grammar

1. Starting unit: words are *given a category (part-of-speech)*
2. Combining words into phrases with categories
3. Combining phrases into bigger phrases recursively

I, prefer, a, morning, flight

PRP VBP DT NN NN

1 Linguistic Structure

A sample context-free grammar

I, prefer, a, morning, flight

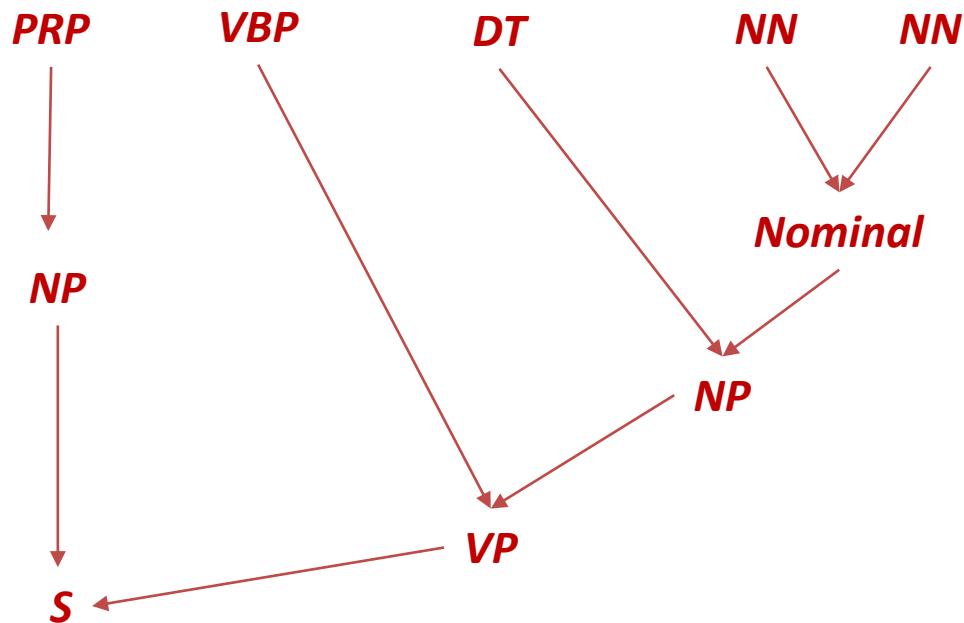
PRP VBP DT NN NN

Grammar rule	Example
S → NPVP	I + want a morning flight
NP → Pronoun	I
NP → Proper-Noun	Sydney
NP → Det Nominal	a flight
Nominal → Nominal Noun	morning flight
Nominal → Noun	flights
VP → Verb	do
VP → Verb NP	want + a flight
VP → Verb NP PP	leave + Melbourne + in the morning
VP → Verb PP	leaving + on Thursday
PP → Preposition NP	from + Sydney

A sample context-free grammar

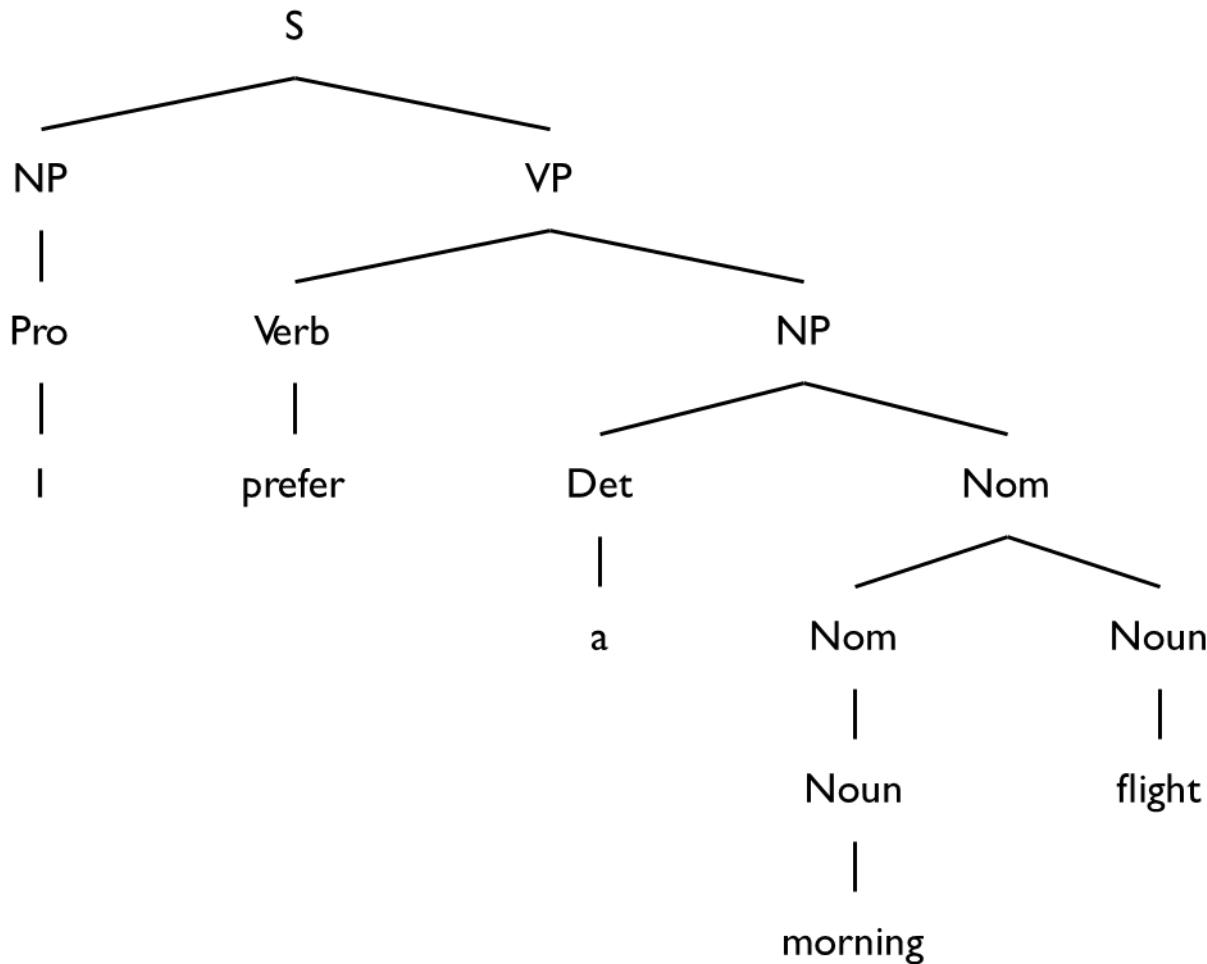
1. Starting unit: words are given a category (part-of-speech)
2. Combining words into **phrases with categories**
3. Combining phrases into **bigger phrases** recursively

I, prefer, a, morning, flight



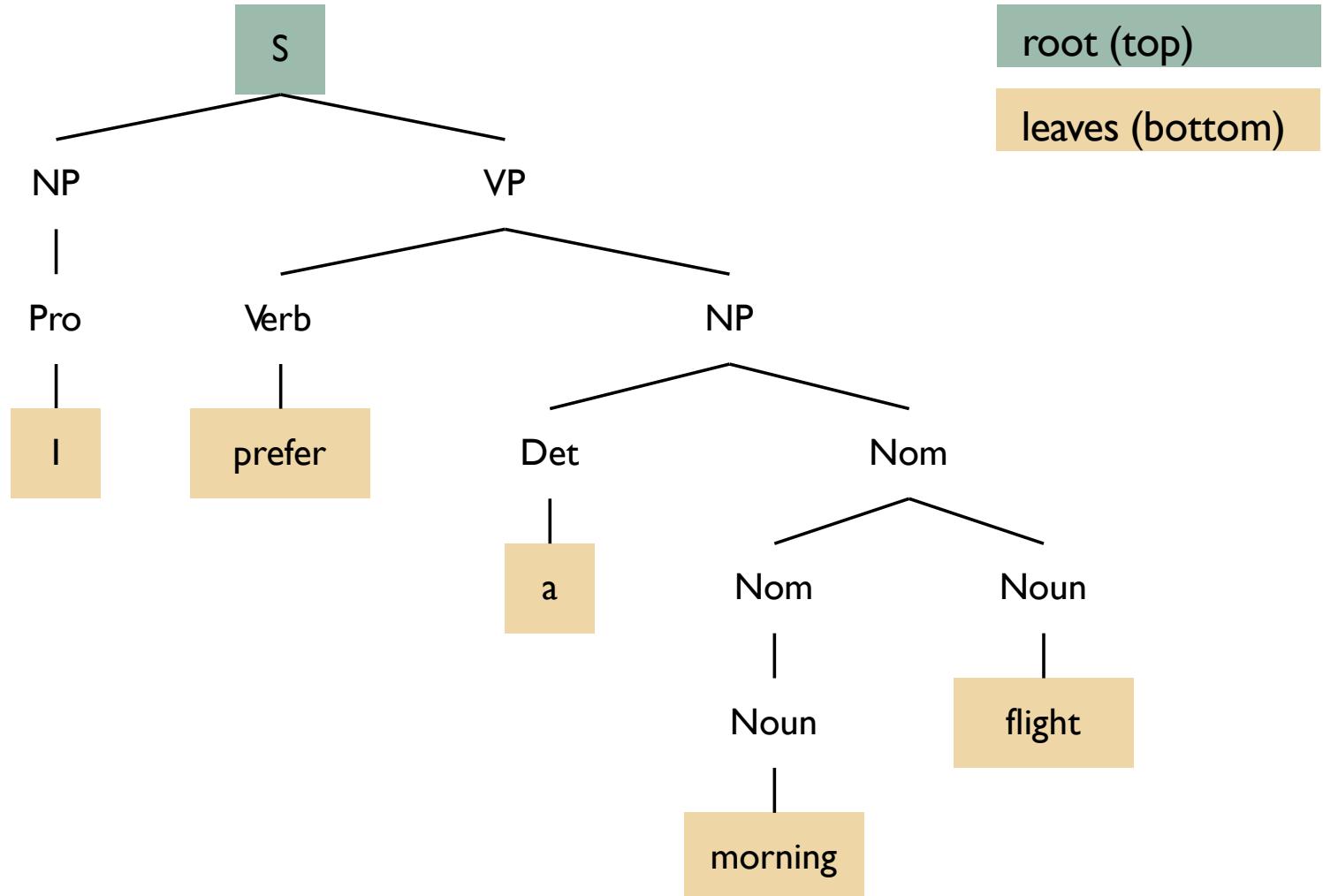
1 Linguistic Structure

A sample context-free grammar



1 Linguistic Structure

A sample context-free grammar



1 Linguistic Structure

Treebanks

Corpora where each sentence is annotated with a parse tree

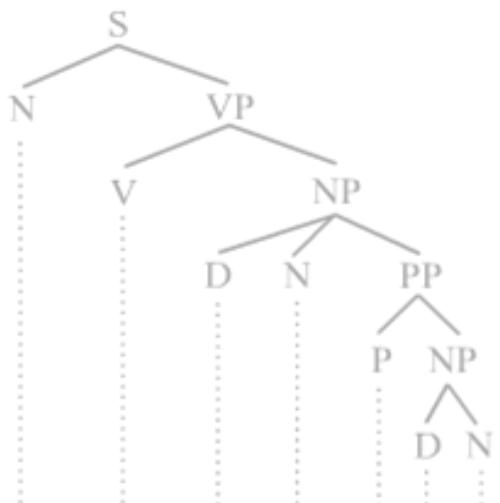
- Treebanks are generally created by
 - parsing texts with an existing parser
 - having human annotators correct the result
- This requires detailed annotation guidelines for annotating different grammatical constructions
- Penn Treebank is a popular treebank for English (Wall Street Journal section)

```
( (S
  (NP-SBJ
    (NP (NNP Pierre) (NNP Vinken) )
    (, ,)
    (ADJP
      (NP (CD 61) (NNS years) )
      (JJ old) )
    (, ,) )
  (VP (MD will)
    (VP (VB join)
      (NP (DT the) (NN board) )
      (PP-CLR (IN as)
        (NP (DT a) (JJ nonexecutive) (NN director) ))
      (NP-TMP (NNP Nov.) (CD 29) )))
    (. .) ))
```

1 Linguistic Structure

Two main views of linguistic structure

Constituency Parsing

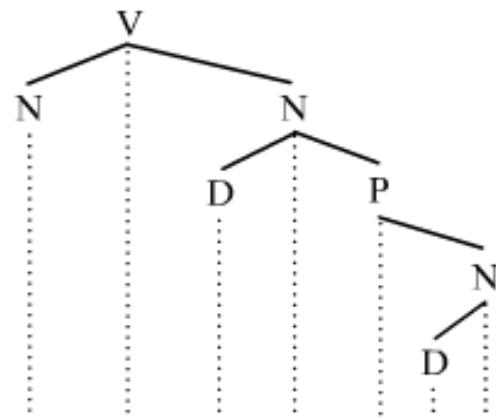


a. They killed the man with a gun.

Constituency grammars

One-to-one-or-more correspondence. For every word in a sentence, there is at least one node in the syntactic structure that corresponds to that word.

Dependency Parsing



b. They killed the man with a gun.

Dependency grammars

one-to-one relation; for every word in the sentence, there is exactly one node in the syntactic structure that corresponds to that word

0 LECTURE PLAN

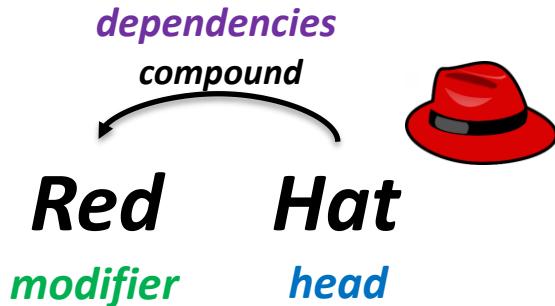
Lecture 7: Parsing

1. Linguistic Structure
2. **Dependency Structure**
3. Dependency Parsing Algorithms
4. Transition-based Dependency Parsing
5. Deep Learning-based Dependency Parsing

2 Dependency Structure

Dependency Structure

Syntactic structure: lexical items linked by binary asymmetrical relations (“arrows”) called **dependencies**



Red – **modifier**, dependent, child, subordinate

Hat - **head**, governor, parent, regent

Compound – **dependency relations** (e.g. subject, prepositional object, etc)

***Head** determines the syntactic/semantic category of the construct

*The arrows are commonly typed with the name of **grammatical relations**

2 Dependency Structure

Dependency Parsing

Represents Lexical/syntactic dependencies between words

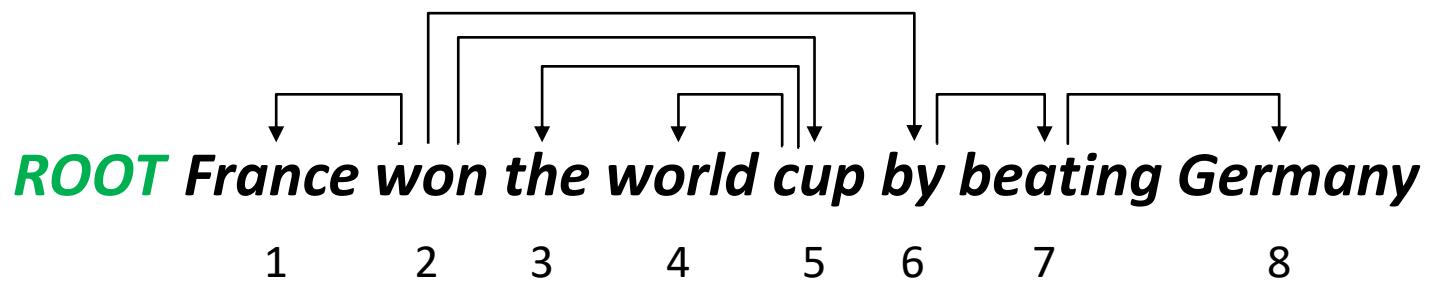
- A sentence is parsed by choosing for each word what other word (including ROOT) is it a dependent of

Dependencies form a tree (connected, acyclic, single-head)

- ***How to make the dependencies a tree - Constraints***

Only one word is a dependent of ROOT (the main predicate of a sentence)

- Don't want cycles $A \rightarrow B, B \rightarrow A$



Dependency Structure

Dependency Grammar/Parsing History

Panini's grammar (4th century BCE)

The notion of dependencies between grammatical units

Ibn Maṭā' (12th century)

The first grammarian to use the term dependency in the grammatical sense

Sámuel Brassai, Franz Kern, Heimann Hariton Tiktin (1800 - 1930)

The dependency seems to have coexisted side by side with that of phrase structure

Lucien Tesnière (1959)

Was dominant approach in “East” in 20th Century (Russia, China, ...)

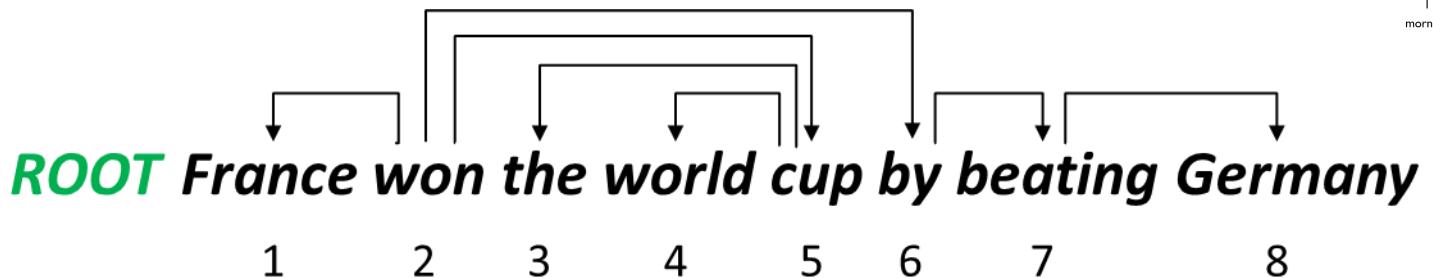
Good for free-er word order languages

David Hays (1962)

The great development surrounding dependency-based theories has come from computational linguistics

2 Dependency Structure

Dependency Grammar/Parsing



Some people draw the arrows one way; some the other way!

- *Tesnière had them point from head to dependent...*

Usually add a fake ROOT so every word is a dependent of precisely 1 other node

Projectivity vs Non-Projectivity

- There are no crossing dependency arcs when the words are laid out in their linear order, with all arcs above the words
- Dependencies parallel to a CFG tree must be projective
 - Forming dependencies by taking 1 child of each category as head
- But dependency theory normally does allow non-projective structures to account for displaced constituents

2 Dependency Structure

Dependency Grammar/Parsing

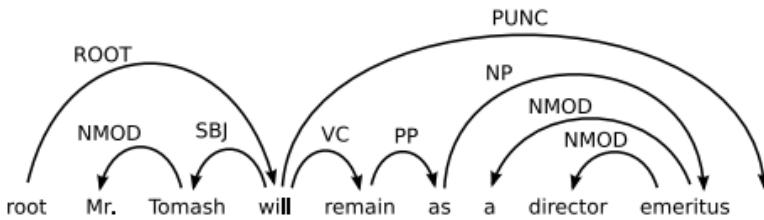


Figure 1: A projective dependency graph.

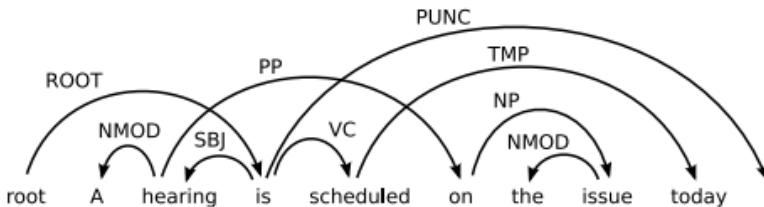


Figure 2: Non-projective dependency graph.

Projectivity vs Non-Projectivity

- There are no crossing dependency arcs when the words are laid out in their linear order, with all arcs above the words
- Dependencies parallel to a CFG tree must be projective
 - Forming dependencies by taking 1 child of each category as head
- But dependency theory normally does allow non-projective structures to account for displaced constituents

2 Dependency Structure

Dependency Relations

- *The following list shows the 37 universal **syntactic relations** used in Universal Dependencies v2.*

- `acl`: clausal modifier of noun (adjectival clause)
- `advcl`: adverbial clause modifier
- `advmod`: adverbial modifier
- `amod`: adjectival modifier
- `appos`: appositional modifier
- `aux`: auxiliary
- `case`: case marking
- `cc`: coordinating conjunction
- `ccomp`: clausal complement
- `clf`: classifier
- `compound`: compound
- `conj`: conjunct
- `cop`: copula
- `csubj`: clausal subject
- `dep`: unspecified dependency
- `det`: determiner
- `discourse`: discourse element
- `dislocated`: dislocated elements
- `expl`: expletive
- `fixed`: fixed multiword expression
- `flat`: flat multiword expression
- `goeswith`: goes with
- `iobj`: indirect object
- `list`: list
- `mark`: marker
- `nmod`: nominal modifier
- `nsubj`: nominal subject
- `nummod`: numeric modifier
- `obj`: object
- `obl`: oblique nominal
- `orphan`: orphan
- `parataxis`: parataxis
- `punct`: punctuation
- `reparandum`: overridden disfluency
- `root`: root
- `vocative`: vocative
- `xcomp`: open clausal complement

2 Dependency Structure

Dependency Relations with annotations

- The idea of dependency structure goes back a long way
- *[Universal Dependencies: <http://universaldependencies.org/> ;*
- *cf. Marcus et al. 1993, The Penn Treebank, Computational Linguistics]*
- Starting off, building a treebank seems a lot slower and less useful than building a grammar
- But a treebank gives us many things
 - Reusability of the labor
 - Many parsers, part-of-speech taggers, etc. can be built on it
 - Valuable resource for linguistics
 - Broad coverage, not just a few intuitions
 - Frequencies and distributional information
 - A way to evaluate systems

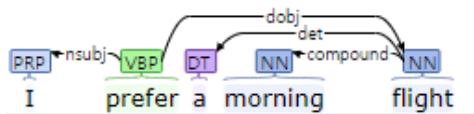
2 Dependency Structure

Dependency Parsing

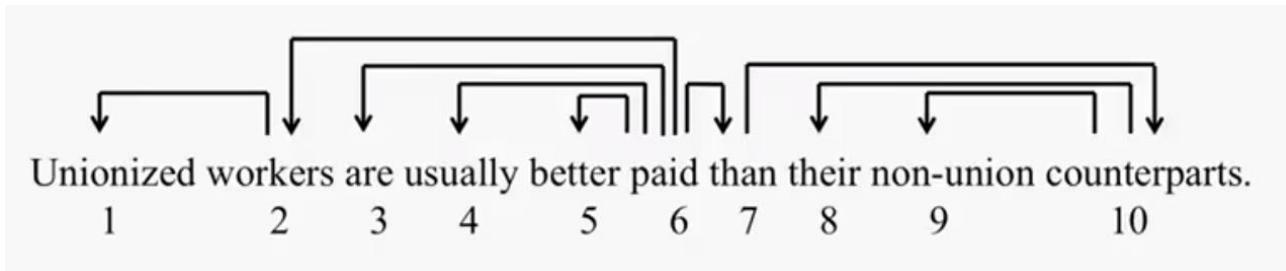
Exercise – Let's do it together!

- Simpler to parse than context-free grammars

ROOT *I prefer a morning flight*



ROOT *Unionised workers are usually better paid than their non-union counterparts*



0 LECTURE PLAN

Lecture 7: Parsing

1. Linguistic Structure
2. Dependency Structure
3. **Dependency Parsing Algorithms**
4. Transition-based Dependency Parsing
5. Deep Learning-based Dependency Parsing

Dependency Parsing Approaches

Methods of Dependency Parsing

- Dynamic programming
Extension of the CYK algorithm to dependency parsing (Eisner, 1996)
- Constraint Satisfaction (Karlsson, 1990)
 $\text{word}(\text{pos}(x)) = \text{DET} \rightarrow (\text{label}(X)=\text{NMOD}, \text{word}(\text{mod}(x))=\text{NN}, \text{pos}(x) < \text{mod}(x))$
A determiner (DET) modifies a noun (NN) on the right with the label NMOD.
- ***Graph-based Dependency Parsing***
Create a ***Maximum Spanning Tree*** for a sentence
McDonald et al.'s (2005) MSTParser scores dependencies independently using a Machine Learning classifier
- ***Transition-based Dependency Parsing (Nivre 2008)***
- ***Neural Dependency Parsing***

Dependency Parsing Approaches

Graph-based dependency parsers

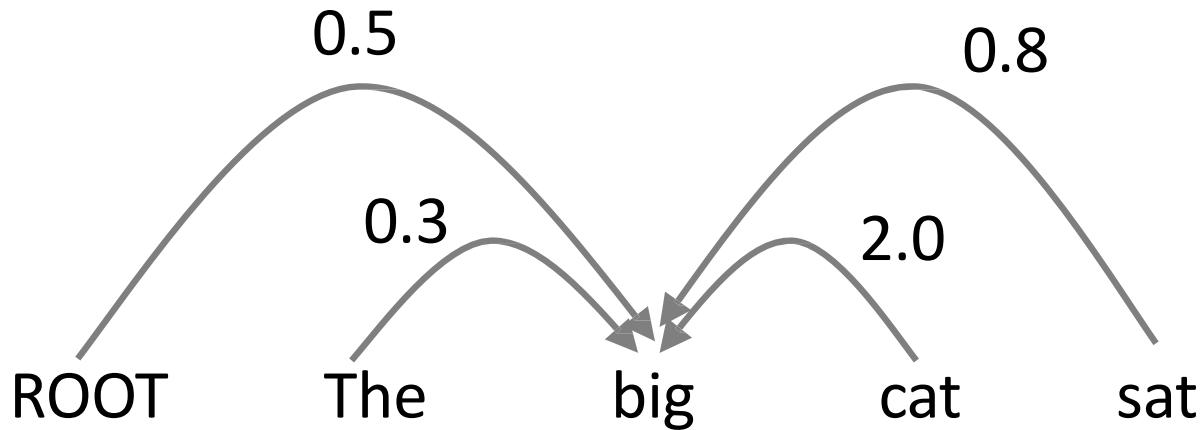
MST Parser (McDonald et al., 2005)

- **Projectivity**
 - English dependency trees are mostly projective (can be drawn without crossing dependencies)
 - Other languages are not
- **Idea**
 - Dependency parsing is equivalent to search for a maximum spanning tree in a directed graph
 - Chu and Liu (1965) and Edmonds (1967) give an efficient algorithm for finding MST for directed graphs

Dependency Parsing Approaches

Graph-based dependency parsers

- Compute a score for every possible dependency for each edge

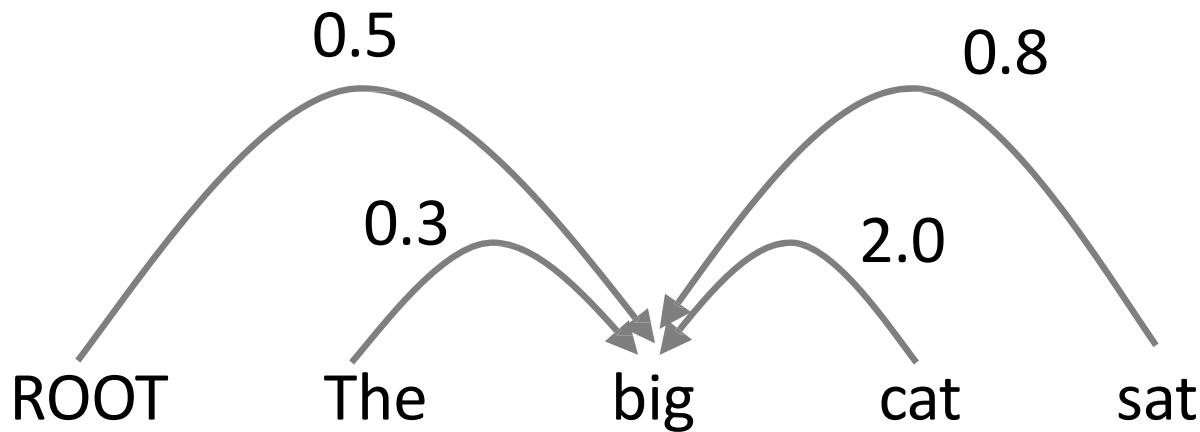


e.g., picking the head for “big”

Dependency Parsing Approaches

Graph-based dependency parsers

- Compute a score for every possible dependency for each edge
 - Then add an edge from each word to its highest-scoring candidate head
 - And repeat the same process for each other word

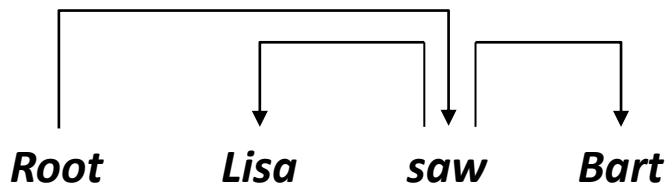
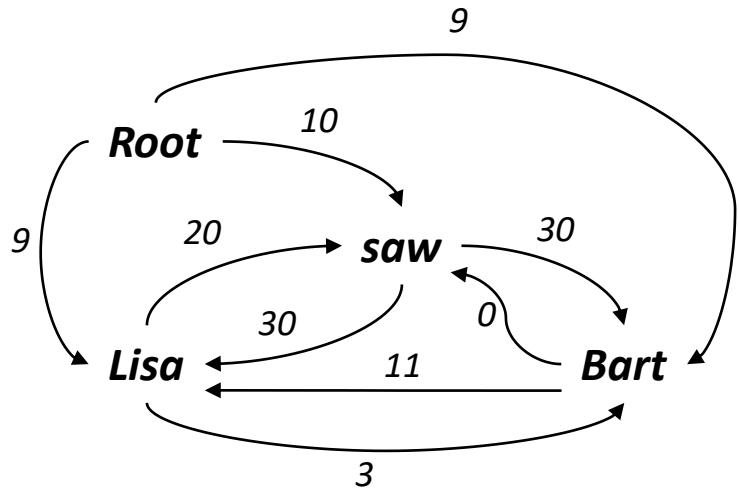


e.g., picking the head for “big”

Dependency Parsing Approaches

Graph-based dependency parsers

- Consider the sentence “Lisa saw Bart”



Dependency Parsing Approaches

Methods of Dependency Parsing

Graph-based Dependency Parsing

- Build a complete graph with directed/weighted edges
- Find the highest scoring tree from a complete dependency graph

greedy algorithm *Transition-based Dependency Parsing*

- Build a tree by applying a sequence of transition actions
- Find the highest scoring action sequence that builds a legal tree

0 LECTURE PLAN

Lecture 7: Parsing

1. Linguistic Structure
2. Dependency Structure
3. Dependency Parsing Algorithms
4. **Transition-based Dependency Parsing**
5. Deep Learning-based Dependency Parsing

Transition-based Parsing

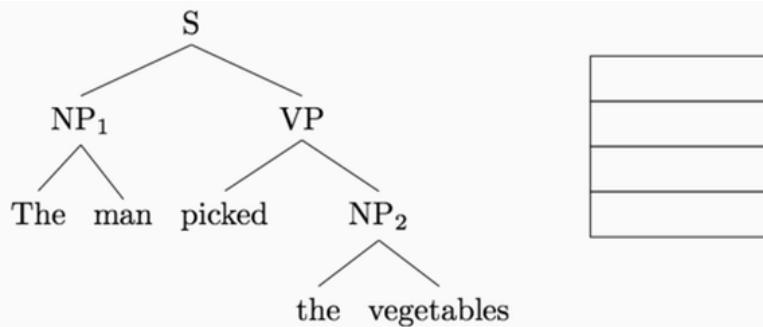
Greedy transition-based parsing (Nivre 2008)

- A simple form of greedy discriminative dependency parser
- **Transition:** an operation that searches for a dependency relation between each pair of words (e.g. Left-Arc, Shift, etc.)
- Design a dumber but really fast algorithm and let the machine learning(deep learning) do the rest.
- Eisner's algorithm (Dynamic Programming-based Dependency Parsing) searches over many different dependency trees at the same time.
- A transition-based dependency parser only builds one tree, in one left-to-right sweep over the input

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

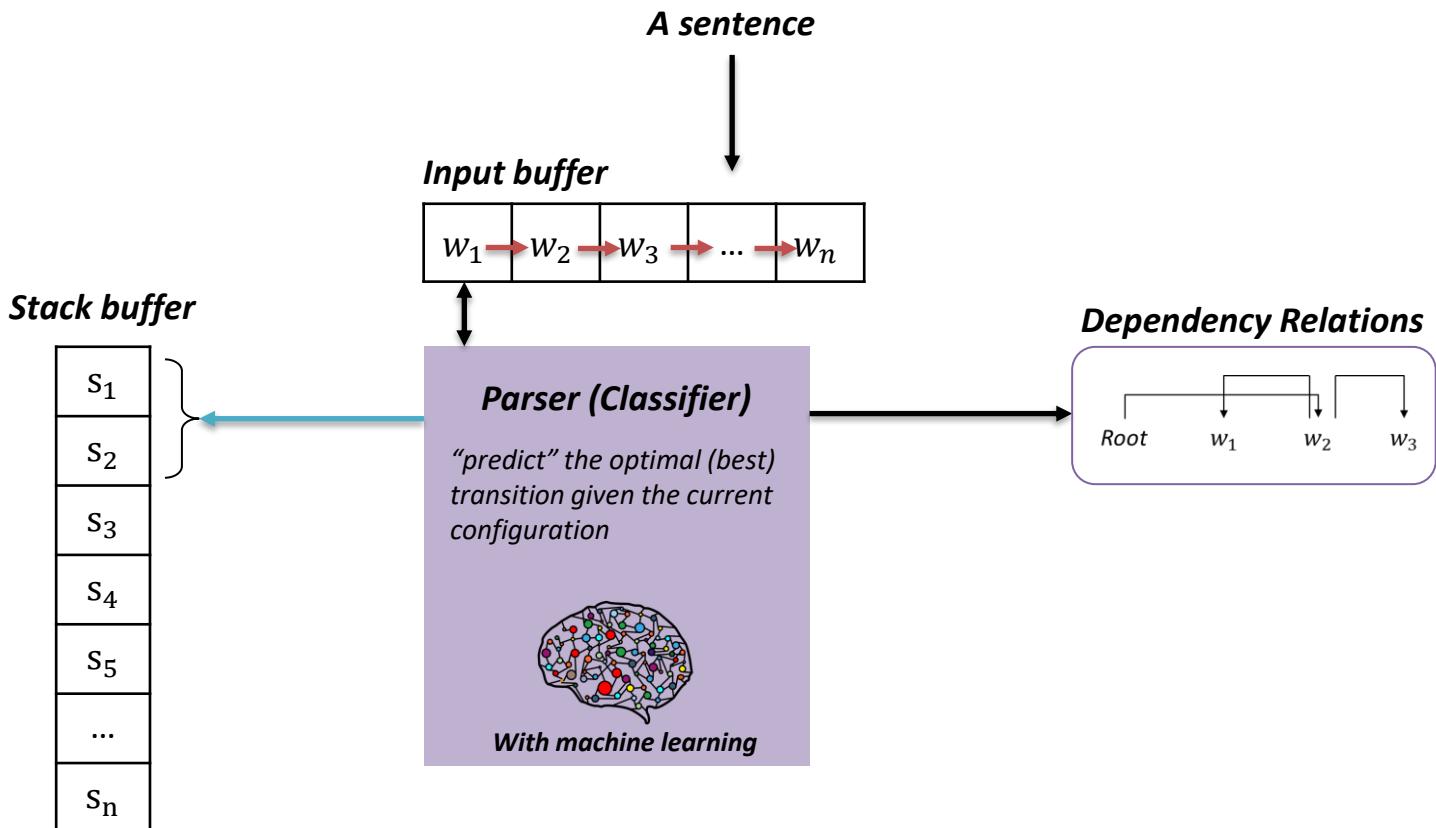
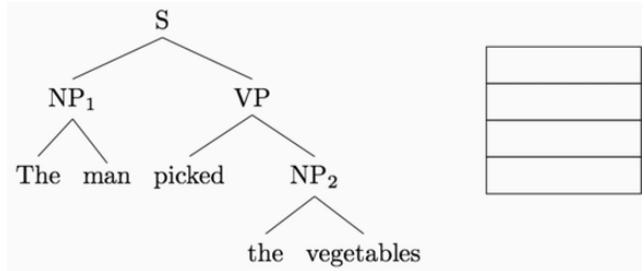
- A sequence of bottom up actions
 - Roughly like “shift” or “reduce” in a shift-reduce parser, but the “reduce” actions are specialized to create dependencies with head on left or right



- It is implemented in most practical transition- based dependency parsers, including **MaltParser**. The arc-standard algorithm is a simple algorithm for transition-based dependency parsing.

Transition-based Parsing

Transition-based parsing



Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

Stack



Buffer



Dependency Graph



Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

Stack

ROOT

Buffer

book

me

a

morning

flight

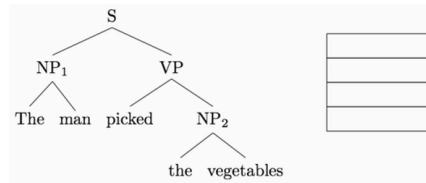
Dependency Graph



- ***Initial configuration:***

- All words are in the buffer.
- The stack is empty or starts with the ROOT symbol
- The dependency graph is empty.

Transition-based Parsing



Transition-based parsing – The arc-standard algorithm

Stack

ROOT

Buffer

book

me

a

morning

flight

Dependency Graph



Possible Transition

Shift

- *Push the next word in buffer onto the stack*

Left-Arc

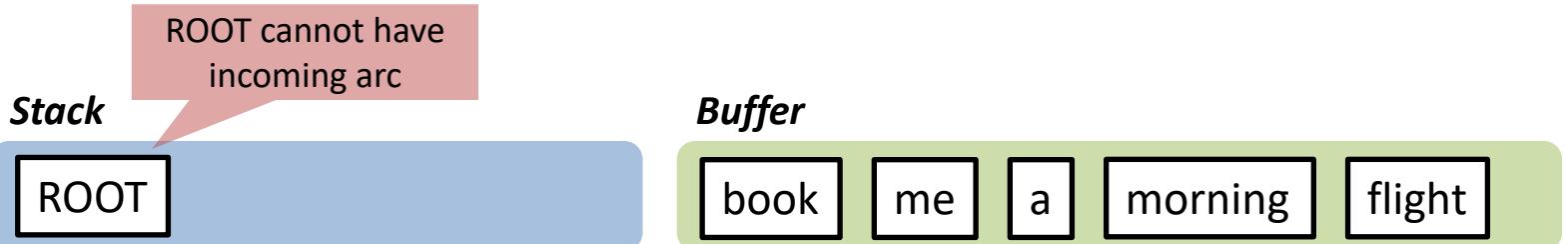
- *Add an arc from the topmost word to the 2nd-topmost word on the stack*
- *Remove 2nd word from stack*

Right-Arc

- *Add an arc from the 2nd-topmost word to the topmost word on the stack*
- *Remove the topmost word from stack*

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm



Dependency Graph

Possible Transition

Shift

- Push the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

Left Arc and Right Arc require 2 elements in stack to be applied

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

Stack

ROOT

Buffer

book

me

a

morning

flight

Dependency Graph



Possible Transition

Shift

- Push the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

Stack

ROOT

book

Buffer

me

a

morning

flight

Dependency Graph



Possible Transition

Shift

- Push the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

Stack

ROOT book me

Buffer

a morning flight

Dependency Graph



Possible Transition

Shift

- Push the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

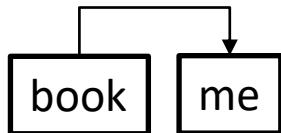
Stack

ROOT book me

Buffer

a morning flight

Dependency Graph



Possible Transition

Shift

- Push the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

Stack

ROOT

book

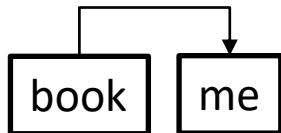
a

Buffer

morning

flight

Dependency Graph



Possible Transition

Shift

- Push the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

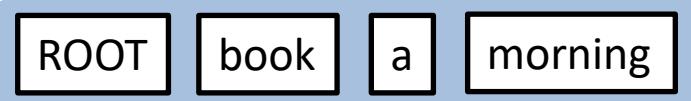
Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

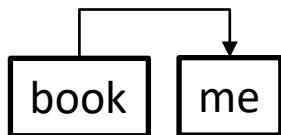
Stack



Buffer



Dependency Graph



Possible Transition

Shift

- Push the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

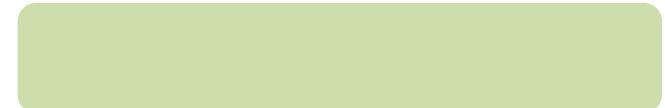
Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

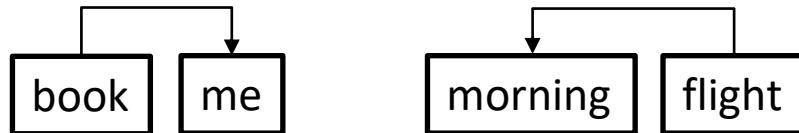
Stack



Buffer



Dependency Graph



Possible Transition

Shift

- *Push* the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

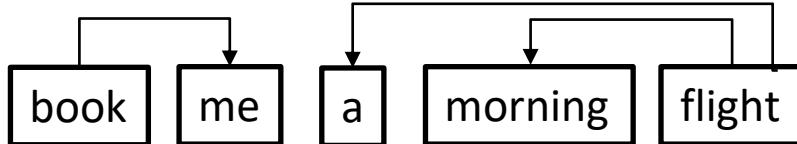
Transition-based parsing – The arc-standard algorithm

Stack

ROOT book a flight

Buffer

Dependency Graph



Possible Transition

Shift

- Push the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

Stack

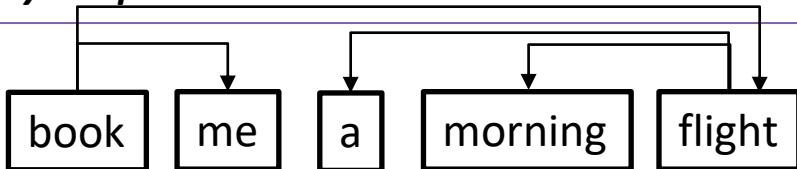
ROOT

book

flight

Buffer

Dependency Graph



Possible Transition

Shift

- *Push* the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

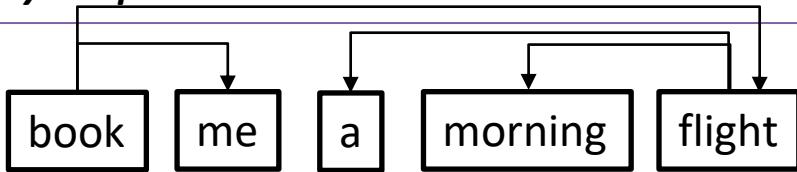
Stack

ROOT

book

Buffer

Dependency Graph



Possible Transition

Shift

- Push the next word in buffer onto the stack

Left-Arc

- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

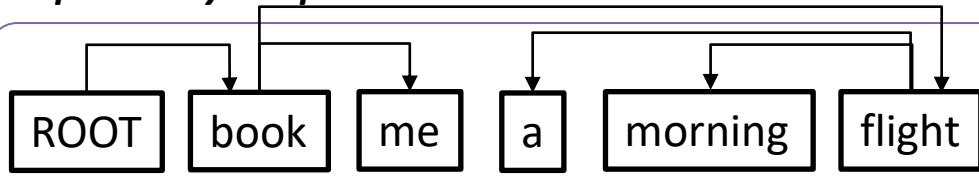
Stack

ROOT

book

Buffer

Dependency Graph



Possible Transition

Shift

- Push the next word in buffer onto the stack

Left-Arc

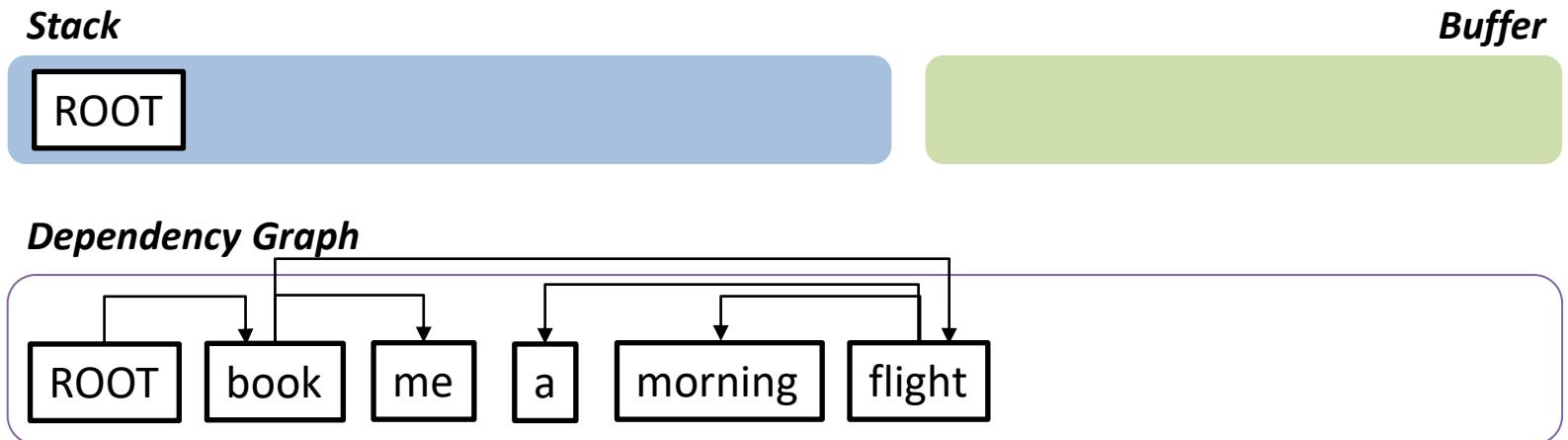
- Add an arc from the topmost word to the 2nd-topmost word on the stack
- Remove 2nd word from stack

Right-Arc

- Add an arc from the 2nd-topmost word to the topmost word on the stack
- Remove the topmost word from stack

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm



- **Terminal configuration:**
 - The buffer is empty.
 - The stack contains a single word.

Transition-based Parsing

Transition-based parsing



- (a) Arc-standard: *is* and *example* are eligible for arcs.



- (b) Arc-eager: *example* and *with* are eligible for arcs.



- (c) Easy-first: All unreduced tokens are active (bolded).

Transition-based Parsing

Transition-based parsing – The arc-standard algorithm

Start: $\sigma = [\text{ROOT}], \beta = w_1, \dots, w_n, A = \emptyset$

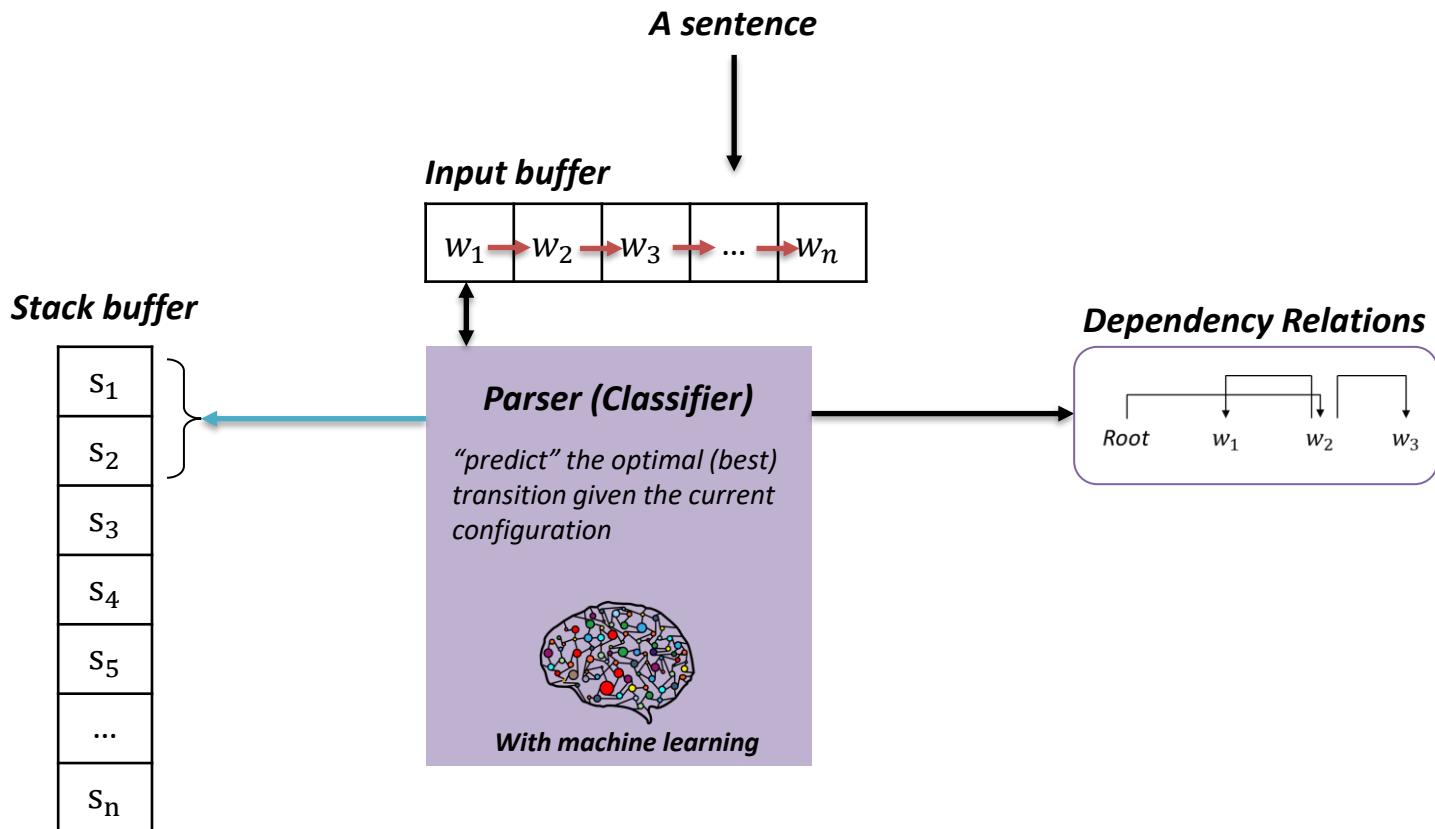
1. Shift $\sigma, w_i | \beta, A \rightarrow \sigma | w_i, \beta, A$
2. Left-Arc_r $\sigma | w_i | w_j, \beta, A \rightarrow \sigma | w_j, \beta, A \cup \{r(w_j, w_i)\}$
3. Right-Arc_r $\sigma | w_i | w_j, \beta, A \rightarrow \sigma | w_i, \beta, A \cup \{r(w_i, w_j)\}$

Finish: $\sigma = [w], \beta = \emptyset$

How to choose the next action?

Transition-based Parsing

Transition-based parsing



Transition-based Parsing

How to choose the next action?

Stand back, You know machine learning!

Goal: Predict the next transition (class), given the current configuration.

- We let the parser run on gold-standard trees.
- Every time there is a choice to make, we simply look into the tree and do ‘the right thing’.
- We collect all (configuration, transition) pairs and train a classifier on them.
- When parsing unseen sentences, we use the trained classifier as a guide.

What if the number of pairs is far too large?

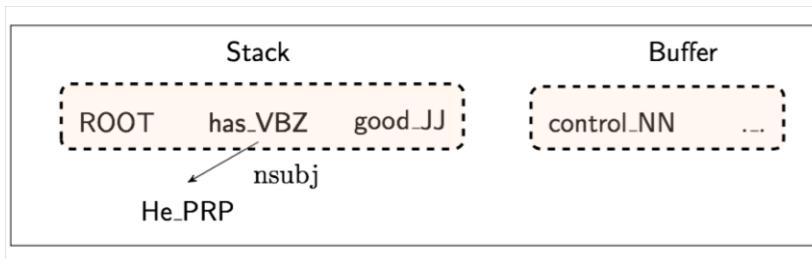
Transition-based Parsing

Feature Representation

- Define a set of features of configurations that you consider to be relevant for the task of predicting the next transition.

Example: word forms of the topmost two words on the stack and the next two words in the buffer

- Describe every configuration in terms of a feature vector.



- In practical systems, we have thousands of features and hundreds of transitions.
- There are several machine-learning paradigms that can be used to train a guide for such a task
- Examples: perceptron, decision trees, support-vector machines, memory-based learning

Transition-based Parsing

Evaluation of Dependency Parsing

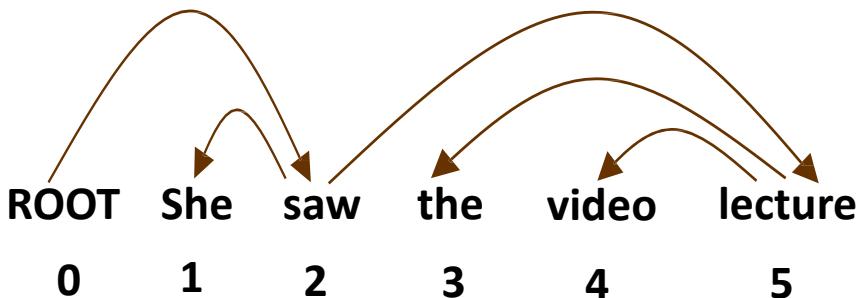
$$\text{Accuracy} = \frac{\# \text{ correct deps}}{\# \text{ of deps}}$$

Unlabeled attachment score (UAS) = head

Labeled attachment score (LAS) = head and label

Transition-based Parsing

Evaluation of Dependency Parsing



Gold Standard

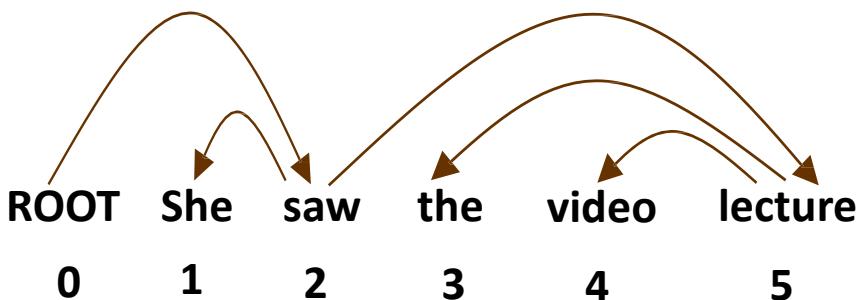
1	2	she	nsubj
2	0	saw	root
3	5	the	det
4	5	video	nn
5	2	lecture	obj

Parsed (assume this is what you classified)

1	2	she	nsubj
2	0	saw	root
3	4	the	det
4	5	video	nn
5	2	lecture	ccomp

Transition-based Parsing

Evaluation of Dependency Parsing



Gold Standard

1	2	she	nsubj
2	0	saw	root
3	5	the	det
4	5	video	nn
5	2	lecture	obj

Parsed (assume this is what you classified)

1	2	she	nsubj
2	0	saw	root
3	4	the	det
4	5	video	nn
5	2	lecture	ccomp

$$\text{Unlabeled attachment score (UAS)} = 4 / 5 = 80\%$$

$$\text{Labeled attachment score (LAS)} = 2 / 5 = 40\%$$

0 LECTURE PLAN

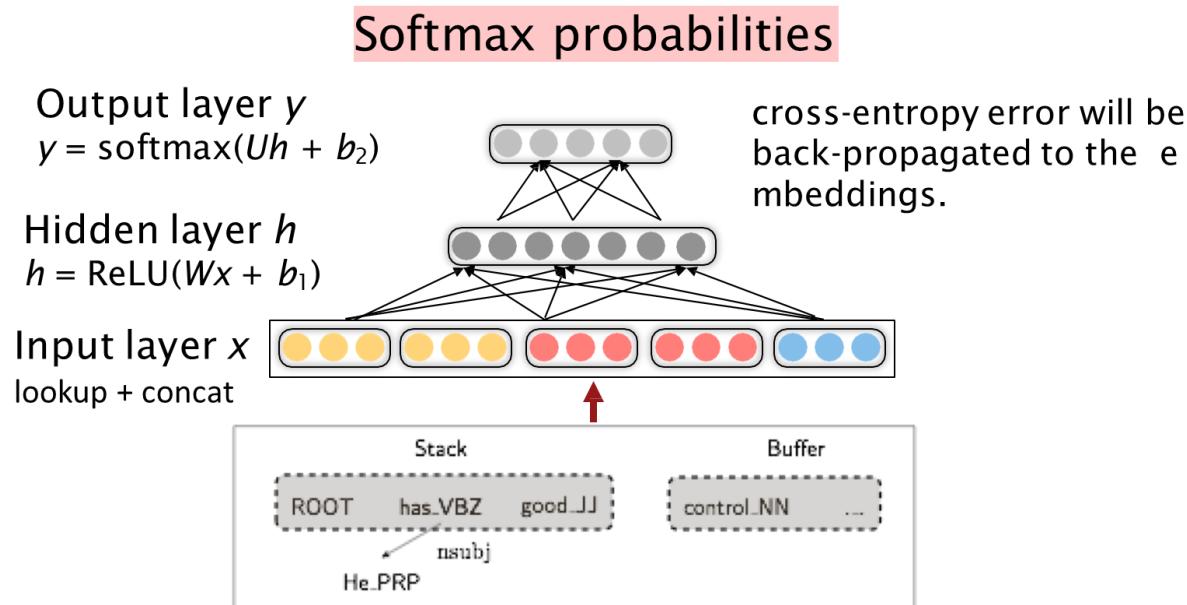
Lecture 7: Parsing

1. Linguistic Structure
2. Dependency Structure
3. Dependency Parsing Algorithms
4. Transition-based Dependency Parsing
5. Deep Learning-based Dependency Parsing

Distributed Representations

- Represent each word as a d-dimensional dense vector (i.e., word embedding)
 - Similar words are expected to have close vectors.
 - NNS (plural noun) should be close to NN (singular noun).
- Meanwhile, part-of-speech tags (POS) and dependency labels are also represented as d-dimensional vectors.
- The smaller discrete sets also exhibit many semantical similarities

Neural Dependency Parsing (Chen & Manning, 2014)



Neural Dependency Parsing

*Accuracy and parsing speed
on PTB + Stanford dependencies.*

Parser	Dev		Test		Speed (sent/s)
	UAS	LAS	UAS	LAS	
standard	90.2	87.8	89.4	87.3	26
eager	89.8	87.4	89.6	87.4	34
Malt:sp	89.8	87.2	89.3	86.9	469
Malt:eager	89.6	86.9	89.4	86.8	448
MSTParser	91.4	88.1	90.7	87.6	10
Our parser	92.0	89.7	91.8	89.6	654

Accuracy and parsing speed on CTB

Parser	Dev		Test		Speed (sent/s)
	UAS	LAS	UAS	LAS	
standard	82.4	80.9	82.7	81.2	72
eager	81.1	79.7	80.3	78.7	80
Malt:sp	82.4	80.5	82.4	80.6	420
Malt:eager	81.2	79.3	80.2	78.4	393
MSTParser	84.0	82.1	83.0	81.2	6
Our parser	84.0	82.4	83.9	82.4	936

Chen, D., & Manning, C. (2014). A fast and accurate dependency parser using neural networks. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 740-750).

- PTB: English Penn Treebank
- CTB: Chinese Penn Treebank

5

Deep Learning-based Parsing

Dependency Parsing Trends – Penn Treebank

RANK	MODEL	LAS ↑	UAS	POS	PAPER	CODE	RESULT	YEAR
1	Label Attention Layer + HPSG + XLNet	96.26	97.42	97.3	Rethinking Self-Attention: Towards Interpretability in Neural Parsing			2019
2	Deep Biaffine + RoBERTa	95.83	97.29		Deep Biaffine Attention for Neural Dependency Parsing			2016
3	HPSG Parser (Joint)	95.72	97.20	97.3	Head-Driven Phrase Structure Grammar Parsing on Penn Treebank			2019
4	ACE	95.7	97.2		Automated Concatenation of Embeddings for Structured Prediction			2020
5	MFVI	95.34	96.91		Second-Order Neural Dependency Parsing with Message Passing and End-to-End Training			2020
6	CVT + Multi-Task + Large	95.02	96.61		Semi-Supervised Sequence Modeling with Cross-View Training			2018
7	CVT + Multi-Task	94.83	96.44		Semi-Supervised Sequence Modeling with Cross-View Training			2018
8	SpanRel	94.7			Generalizing Natural Language Analysis through Span-relation Representations			2019
9	CRFPar	94.49	96.14		Efficient Second-Order TreeCRF for Neural Dependency Parsing			2020
10	Left-to-Right Pointer Network	94.43	96.04	97.3	Left-to-Right Dependency Parsing with Pointer Networks			2019

/ Final Exercise

VB PRP

Thank you!

VBP DT JJ NN

Have a great day!

/ Reference

Reference for this lecture

- Deng, L., & Liu, Y. (Eds.). (2018). Deep Learning in Natural Language Processing. Springer.
- Rao, D., & McMahan, B. (2019). Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. " O'Reilly Media, Inc.".
- Manning, C. D., Manning, C. D., & Schütze, H. (1999). Foundations of statistical natural language processing. MIT press.
- Nivre, J (2016). Transition-based dependency parsing, lecture notes, Uppsala Universitet
- Manning, C 2017, Natural Language Processing with Deep Learning, lecture notes, Stanford University
- Chen, D., & Manning, C. (2014). A fast and accurate dependency parser using neural networks. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 740-750).
- Eisner, J. M. (1996, August). Three new probabilistic models for dependency parsing: An exploration. In Proceedings of the 16th conference on Computational linguistics-Volume 1 (pp. 340-345). Association for Computational Linguistics.

COMP5046

Natural Language Processing

*Lecture 8: Language Model and
Natural Language Generation*

Dr. Caren Han

Semester 1, 2022

School of Computer Science,
University of Sydney



0 The course topics

What will you learn in this course?

Week 1: Introduction to Natural Language Processing (NLP)

Week 2: Word Embeddings (Word Vector for Meaning)

Week 3: Word Classification with Machine Learning I

Week 4: Word Classification with Machine Learning II

NLP and
Machine
Learning

Week 5: Language Fundamental

Week 6: Part of Speech Tagging

Week 7: Dependency Parsing

Week 8: Language Model and Natural Language Generation

NLP
Techniques

Week 9: Information Extraction: Named Entity Recognition

Week 10: Advanced NLP: Attention and Reading Comprehension

Week 11: Advanced NLP: Transformer and Machine Translation

Week 12: Advanced NLP: Pretrained Model in NLP

Advanced
Topic

Week 13: Future of NLP and Exam Review

0 LECTURE PLAN

Lecture 8: Language Model and Natural Language Generation

1. Language Model
2. Traditional Language Model
3. Neural Language Model
4. Natural Language Generation
5. NLG Tasks
6. Language Model and NLG Evaluation

1 Language Model

What is Language Model

- is the task of predicting what word comes next based on the given words.
- is a probabilistic model which predicts the probability that a sequence of tokens belongs to a language.

Can you come _____



$x^{(1)}, x^{(2)}, \dots, x^{(t)}$ x^1, x^2, x^3

Given a sequence of words, ***Can, you, come***, compute the probability distribution of the next word.

$\boldsymbol{x}^{(t+1)}$ (*can be any word in the vocabulary*)

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

1 Language Model

What is Language Model

- is the task of predicting what word comes next based on the given words.
- is a probabilistic model which predicts the probability that a sequence of tokens belongs to a language.

Can you come here

Can you come there



$P(\text{Can}, \text{you}, \text{come}, \text{here})$ vs $P(\text{Can}, \text{you}, \text{come}, \text{there})$

$P(\text{here}|\text{can}, \text{you}, \text{come})$ vs $P(\text{there}|\text{can}, \text{you}, \text{come})$

$$P(\mathbf{x}^{(t+1)} | \mathbf{x}^{(t)}, \dots, \mathbf{x}^{(1)})$$

1 Language Model

What is Language Model

- is the task of predicting what word comes next based on the given words.
- is a probabilistic model which predicts the probability that a sequence of tokens belongs to a language.

Can you come here

Can you come there



$P(\text{Can}, \text{you}, \text{come}, \text{here})$ vs $P(\text{Can}, \text{you}, \text{come}, \text{there})$

$P(\text{here}|\text{can, you, come})$ vs $P(\text{there}|\text{can, you, come})$

$$P(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)}) = P(\mathbf{x}^{(1)}) \times P(\mathbf{x}^{(2)} | \mathbf{x}^{(1)}) \times \dots \times P(\mathbf{x}^{(T)} | \mathbf{x}^{(T-1)}, \dots, \mathbf{x}^{(1)})$$

$$= \prod_{t=1}^T P(\mathbf{x}^{(t)} | \mathbf{x}^{(t-1)}, \dots, \mathbf{x}^{(1)})$$

↑

Conditional Probability

1 Language Model

Language Modeling in NLP

- The probabilities returned by a language model are mostly useful to compare the likelihood that different sentences are "good sentences". Useful in many practical tasks, for example:

Spell correction/Automatic Speech Recognition

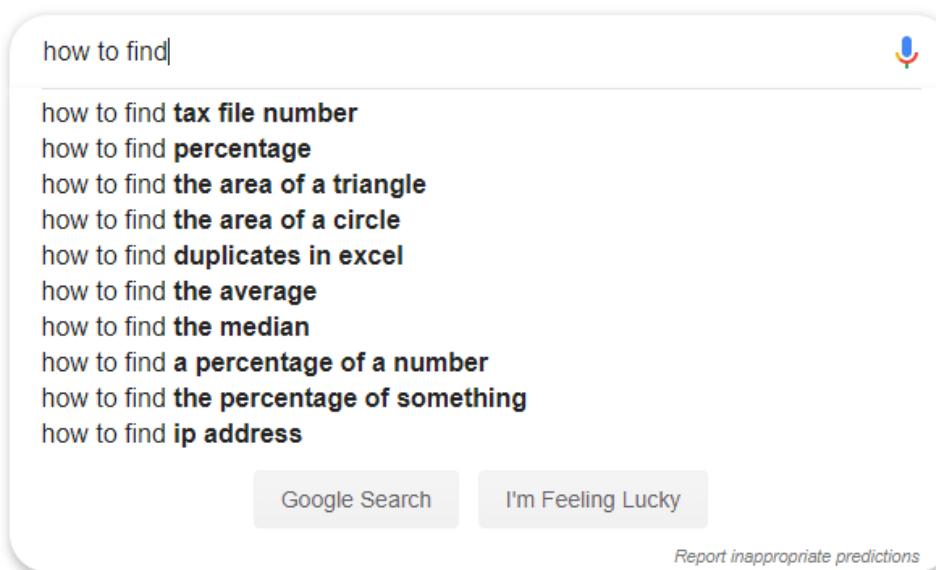
- I would like to read that **book** *Closest words= [book, boog, boat, ...]*

Natural Language Generation

- Dialogue (chit chat and task-based)
- Abstractive Summarisation
- Machine Translation
- Creative Writing: Story Telling, ...

1 Language Model

Do we use Language Model?



how to find|

how to find **tax file number**
how to find **percentage**
how to find **the area of a triangle**
how to find **the area of a circle**
how to find **duplicates in excel**
how to find **the average**
how to find **the median**
how to find **a percentage of a number**
how to find **the percentage of something**
how to find **ip address**

Google Search I'm Feeling Lucky

Report inappropriate predictions

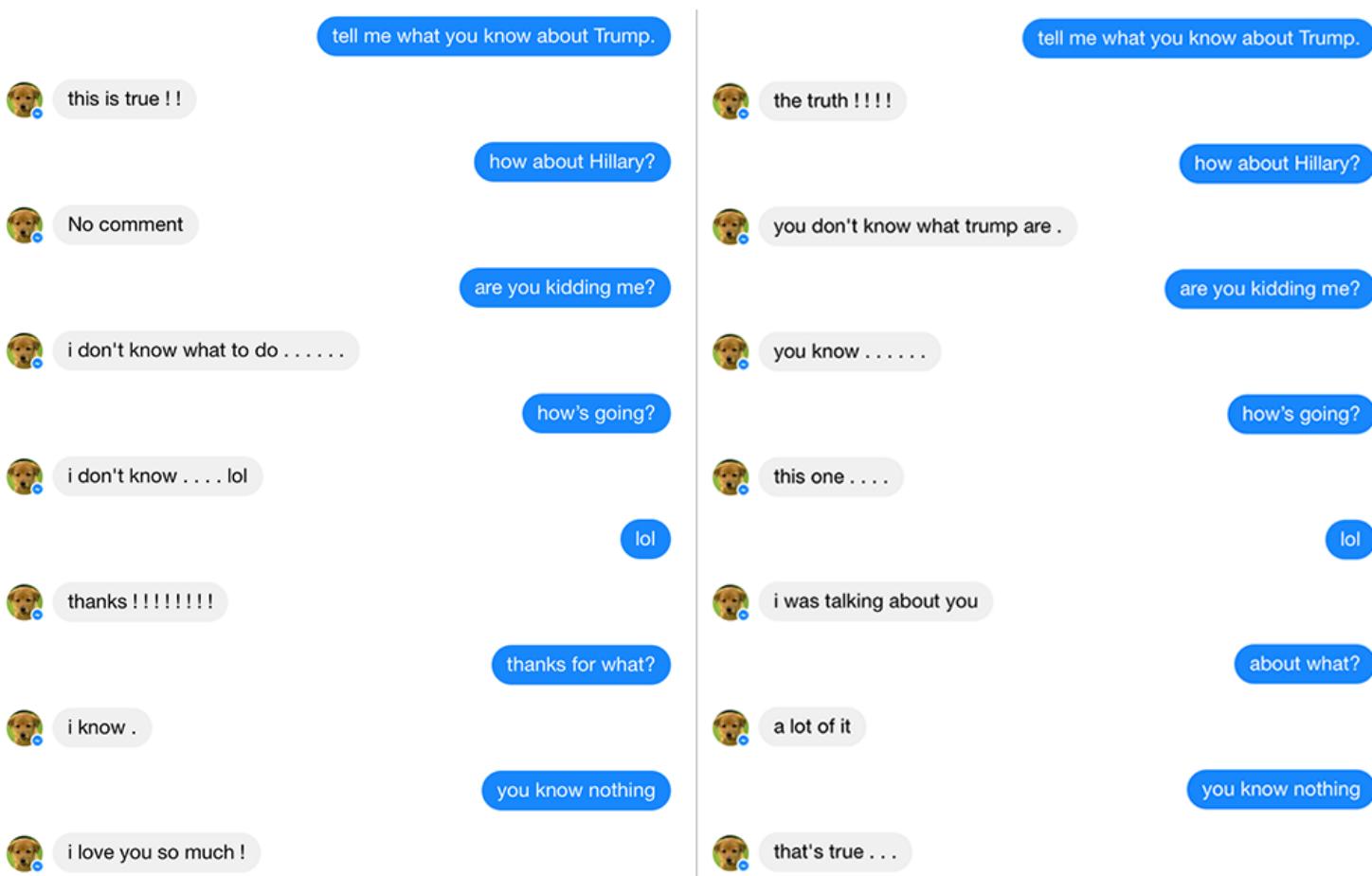
1 Language Model

Yes but sometimes fail..



1 Language Model

Language Model in Dialog System



Without anti-language model

With anti-language model

1 Language Model

Language Modeling in Natural Language Generation

Conditional Language Modeling: the task of predicting the next word, given the words so far, and also some other input x :

Natural Language Generation Tasks

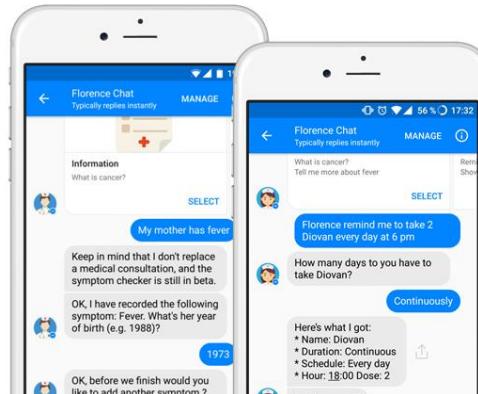
- Dialogue (chit chat and task-based)
 $x=\text{dialogue history}, y=\text{next utterance}$
- Abstractive Summarisation
 $x=\text{input text}, y=\text{summarized text}$
- Machine Translation (in later week)
 $x=\text{source sentence}, y=\text{target sentence}$

1 Language Models

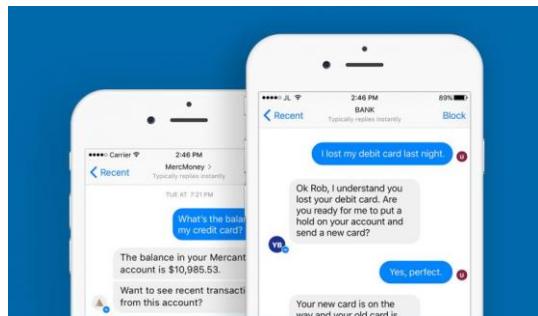
Tips for using Language Model (you already knew!)

It is extremely important to collect and learn the model with the corpus that includes documents about the domain that your system/application will be used.

Medical Documents



Financial Documents



0 LECTURE PLAN

Lecture 8: Language Model and Natural Language Generation

1. Language Model
2. **Traditional Language Model**
3. Neural Language Model
4. Natural Language Generation
5. NLG Tasks
6. Language Model and NLG Evaluation

Statistical Language Model (SLM)

- ***Conditional Language Modeling***: the task of predicting the next word, given the words so far, and also some other input \mathbf{x} :

$$\begin{aligned} P(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)}) &= P(\mathbf{x}^{(1)}) \times P(\mathbf{x}^{(2)} | \mathbf{x}^{(1)}) \times \dots \times P(\mathbf{x}^{(T)} | \mathbf{x}^{(T-1)}, \dots, \mathbf{x}^{(1)}) \\ &= \prod_{t=1}^T P(\mathbf{x}^{(t)} | \mathbf{x}^{(t-1)}, \dots, \mathbf{x}^{(1)}) \end{aligned}$$

“An adorable little boy is spreading smiles”

P(An, adorable, little, boy, is, spreading, smiles)

= $P(\text{An}) \times P(\text{adorable} | \text{An}) \times P(\text{little} | \text{An adorable}) \times P(\text{boy} | \text{An adorable little}) \times P(\text{is} | \text{An adorable little boy}) \times P(\text{spreading} | \text{An adorable little boy is}) \times P(\text{smiles} | \text{An adorable little boy is spreading})$

Statistical Language Model (SLM)

- ***Conditional Language Modeling***: the task of predicting the next word, given the words so far, and also some other input \mathbf{x} :

$$\begin{aligned} P(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)}) &= P(\mathbf{x}^{(1)}) \times P(\mathbf{x}^{(2)} | \mathbf{x}^{(1)}) \times \dots \times P(\mathbf{x}^{(T)} | \mathbf{x}^{(T-1)}, \dots, \mathbf{x}^{(1)}) \\ &= \prod_{t=1}^T P(\mathbf{x}^{(t)} | \mathbf{x}^{(t-1)}, \dots, \mathbf{x}^{(1)}) \end{aligned}$$

“An adorable little boy is”

$P(\text{is} | \text{An adorable little boy})?$

Trained Corpus

An adorable little boy is
..... An adorable little boy is
..... An adorable little boy laughed
.....
.....

Simplest method

$$\begin{aligned} &= \text{Count}(\text{An adorable little boy is}) / \text{Count}(\text{An adorable little boy}) \\ &= 30/100 = 0.3 \end{aligned}$$

Q: What if there is no ‘An adorable little boy is’ phrase in the corpus?

N-gram Language Models

- *An N-gram is a sequence of N words.*
- *An N-gram model predicts the probability of a given N-gram within any sequence of words in the language.*

“An adorable little boy is spreading smiles”

A ***n*-gram** is a chunk of *n* consecutive words.

- **uni**grams : an, adorable, little, boy, is, spreading, smiles
- **bi**grams : an adorable, adorable little, little boy, boy is, is spreading, spreading smiles
- **tri**grams : an adorable little, adorable little boy, little boy is, boy is spreading, is spreading smiles
- **4**-grams : an adorable little boy, adorable little boy is, little boy is spreading, boy is spreading smiles

Traditional Language Models

N-gram Language Models: Exercise

- Assume that we learn a **trigram language model**

"An adorable little boy is spreading ? "

n-1 words only
(3-1) words only

$$P(w|is\ spreading) =$$

$$\text{Count}(is\ spreading\ w) / \text{Count}(is\ spreading)$$

Trained Corpus

boy is spreading smile.....
..... boy is spreading rumours
.....
An adorable little boy is spreading
.....
.....
.....
.....

$$P(rumours|is\ spreading)$$

$$= \text{Count}(is\ spreading\ rumours) / \text{Count}(is\ spreading)$$
$$= 500/1000 = 0.5$$

$$P(smiles|is\ spreading)$$

$$= \text{Count}(is\ spreading\ smiles) / \text{Count}(is\ spreading)$$
$$= 200/1000 = 0.2$$

N-gram Language Models: Beautiful Formula 😊

- *Simplifying assumption: the next word, $x^{(t+1)}$, depends only on the preceding n-1 words.*

$$P(x^{(t+1)} | x^{(t)}, \dots, x^{(1)}) = P(x^{(t+1)} | \overbrace{x^{(t)}, \dots, x^{(t-n+2)}})$$

- *How do we get these n-gram and (n-1)-gram probabilities?
Counting them!*

$$\approx \frac{\text{count}(x^{(t+1)}, x^{(t)}, \dots, x^{(t-n+2)})}{\text{count}(x^{(t)}, \dots, x^{(t-n+2)})}$$

N-gram Language Model Limitation: Trade-off Issue

- Mostly, $n=2$ works better than $n=1$ in n-gram language model
- We learned a **trigram language model**

"An adorable little boy is spreading ?"

$n-1$ words only
(3-1) words only

- Find the optimal n is important! (OOV issue or Model Size issue)

$$P(w | \text{is spreading}) = \\ \frac{\text{Count(is spreading } w)}{\text{Count(is spreading)}}$$

Need to store count for all n-grams that you saw in the corpus.

**If you increase n or corpus,
the model size will be increased!**

N-gram Language Model Limitation: Zero Count Issue

$$P(w | \text{is spreading}) = \\ \text{Count}(\text{is spreading } w) / \text{Count}(\text{is spreading})$$

1. *What if the ‘**is spreading w**’ phrase never occurred in the corpus?*
The probability will be 0.
 - *Alternative solution: Smoothing (Add small δ to the count for every w in the corpus)*

2. *What if the ‘**is spreading**’ phrase never occurred in the corpus?*
*It is impossible to calculate the probability for any **w**.*
 - *Alternative solution: Backoff (Just condition on “spreading” instead)*

N-gram Language Model Demo

Generating text with a n-gram language model

Algorithmia Valentines Generator

Enter your lover's name

GENERATE

Dear Jessica, Love is the secret vault of my sight. There's nothing in this dream I never gave up hope.. I wasn't truly sure why you need me, and that is genuine, it has been inscribed in my heart in your strong arms wrapped around me again because I don't know the way I love you with open arms where you want to say. You are my everything and you're always on my face and even sometimes blinds us, but second chances no matter the consequences, I will always be alive knowing that you are the reason I smile when I always will and there that I get through the hurt in the past few months I have let the fears you are, 7 1/2 years, I need to wait to spend apart near about killed me but more three months alone. So long as we are walking different paths. Two years is not falling in love for you deeper every day is worth living. Nothing can be together. Thank you ...

Algorithmia Valentines Generator

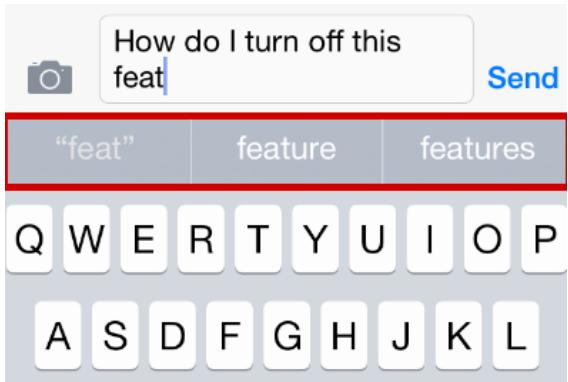
Enter your lover's name

GENERATE

Dear James, I really feel. You captured my heart, I left, but I have had ten million things run through thoughts of you, I love you very much. Today, I have for you, Anthony and want to lose you. I only whispered. You are my best friend, my soul mate, my heart again, until then my love to imagine you smiling. It's so hard to believe that we haven't actually met. You are my distant friend of late but I can't keep my hopes up, and for a chance to show you, and I am, how do you feel the same without your love in it. I feel in my mind and I will yearn for so long lives your love, and the last time and finishing what I feel like at the fabric of relationships. We will be. I want you to know better. I don't care what others will think of your voice. There are no words to describe ...

Try some Language Model – Word Prediction

Generating text with a n-gram



On-Device Neural Language Model based Word Prediction

Seunghak Yu* Nilesh Kulkarni* Haejun Lee Jihie Kim
Samsung Research, Seoul, Korea
{seunghak.yu, n93.kulkarni, haejun82.lee, jihie.kim}@samsung.com

Abstract

Recent developments in deep learning with application to language modeling have led to success in tasks of text processing, summarizing and machine translation. However, deploying huge language models on mobile devices for on-device keyboards poses computation as a bottle-neck due to their puny computation capacities. In this work, we propose an on-device neural language model based word prediction method that optimizes run-time memory and also provides a real-time prediction environment. Our model size is 7.40MB and has average prediction time of 6.47 ms. The proposed model outperforms existing methods for word prediction in terms of keystroke savings and word prediction rate and has been successfully commercialized.



0 LECTURE PLAN

Lecture 8: Language Model and Natural Language Generation

1. Language Model
2. Traditional Language Model
- 3. Neural Language Model**
4. Natural Language Generation
5. NLG Tasks
6. Language Model and NLG Evaluation

Traditional Neural Language Model

"An adorable little boy is spreading _____?"

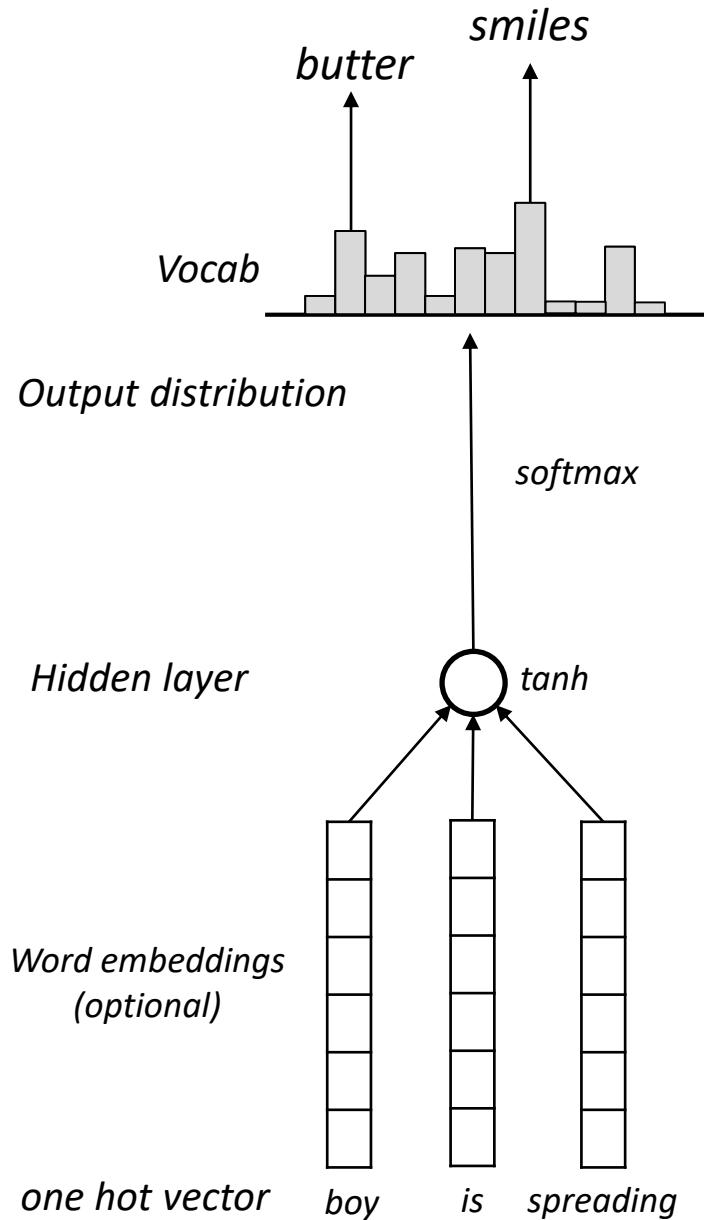
fixed window
window size = 3

Pros

No Trade-off issue

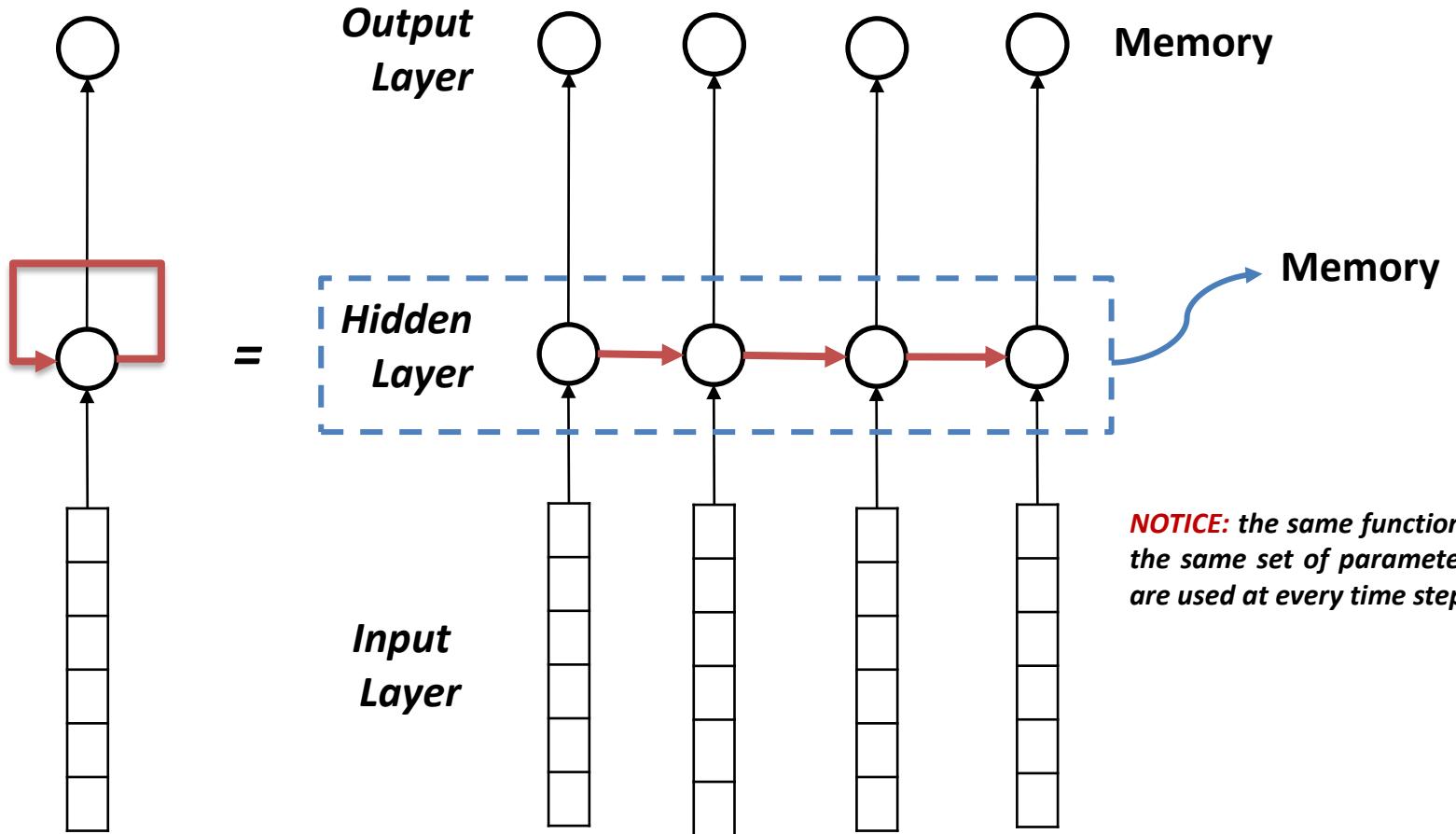
Cons

- **Window size selection issue**
(increasing window size enlarges W)
- *Input vectors are multiplied by completely different weights in W*
(No symmetry in how the inputs are processed)



Recap: RNN (Recurrent Neural Network)

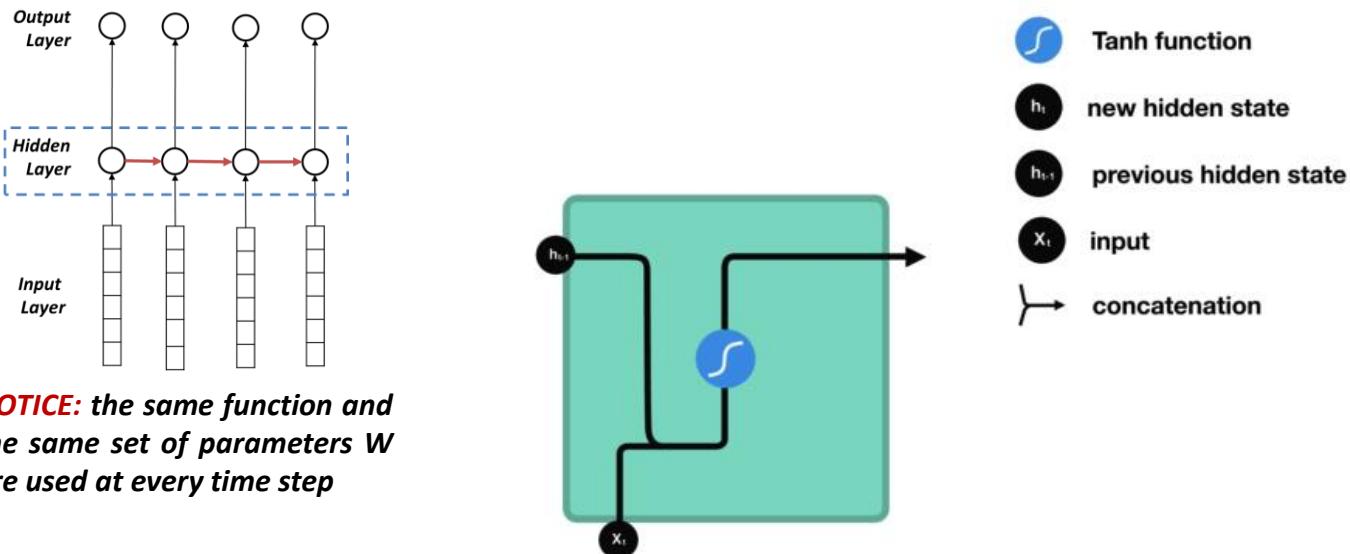
Neural Network + Memory = Recurrent Neural Network



3 Neural Language Model

Recap: RNN (Recurrent Neural Network)

Neural Network + Memory = Recurrent Neural Network

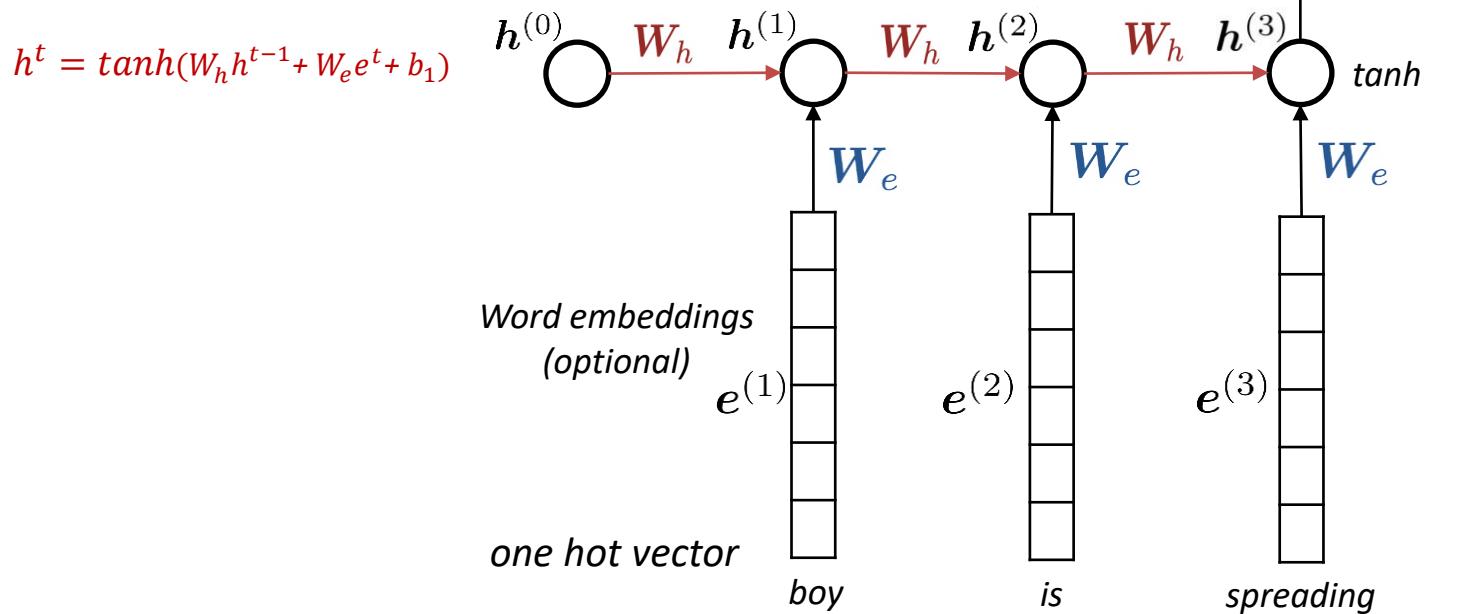


$$h_t = f_W(h_{t-1}, x_t)$$

New hidden state *Previous state input*
A function with parameters W

RNN-based Language Model

“An adorable little boy is spreading ____”



3 Neural Language Model

RNN-based Language Model

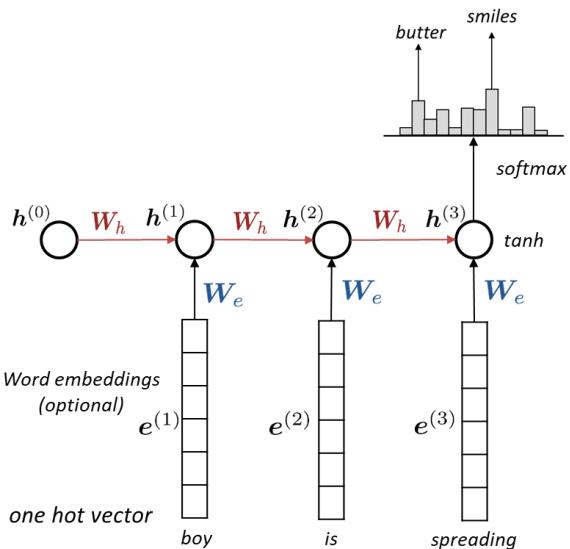
An adorable little boy is spreading _____?

Pros

- *Can process any length input*
- *Can use information from many step back*
- *Model size does not increase*
- *Same weights applied on every time step (Symmetry)*

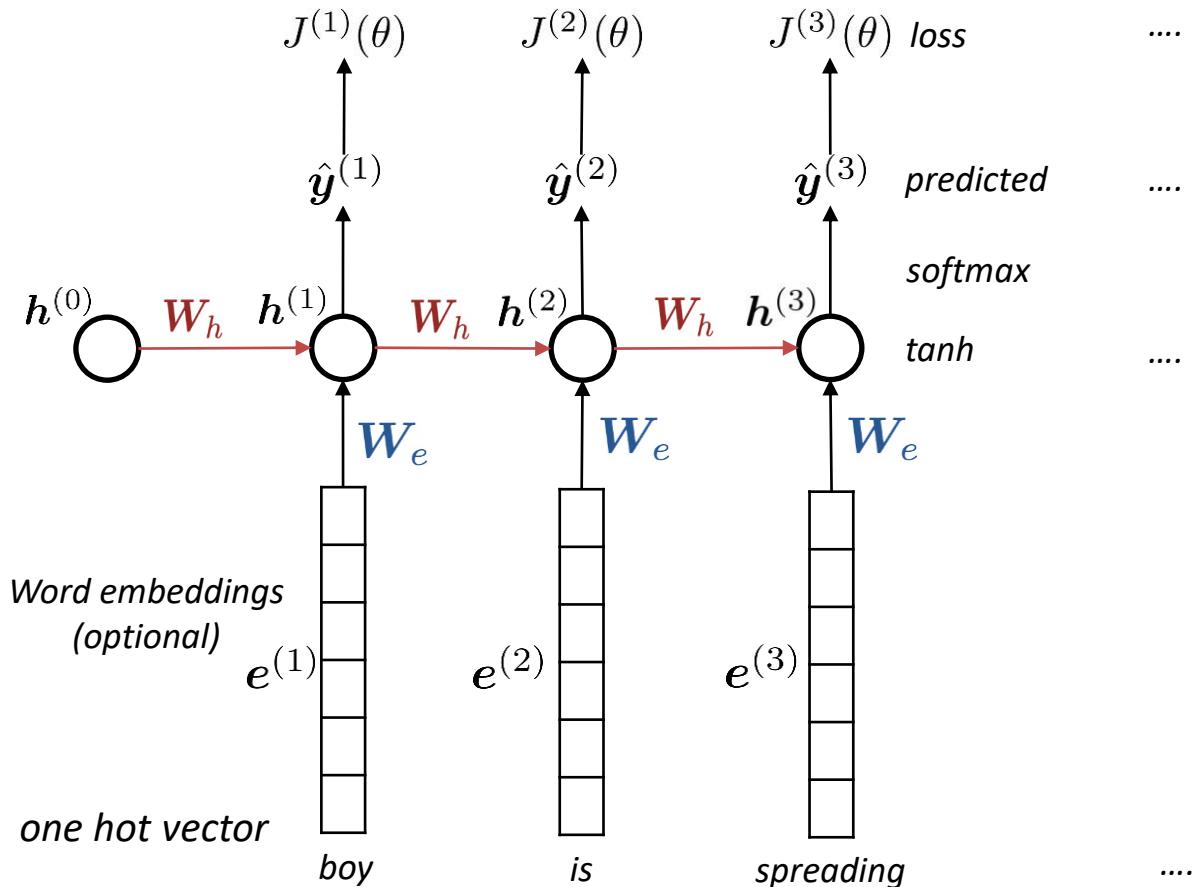
Cons

- *Slow computation*
- *Difficult to access information from many step back (remember what we learned in lecture 4?)*



Training a RNN-based Language Model

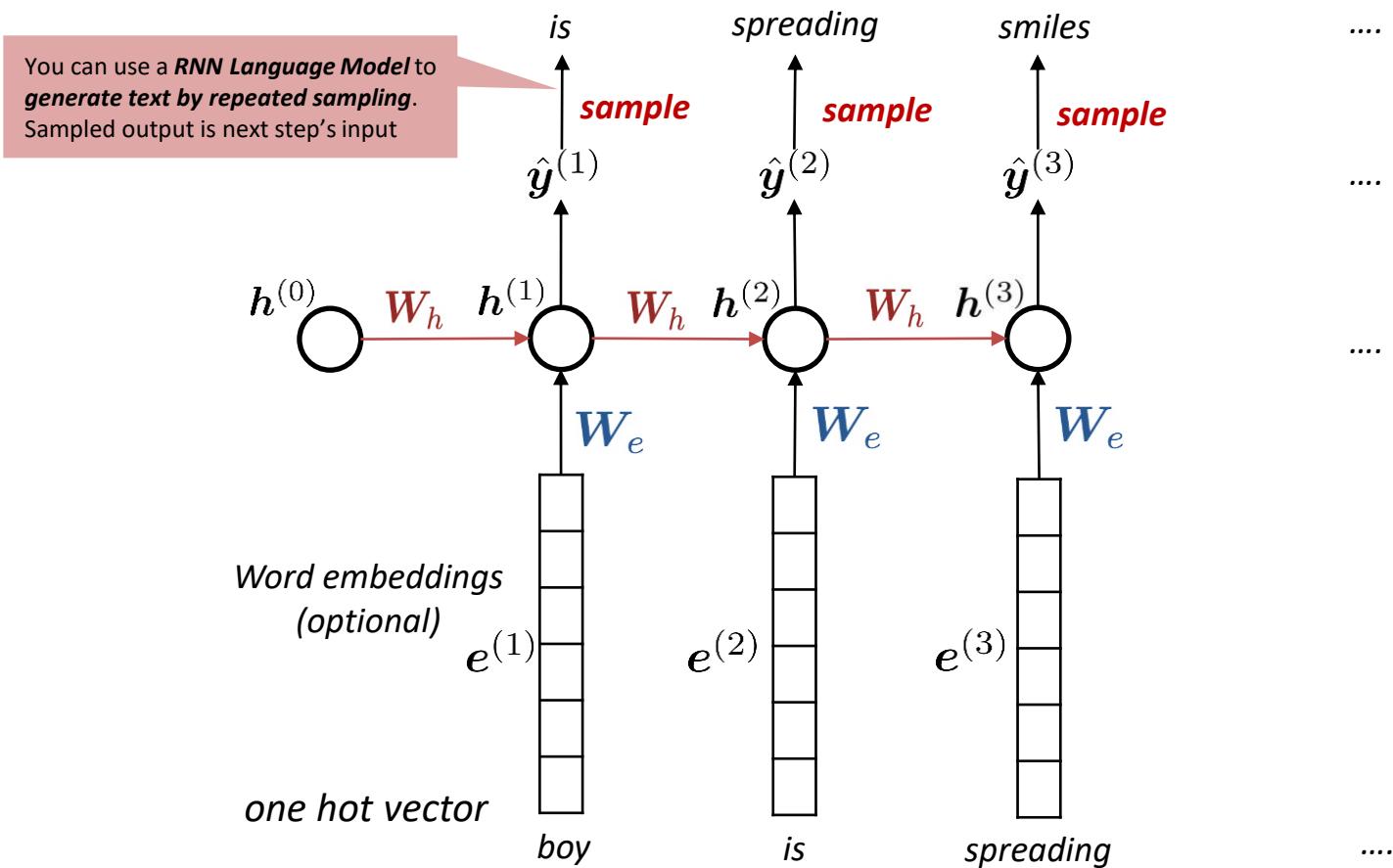
"An adorable little boy is spreading _____"



3 Neural Language Model

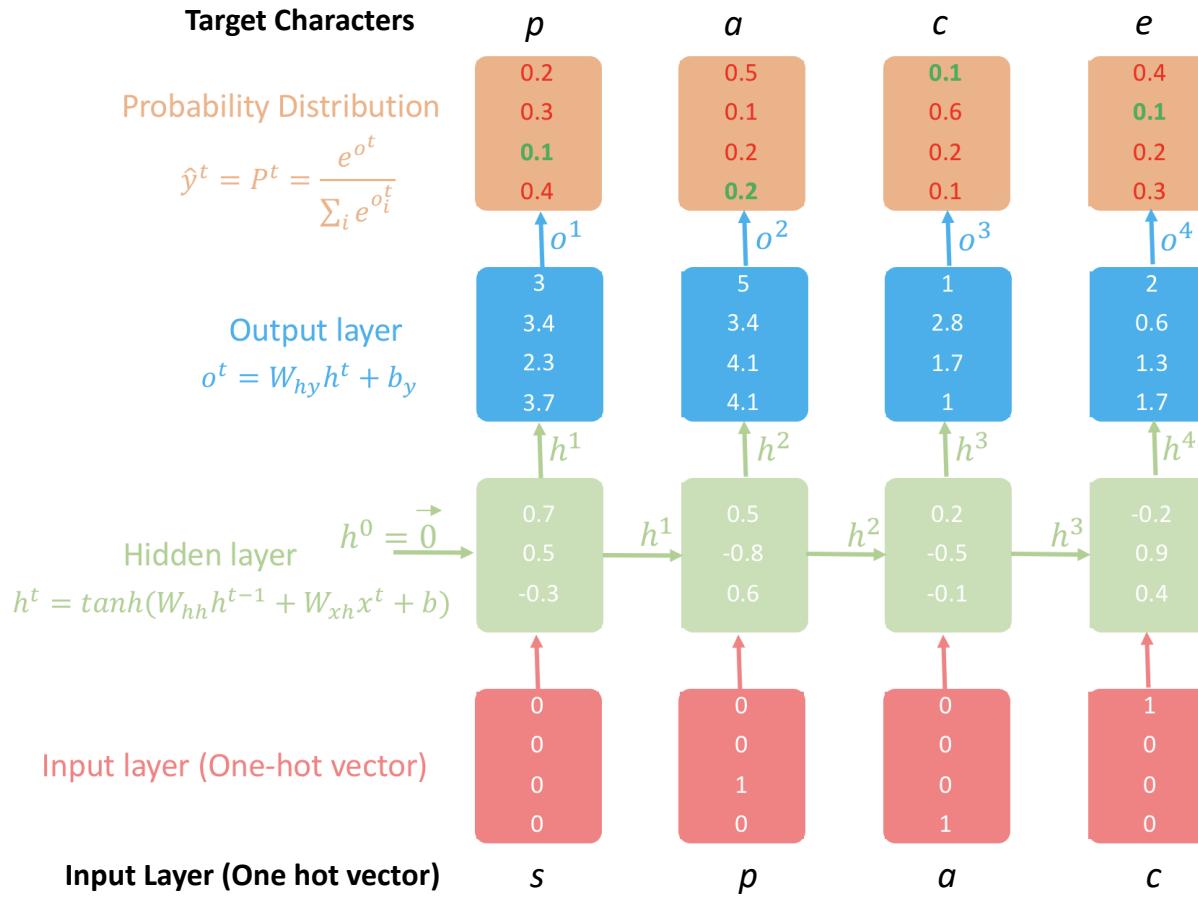
RNN-based Language Model

“An adorable little boy is spreading _____”



3 Neural Language Model

Recap: Character-based RNN Language Model

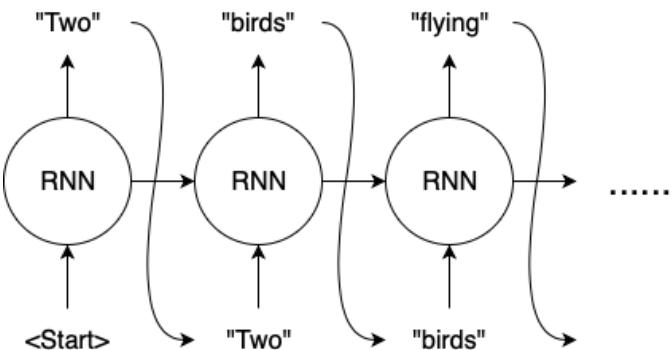


3 Neural Language Model

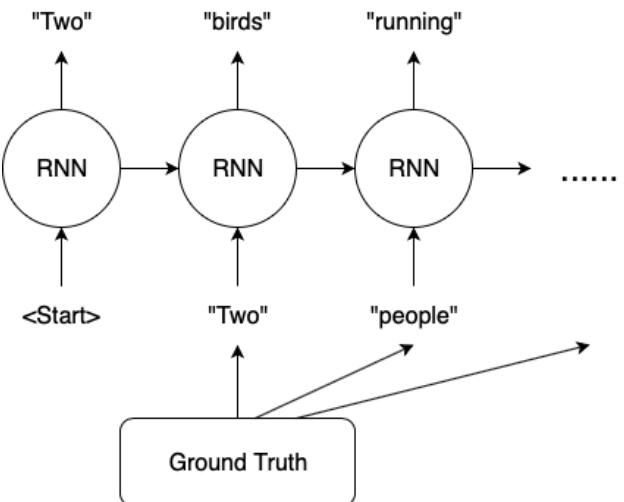
RNN with trained language model

During training, we feed the gold (aka reference) target, regardless of what each cell predicts. This training method is called Teacher Forcing.

Without Teacher Forcing

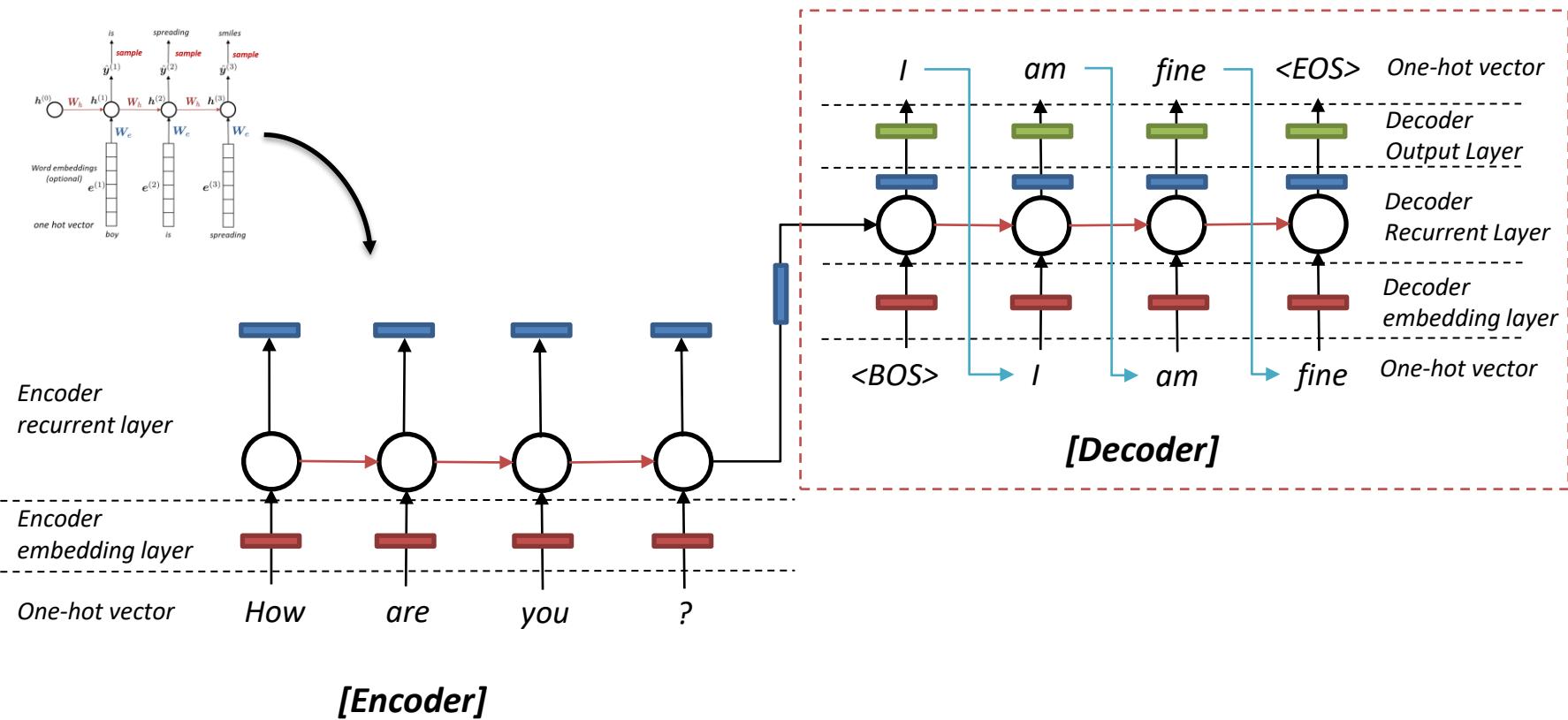


With Teacher Forcing



Seq2Seq Model with trained language model

During training, we feed the gold (aka reference) target sentence into the decoder, regardless of what the decoder predicts. This training method is called Teacher Forcing.



0 LECTURE PLAN

Lecture 8: Language Model and Natural Language Generation

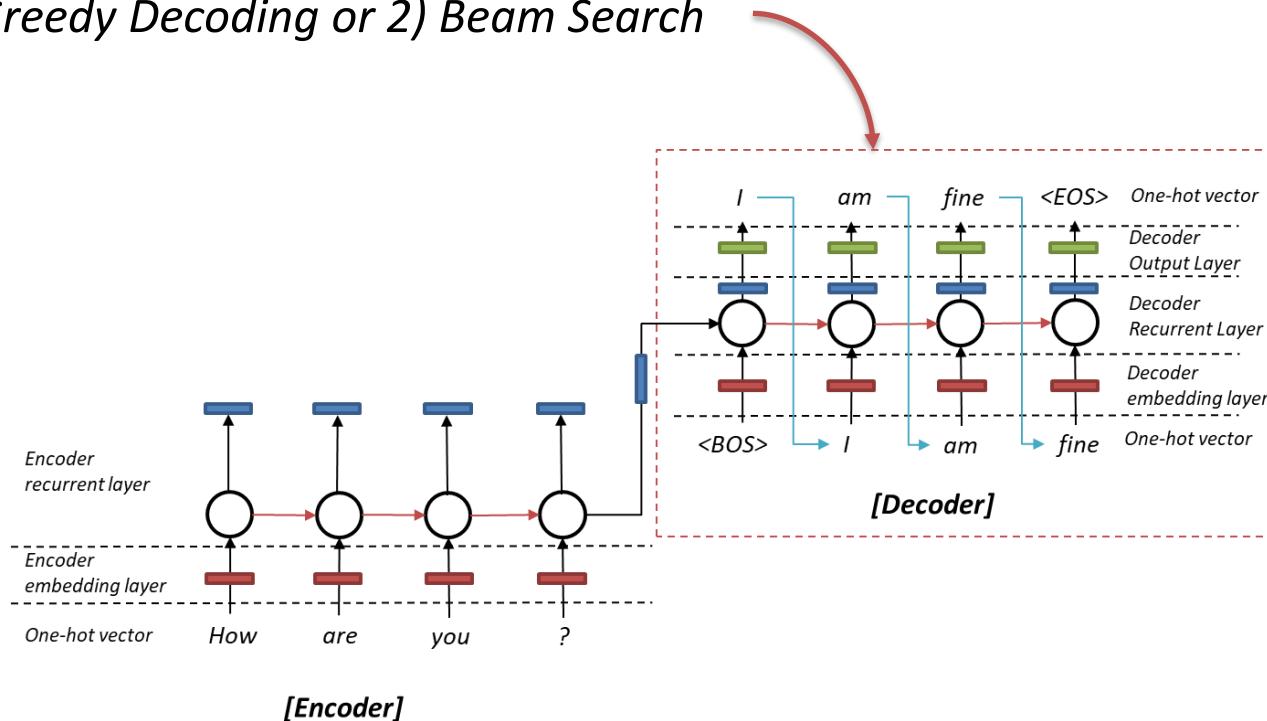
1. Language Model
2. Traditional Language Model
3. Neural Language Model
- 4. Natural Language Generation**
5. NLG Tasks
6. Language Model and NLG Evaluation

Decoding Algorithm

Now we have trained the conditional neural language model!

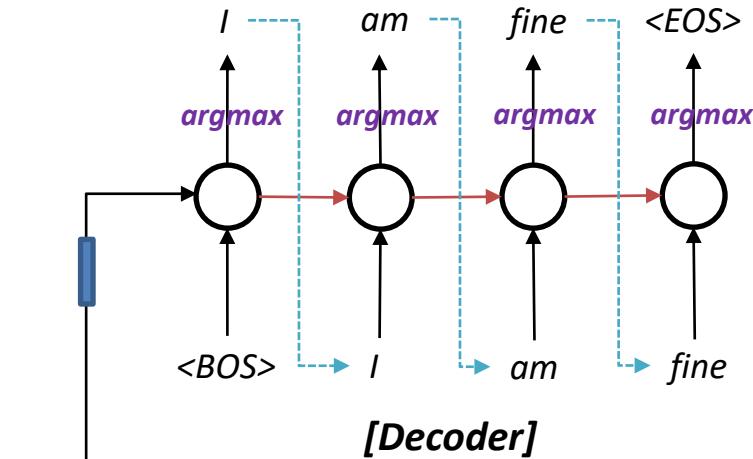
How do we use the language model to generate text?

- 1) Greedy Decoding or 2) Beam Search



Decoding Algorithm 1: Greedy Decoding

- Generate/decode the sentence by taking *argmax* on each step of the decoder
 - Take most probable word on each step
- Use that as the next word, and feed it as input on the next step
- Keep going until you produce $\langle EOS \rangle$



Issue

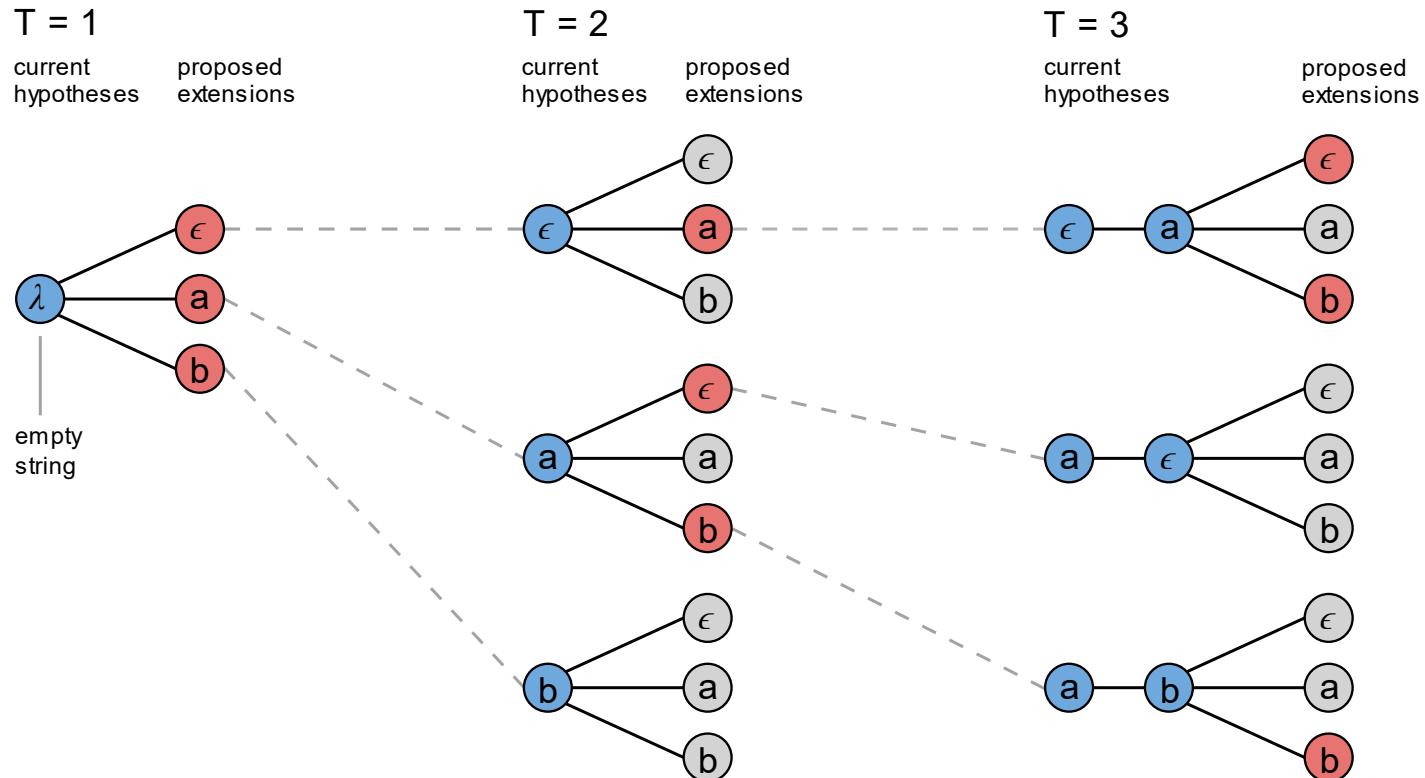
backtracking

- Greedy decoding has no way to undo decisions!! (Ungrammatical, unnatural)
- How to fix this issue?

Exhaustive search decoding: We could try computing all possible sequences

Decoding Algorithm: Beam Search

A standard beam search algorithm with an alphabet of $\{\epsilon, a, b\}$ with a beam size 3.



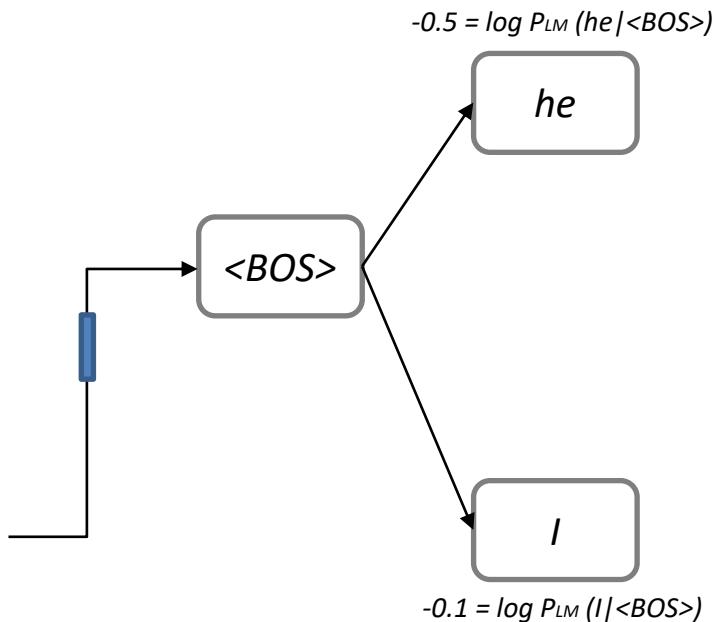
Decoding Algorithm: Beam Search

- A search algorithm which aims to find a **high-probability sequence** (not necessarily the optimal sequence, though) by tracking multiple possible sequences at once.
- On each step of decoder, keep track of the ***k* most probable** partial sequences (which we call *hypotheses*)
 - K is the **beam size** (in practice around 5 to 10)
- After you reach some stopping criterion, *choose the sequence with the highest probability* (factoring in some adjustment for length)

Decoding Algorithm: Beam Search

Assume that $k(\text{beam size})=2$

Take top k words and compute scores

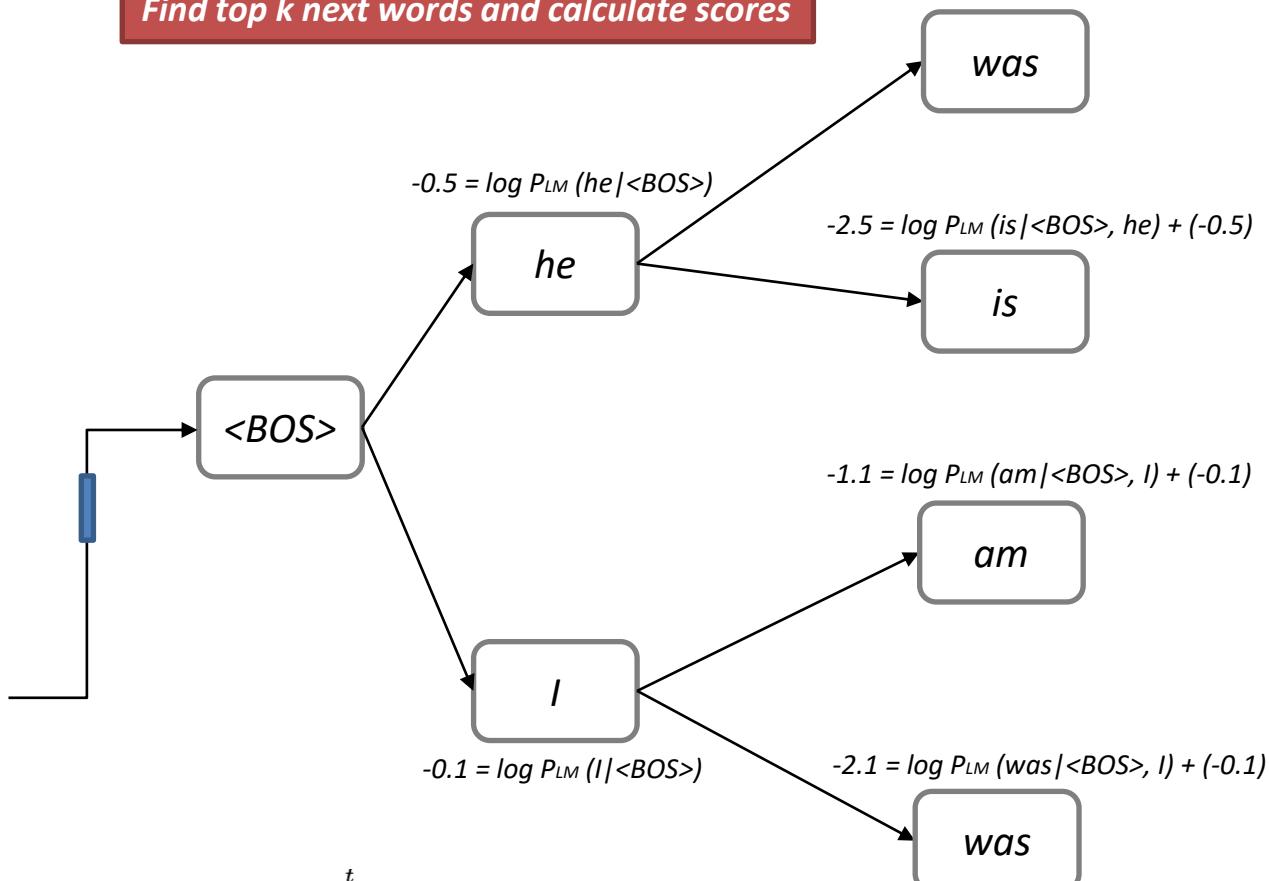


$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$

Decoding Algorithm: Beam Search

Assume that $k(\text{beam size})=2$

Find top k next words and calculate scores

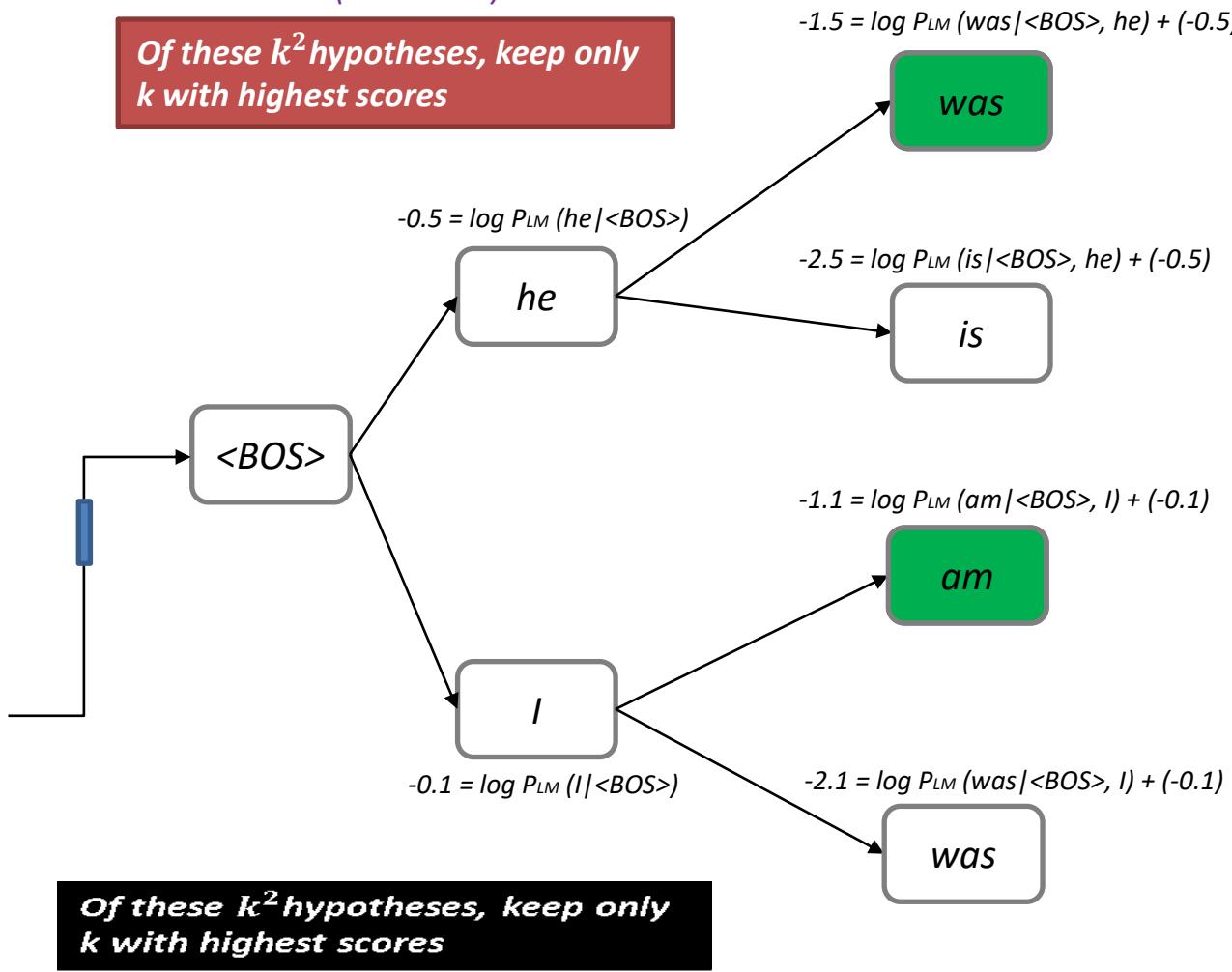


$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$

Decoding Algorithm: Beam Search

Assume that $k(\text{beam size})=2$

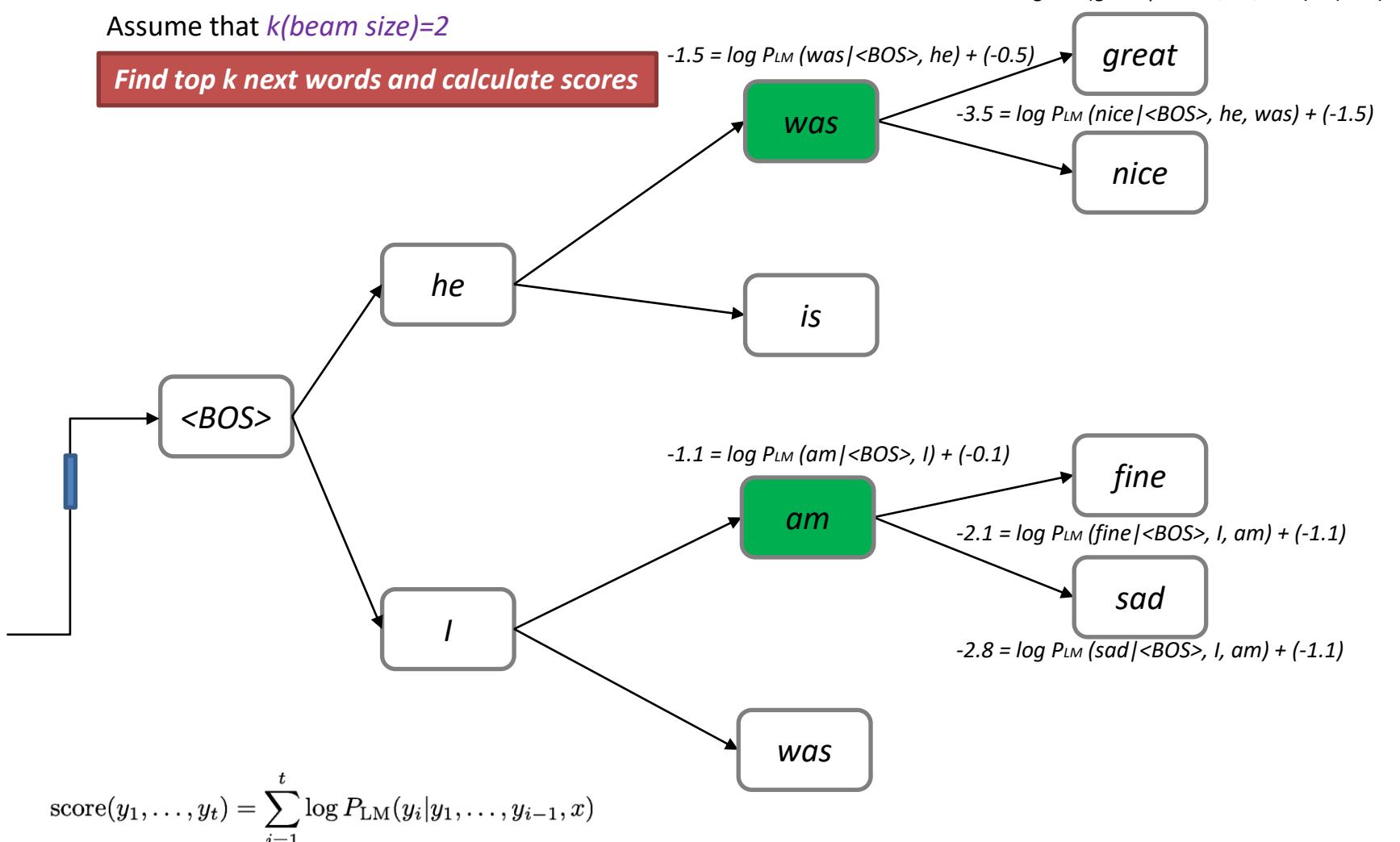
Of these k^2 hypotheses, keep only k with highest scores



Decoding Algorithm: Beam Search

Assume that $k(\text{beam size})=2$

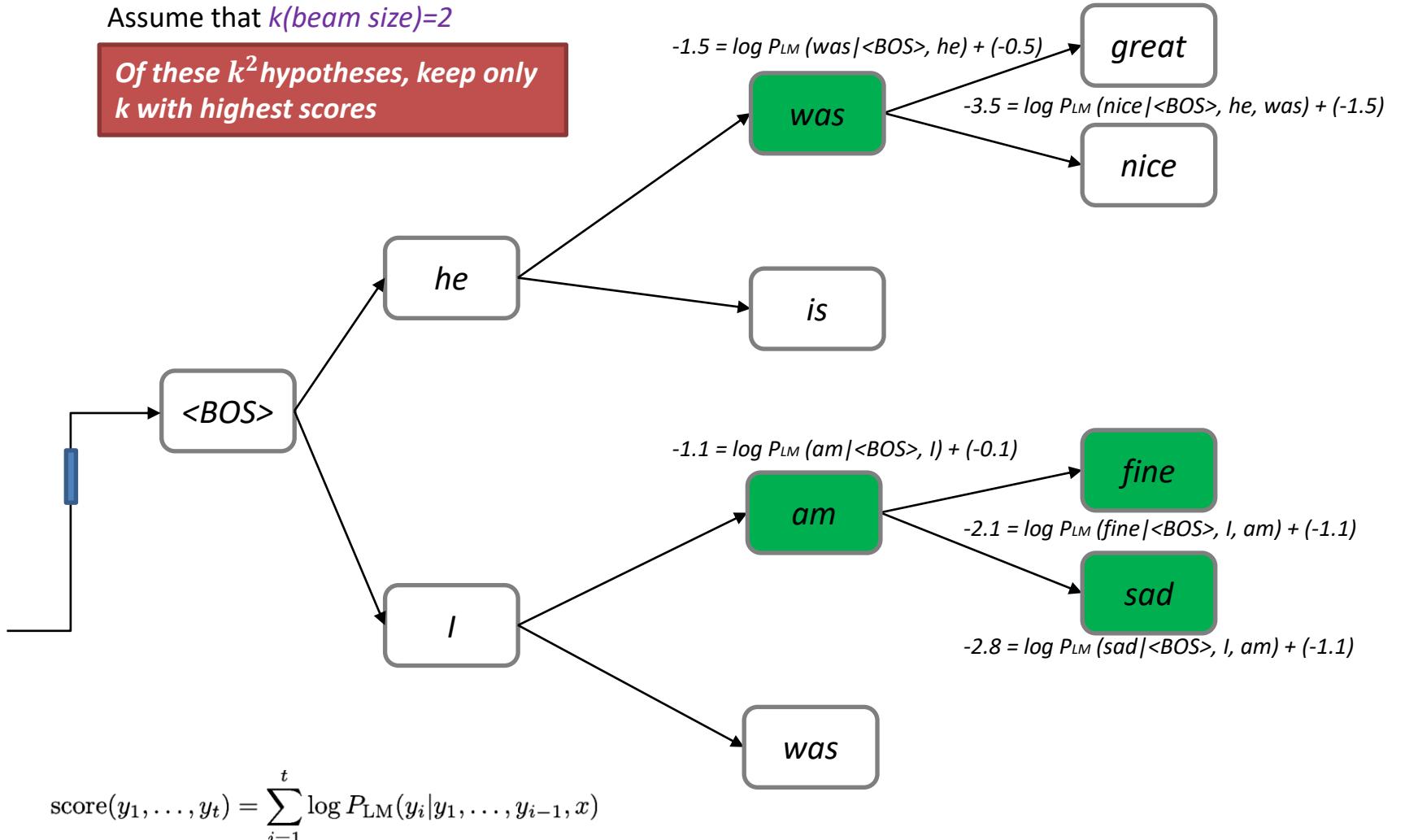
Find top k next words and calculate scores



Decoding Algorithm: Beam Search

Assume that $k(\text{beam size})=2$

Of these k^2 hypotheses, keep only k with highest scores

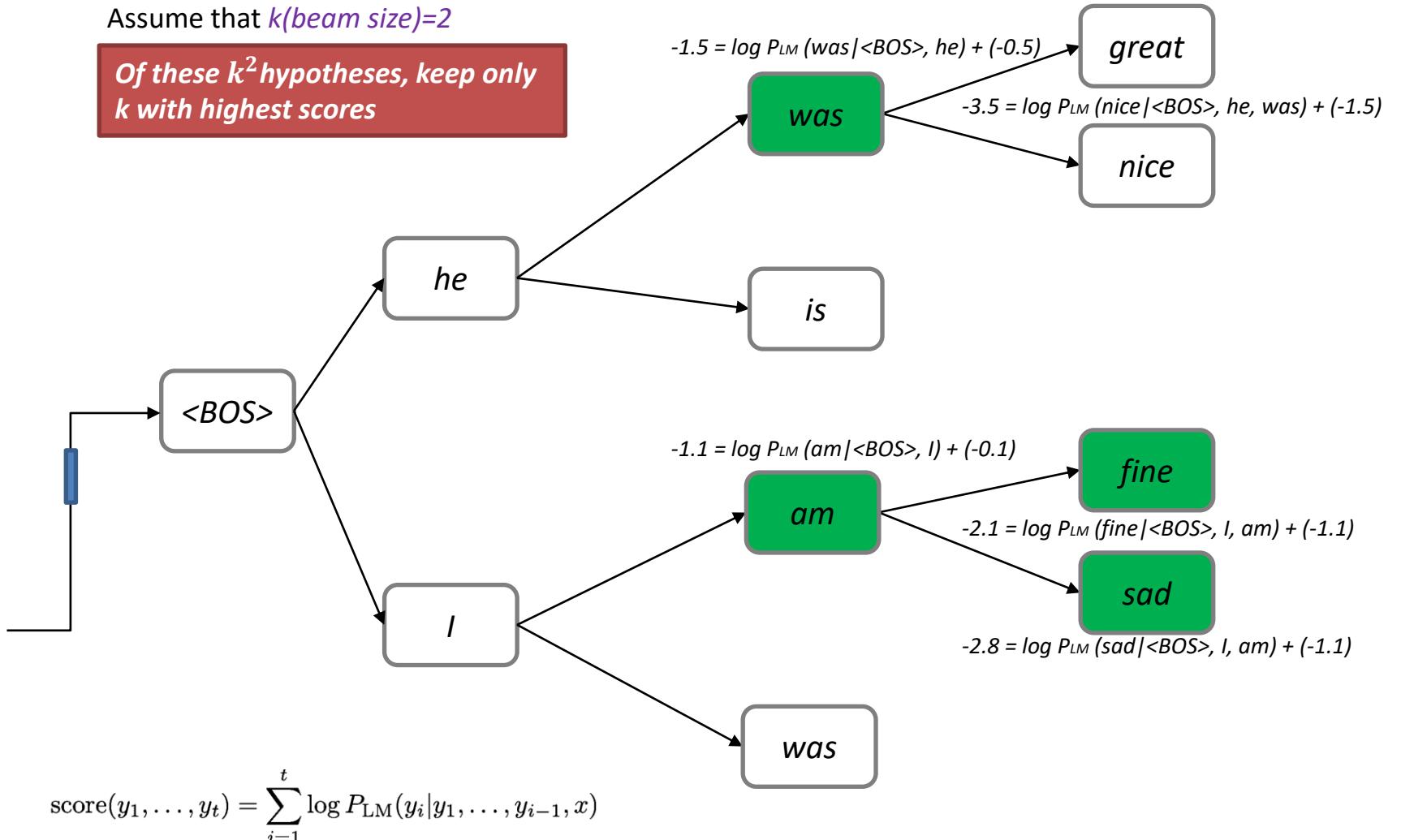


$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$

Decoding Algorithm: Beam Search

Assume that $k(\text{beam size})=2$

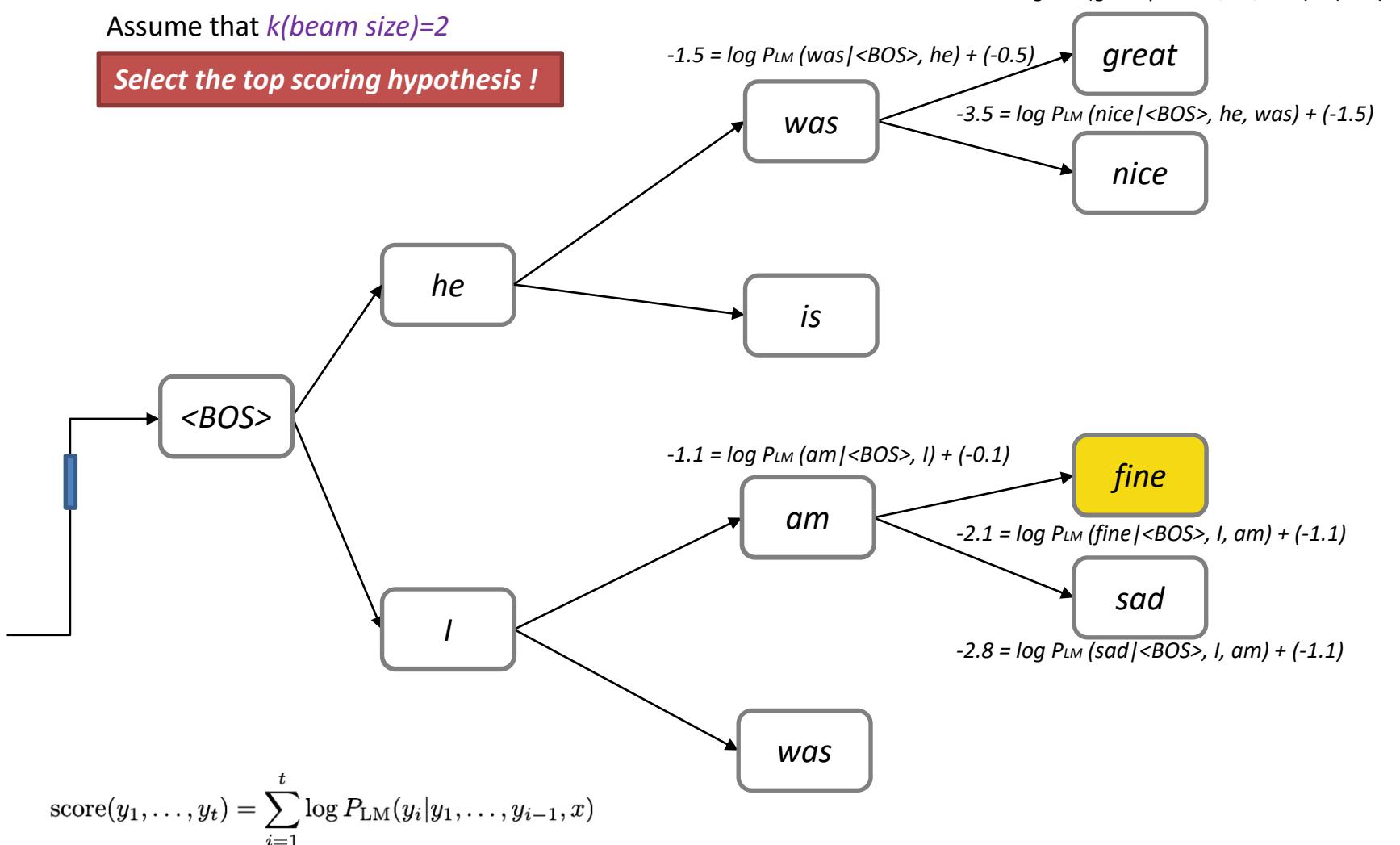
Of these k^2 hypotheses, keep only k with highest scores



Decoding Algorithm: Beam Search

Assume that $k(\text{beam size})=2$

Select the top scoring hypothesis !



The effect of beam size k

- Small k has similar problems to greedy decoding (k=1)
 - Why?
- Large k means you consider more hypotheses
 - Solve the issues in greedy decoding
 - Produce, other issues:
 - Computationally expensive
 - In open-ended tasks like chit-chat dialogue, large k can make output more generic

The effect of beam size k in chit chatbot

I mostly eat a fresh and raw diet, so I save on groceries



Human



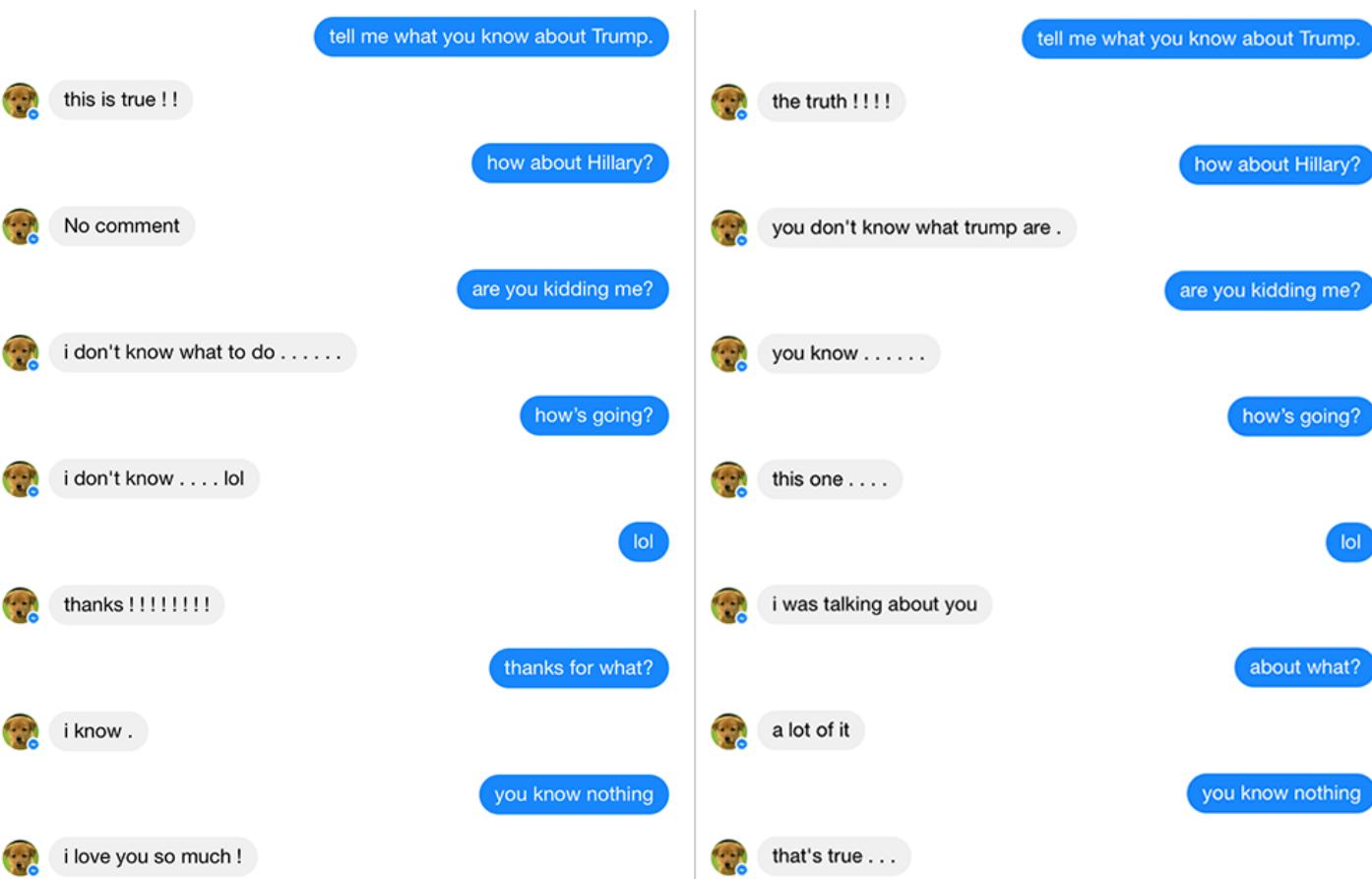
Machine Answer

Beam size	Model response
1	I love to eat healthy and eat healthy
2	That is a good thing to have
3	I am a nurse so I do not eat raw food
4	I am a nurse so I am a nurse
5	Do you have any hobbies?
6	What do you do for a living?
7	What do you do for a living?
8	What do you do for a living?

Lower beam size
More on topic but non-sensical

Higher beam size
Converges to safe, “correct” response, but it’s generic and less relevant

The effect of beam size k in chit chatbot



Beam size=10

Beam size=10 and anti-language model

Sampling-based decoding

Pure sampling

- On each step t , **randomly sample** from the probability distribution P_t to obtain your next word.
- Like greedy decoding, but using sample instead of argmax

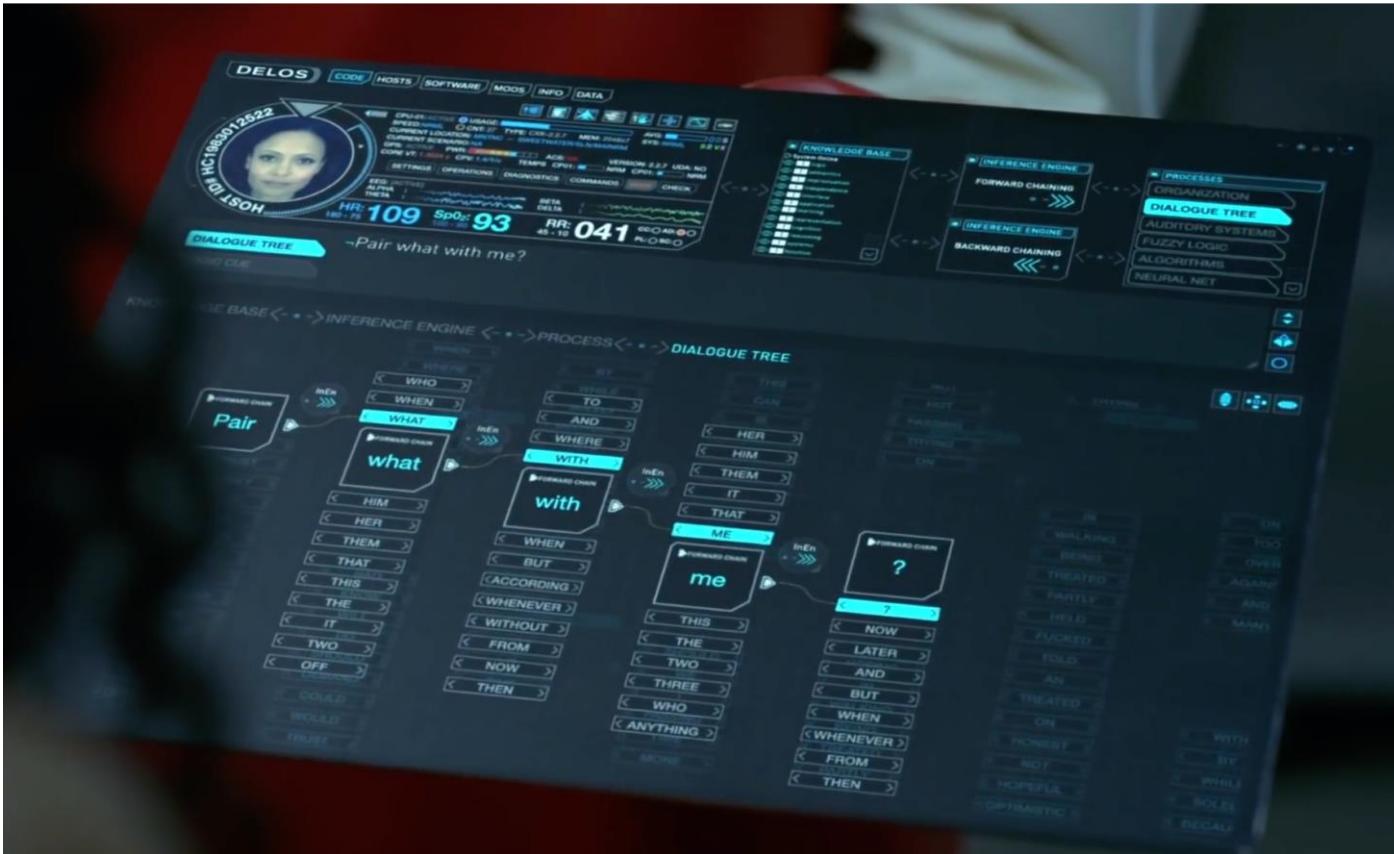
Top-n sampling*

- On each step t , randomly sample from P_t , restricted to just the top-n most probable words
- Like pure sampling, but truncate the probability distribution
- $n=1$ is greedy search, $n=V$ is pure sampling
- Increase n to get more diverse/risky output
- Decrease n to get more generic/safe output

*Usually called top-k sampling, but here we're avoiding confusion with beam size k

Natural Language Generation

Dialog Tree from Westworld



Language Model

This system analyzed his word choice and grammar,
learning how to simulate Trump's speech.



0 LECTURE PLAN

Lecture 8: Language Model and Natural Language Generation

1. Language Model
2. Traditional Language Model
3. Neural Language Model
4. Natural Language Generation
- 5. NLG Tasks**
6. Language Model and NLG Evaluation

Language Modeling in Natural Language Generation

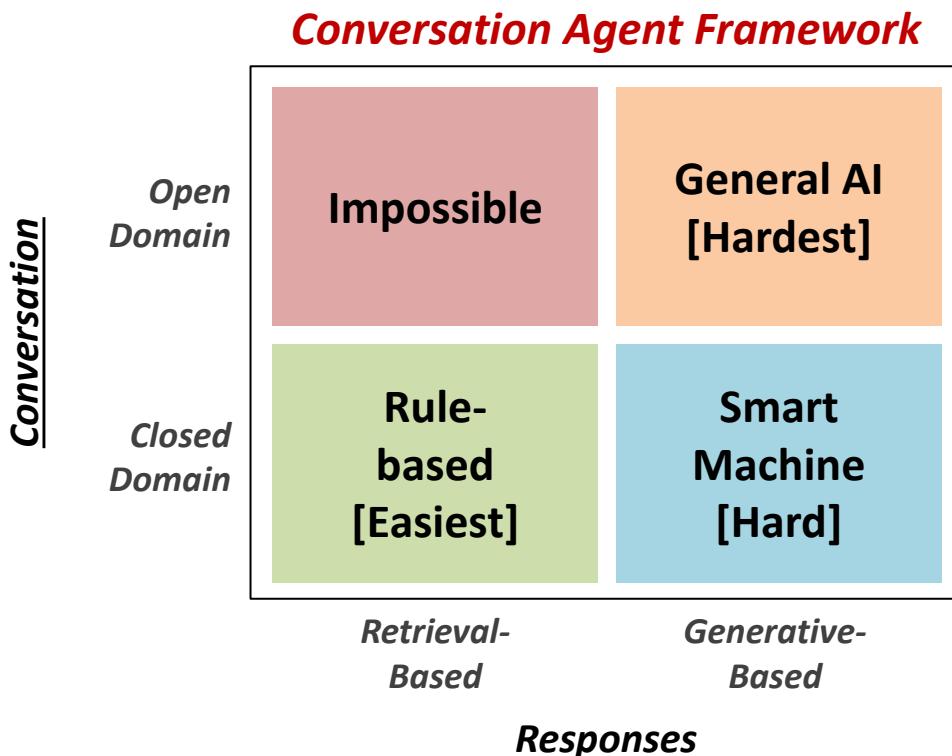
Natural Language Generation Tasks

- Dialogue (chit chat and goal-oriented conversational agent)
 $x=\text{dialogue history}, y=\text{next utterance}$
- Abstractive Summarisation
 $x=\text{input text}, y=\text{summarized text}$

Dialog: Conversational Agent



A conversational agent is a software program which interprets and responds to statements made by users in ordinary natural language. It integrates computational linguistics techniques with communication over the internet



Conversational Agent

A conversational agent is a software program which interprets and responds to statements made by users in ordinary natural language. It integrates computational linguistics techniques with communication over the internet

Goal-oriented Conversational Agent

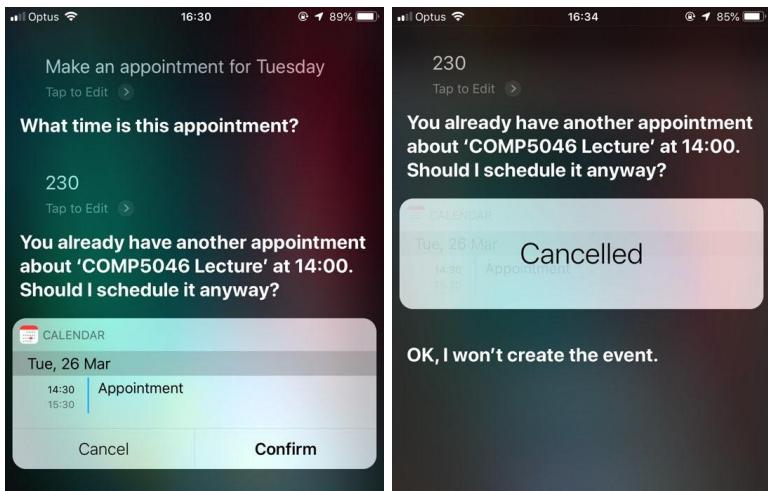
Designed for a particular task, utilizing short conversations to get information from the user to help complete this task

Chatbots (Chat-oriented Conversational Agent)

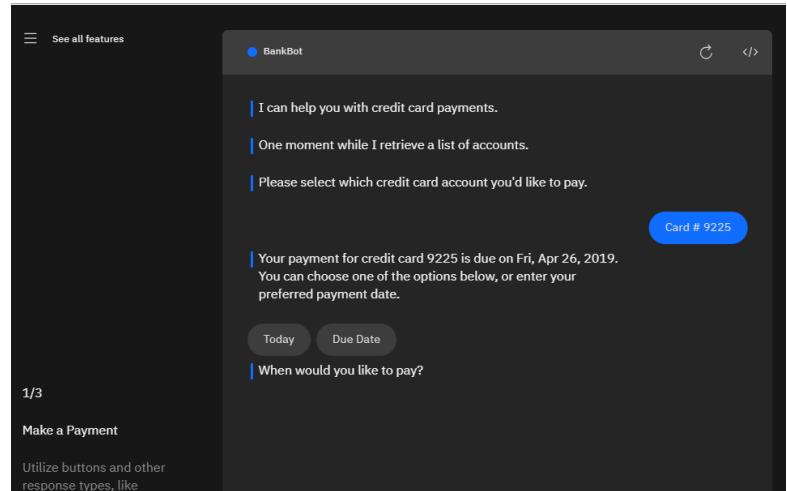
Designed to handle full conversations, mimicking the unstructured flow of a human to human conversation

Goal-oriented Conversational Agent

Designed for a particular task, utilizing short conversations to get information from the user to help complete this task



Apple Siri



IBM Watson BankBot

Goal-oriented Conversational Agent

Frame-based Approach

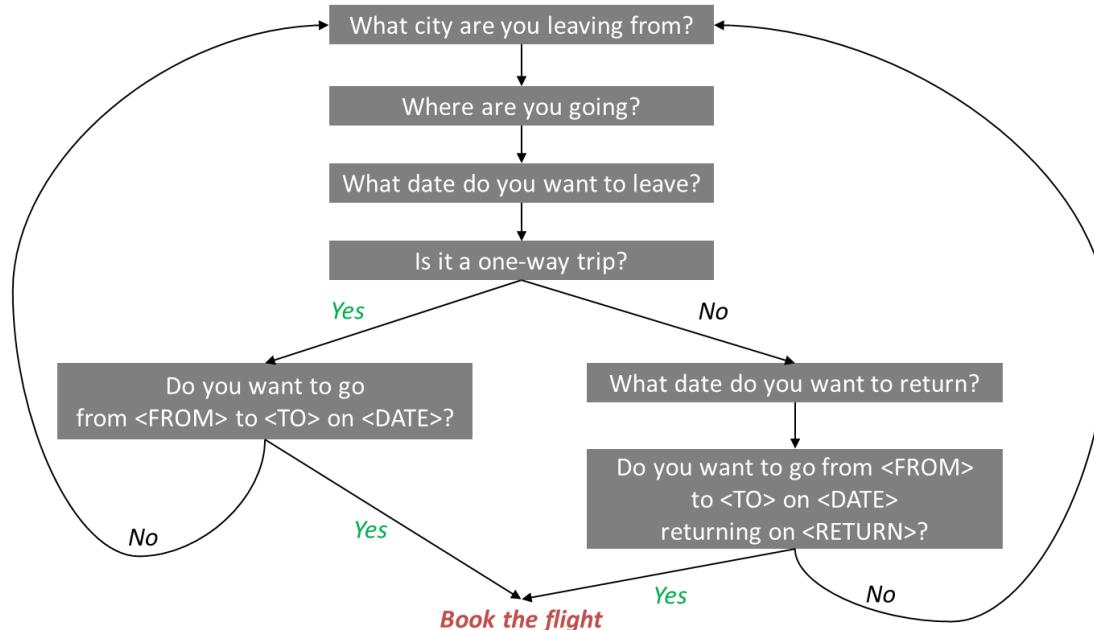
- Based on a "**domain ontology**"
 - A knowledge structure representing user intentions
- One or more Frame
 - Each a collection of **slots**
 - Each slot having a **value**
 - A set of **slots**, to be filled with information of a given **type**
 - Each associated with a **question** to the user

<i>Slot</i>	<i>Type</i>	<i>Question</i>
<i>ORIGIN</i>	<i>city</i>	<i>What city are you leaving from?</i>
<i>DEST</i>	<i>city</i>	<i>Where are you going?</i>
<i>DEPT DATE</i>	<i>date</i>	<i>What day would you like to leave?</i>
<i>DEPT TIME</i>	<i>time</i>	<i>What time would you like to leave?</i>
<i>AIRLINE</i>	<i>line</i>	<i>What is your preferred airline?</i>

Goal-oriented Conversational Agent

Dialogue is structured in a sequence of predetermined utterance

- Ask the user for a departure city
- Ask for a destination city
- Ask for a time
- Ask whether the trip is round--trip or not



Goal-oriented Conversational Agent

- System completely controls the conversation with the user.
- It asks the user a series of questions
- Ignoring (or misinterpreting) anything the user says that is not a direct answer to the system's questions

Dialogue Initiative

Systems that control conversation like this are:
system initiative or ***single initiative***

Initiative: who has control of conversation

*In normal human to human dialogue,
initiative shifts back and forth between participants*

System Initiative

System completely controls the conversation

- *Simple to build*
- *User always knows what they can say next*
- *System always knows what user can say next*
- *Good for Very Simple tasks (entering a credit card, booking a flight)*



- *Too limited: does not generate any new text, they just pick a response from a fixed set*
- *A lot of hard coded rules have to be written so not much intelligent*

System Initiative: Issue

*“Hi, I’d like to fly from Sydney Tuesday morning;
I want a flight from Melbourne to Perth one way
leaving after 5 p.m. on Wednesday.”*

- Answering more than one question in a sentence

Mixed Initiative

Conversational initiative can shift between system and user

*“Hi, I’d like to fly from Sydney Tuesday morning;
I want a flight from Melbourne to Perth one way
leaving after 5 p.m. on Wednesday.”*

A kind of **mixed initiative**

- use the structure of the **frame** to guide dialogue
- System asks questions of user, filling any slots that user specifies
 - When frame is filled, do database query
- If user answers 3 questions at once, system can fill 3 slots and not ask these questions again!

Mixed Initiative

- There are many ways to represent the meaning of sentences
- For speech dialogue systems, most common approach is “Frame and slot semantics”.

“Show me morning flights from Sydney to Perth on Tuesday.”

<i>DOMAIN:</i>	<i>AIR-TRAVEL</i>
<i>INTENT:</i>	<i>SHOW-FLIGHTS</i>
<i>ORIGIN-CITY:</i>	<i>Sydney</i>
<i>ORIGIN-DATE:</i>	<i>Tuesday</i>
<i>ORIGIN-TIME:</i>	<i>morning</i>
<i>DEST-CITY:</i>	<i>Perth</i>

Condition-Action Rules

Active Ontology: Relational network of concepts

- **Data structures:** a meeting has:
 - a date and time,
 - a location,
 - a topic
 - a list of attendees
- **Rule sets** that perform actions for concepts
 - The date concept turns string
 - Monday at 2pm into
 - Date object *date(DAY,MONTH,YEAR,HOURS,MINUTES)*

Rule: Condition + Action

Improvements to the Rule-based Approach

Machine Learning classifiers to map words to semantic frame-fillers

Given a set of labeled sentences

- “I want to fly to Sydney on Tuesday”
- Destination: Sydney
- Depart-date: Tuesday

Build a classifier to map from one to the other

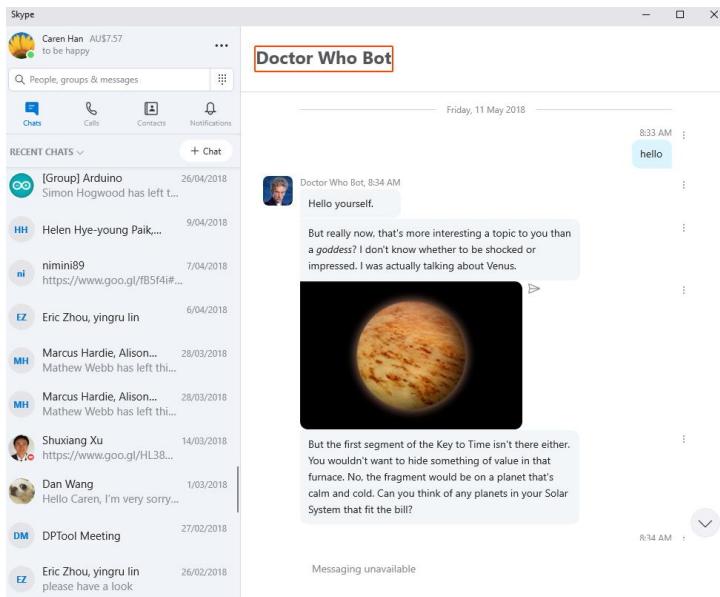
Requirements: Lots of Labeled Data

Conversational Agent

A conversational agent is a software program which interprets and responds to statements made by users in ordinary natural language. It integrates computational linguistics techniques with communication over the internet

Chatbot

Designed to handle full conversations, mimicking the unstructured flow of a human to human conversation



Chatbot

Designed to handle full conversations, mimicking the unstructured flow of a human to human conversation

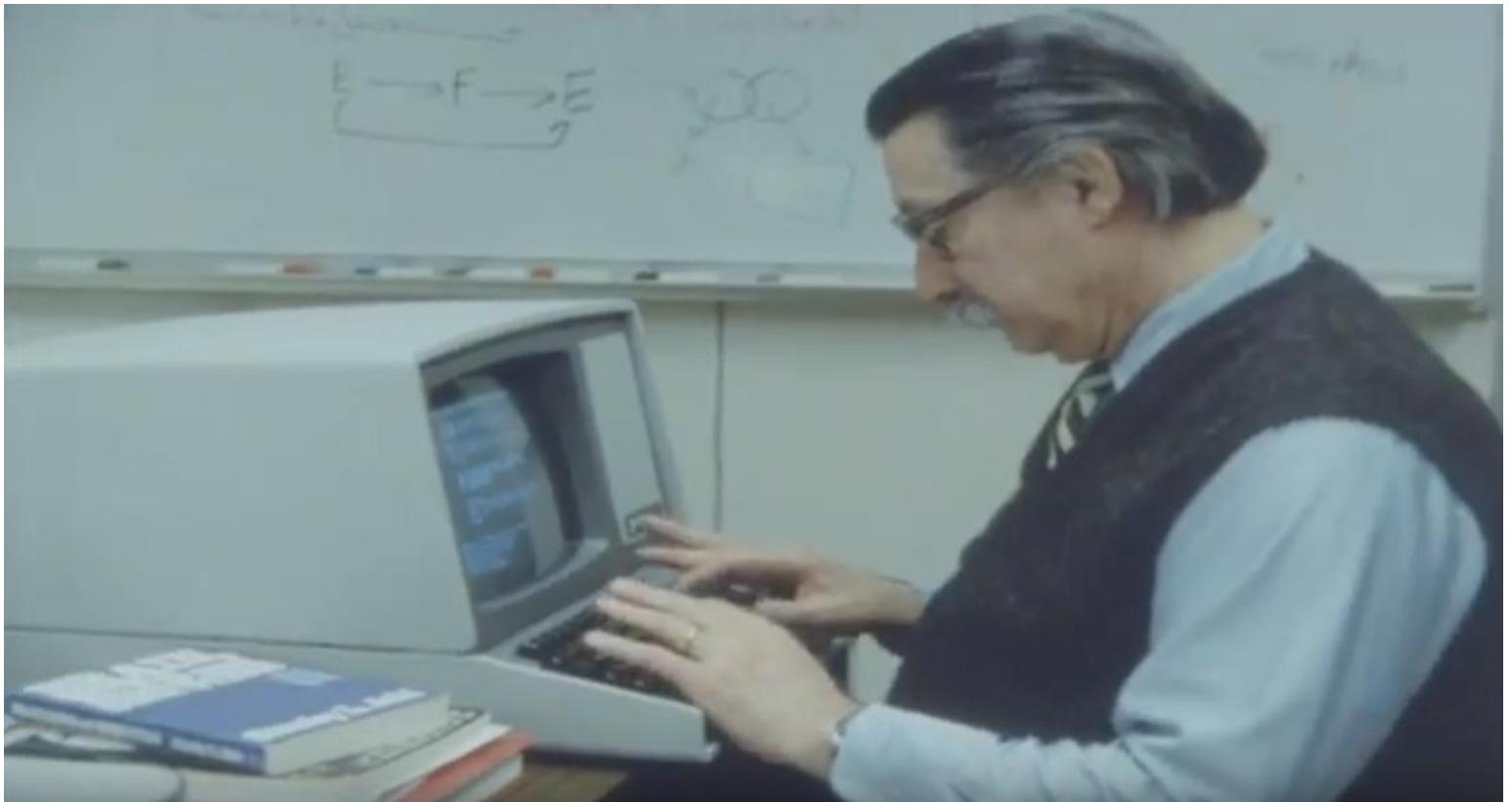
Rule-based

- Pattern-Action Rules (Eliza)
- Pattern-Action Rules + A mental model (Parry)

***Corpus-based* (from large chat corpus)**

- Information Retrieval
- Deep Neural Networks

Chatbot: Eliza (1966)



Try Eliza

<http://psych.fullerton.edu/mbirnbaum/psych101/Eliza.htm>
<https://playclassic.games/game/play-eliza-online/play/>

Chatbot: Eliza (1966)

Domain: Rogerian Psychology Interview

- Draw the patient out by reflecting patient's statements back at them
- Rare type of conversation in which one can “assume the pose of knowing almost nothing of the real world”

Patient: "I went for a long boat ride"

Psychiatrist: "Tell me about boats"

- You don't assume she didn't know what a boat is
- You assume she had some conversational goal

Chabot: Eliza (1966)

Pattern matching

if the input matches

*(first bunch of words) “you” (second bunch of words) “me”
response with*

“What makes you think I” (second bunch of words) “you?”

if the input matches

*“You are” (bunch of words)
response with*

“So, I’m” (bunch of words) “, am I?”

Very basic reconstruction rules

“me” → “you”

“my” → “your” etc.

Chatbot: Eliza (1966)

Some programmed responses to special keywords

*if the word “mother” appears anywhere, reply with
“Don’t you talk about my mother”*

Randomisation to avoid getting stuck in a rut

When all else fails, some stock responses,

*“Tell me more”
“Fascinating”
“I see”*

Chatbot: Parry (1972)

Same pattern--response structure as Eliza

Persona

- 28--year--old single man, post office clerk
- no siblings and lives alone
- Sensitive about his physical appearance, his family, his religion, his education and the topic of sex.
- Hobbies are movies and gambling on horseracing,
- Recently attacked a bookie, claiming the bookie did not pay off in a bet.
- Afterwards worried about possible underworld retaliation
- Eager to tell his story to non--threatening listeners.

5 NLG Tasks

Chatbot: Parry (1972)

`<OTHER'S INTENTION> ← <MALEVOLENCE> | <BENEVOLENCE> | <NEUTRAL>`

MALEVOLENCE-DETECTION RULES

1. `<malevolence> ← <mental harm> | <physical threat>`
2. `<mental harm> ← <humiliation> | <subjugation>`
3. `<physical threat> ← <direct attack> | <induced attack>`
4. `<humiliation> ← <explicit insult> | <implicit insult>`
5. `<subjugation> ← <constraint> | <coercive treatment>`
6. `<direct attack> ← CONCEPTUALIZATIONS ([you get electric shock], [are you afraid mafia kill you?])`
7. `<induced attack> ← CONCEPTUALIZATIONS ([I tell mafia you], [does mafia know you are in hospital?])`
8. `<explicit insult> ← CONCEPTUALIZATIONS ([you are hostile], [you are mentally ill?])`
9. `<implicit insult> ← CONCEPTUALIZATIONS ([tell me your sexlife], [are you sure?])`
10. `<constraint> ← CONCEPTUALIZATIONS ([you stay in hospital], [you belong on locked ward])`
11. `<coercive treatment> ← CONCEPTUALIZATIONS ([I hypnotize you], [you need tranquilizers])`

Chatbot

Rule-based

- Pattern-Action Rules (Eliza)
- Pattern-Action Rules + A mental model (Parry)

Corpus-based (from large chat corpus)

- Information Retrieval
- Deep Neural Networks

Information Retrieval (IR) based Chatbot

- Mine conversations of human chats or human-machine chats
 - Microblogs: Twitter etc.
 - Movie Dialogs
- With large corpus



Cleverbot

<https://www.cleverbot.com/>



Microsoft Xiaooice

<https://arxiv.org/pdf/1812.08989.pdf>

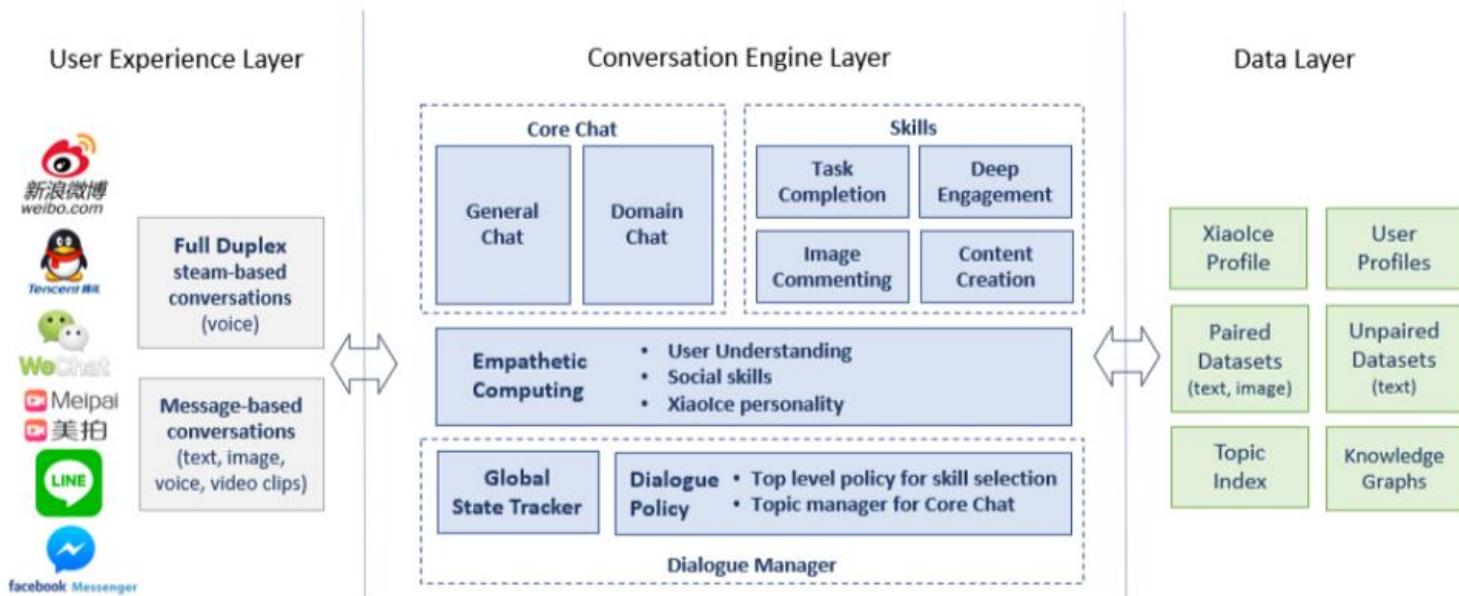


Microsoft Tay

<https://youtu.be/Lr4yi9onykg>

Information Retrieval (IR) based Chatbot

Xiaoice



Information Retrieval (IR) based Chatbot

1. Return the response to the most similar turn
 - Take user's turn (q) and find a (tf-idf) similar turn t in the corpus C

$q = \text{"do you like Doctor Who"}$
 $t = \text{"do you like Doctor Strange"}$

- Grab whatever the response was to t .

$$r = \text{response} \left(\operatorname{argmax}_{t \in C} \frac{q^T t}{\|q\| \|t\|} \right) \quad \text{Yes, love it!}$$

2. Return the most similar turn

$$r = \operatorname{argmax}_{t \in C} \frac{q^T t}{\|q\| \|t\|} \quad \text{Do you like Doctor Strangelove?}$$

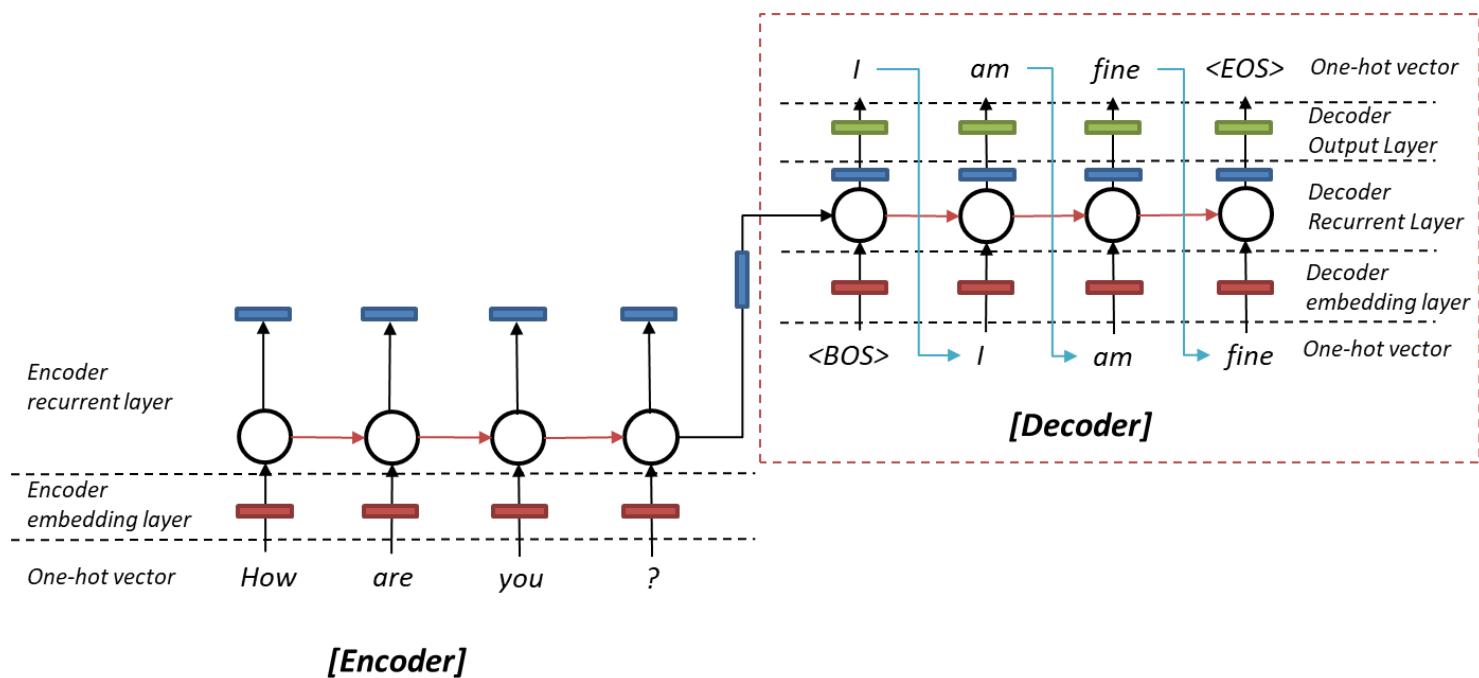
Information Retrieval (IR) based Chatbot

1. Also fine to use other features like user features, or prior turns
2. Or non-dialogue text
 - COBOT chatbot (Isbell et al., 2000)
 - sentences from the Unabomber Manifesto by Theodore Kaczynski, articles on alien abduction, the scripts of “The Big Lebowski” and “Planet of the Apes”.
3. Wikipedia text

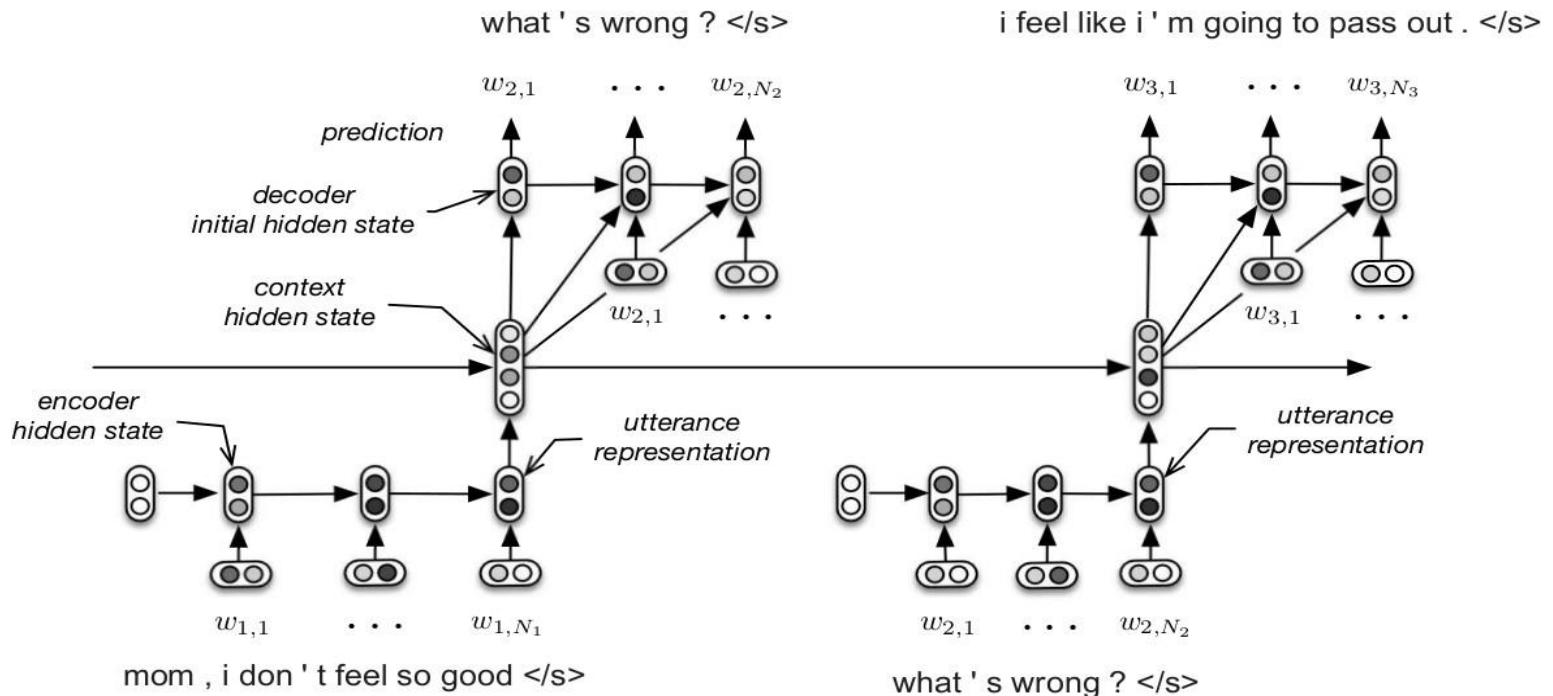
Deep-learning Chatbots

- Think of response generation as a task of transducing from the user's prior turn to the system's turn.
- Train on:
 - Movie Dialogs
 - Twitter Conversations
- Train a deep neural network
 - Map from user 1 turn to user 2 response

Seq2seq model architecture



Seq2seq model architecture



5 NLG Tasks

Deep learning chatbots

Trained on 127M Twitter context-message-response triples

#	Context c	Message m	Generated Response r
1	noone can help this time its just i need a support system	well im here to support you. whenever you need me.	thank you, i appreciate it thank you
2	nothin much, and how's the book?!	its good but i'm only like halfway through cuz i don't feel like reading. i'm so bored ...	that's good! i have the book but i'm bored too.
3	i know! what have you been up to?	nothing really. just enjoying summer and working. what about you?	that's good! nothing much just listening to music and you?
4	thanks, your way better though	i'm just good at expressing it that's but trust me you are freakin amazing and waay better	i know, i don't trust you, i love you too! thank you
5	thanks. yeah i am good thanks! good, but angry!	sorry you're angry, been there, i hope it dies down.	i hope so too, i'm not angry!!
6	figures. bass. never ate bass.	bass is extremely mild with very few bones. my favorite after catfish	that's one of the bones, chuck bass is mine too.
7	today i will walk for miles. clear sky clear mind.	love walking. it's too hot to walk this month. i haven't moved. frozen. heat freeze.	it's too hot for you to get up to honey if you have frozen yogurt to you.
8	arresting people over "nasty tweets"? please. i'm not condoning what the lad did but arresting him is a tad silly.	it appears he may have sent death threats to others while all this was kicking off too.	i did! he was armed with nuclear threats? that's what's happening to you.

Neural based NLG in Dialog: Issue

- Problem: became apparent that a naïve application of standard seq2seq methods has serious pervasive deficiency for (chitchat) dialogue:
 - Either because it's generic (e.g. "I don't know")
 - Or because changing the subject to something unrelated
 - Boring response
 - Repetition problem
 - Lack of consistent persona problem

What else do we have?

Template-based generation

- The most common approach in spoken natural language generation.
- In simplest form, words fill in slots:

“Flights from *ORIGIN* to *DEST* on *DEPT_DATE DEPT_TIME*. Just one moment please”

<i>Slot</i>	<i>Type</i>	<i>Question</i>
<i>ORIGIN</i>	<i>city</i>	<i>What city are you leaving from?</i>
<i>DEST</i>	<i>city</i>	<i>Where are you going?</i>
<i>DEPT DATE</i>	<i>date</i>	<i>What day would you like to leave?</i>
<i>DEPT TIME</i>	<i>time</i>	<i>What time would you like to leave?</i>
<i>AIRLINE</i>	<i>line</i>	<i>What is your preferred airline?</i>

- Most common NLG used in commercial systems
- Used in conjunction with concatenative TTS (text-to-speech) to make natural sounding output

Template-based generation

Pros

- Conceptually Simple: No specialized knowledge required to develop
- Tailored to the domain, so often good quality

Cons

- Lacks generality: Repeatedly encode linguistic rules (e.g. subject-verb agreement)
- Little variation in style
- Difficult to grow/maintain: Each utterance must be manually added

Improvement?

- Need deeper utterance representations
- Linguistic rules to manipulate them

Rule-based Generation



Content Planning

- What information must be communicated?
 - Content selection and ordering

Sentence Planning

- What words and syntactic constructions will be used for describing the content?
 - Aggregation: What elements can be grouped together for more natural-sounding, succinct output?
 - Lexicalisation: What word are used to express the various entities?

Realisation

- How is it all combined into a sentence that is syntactically and morphologically correct?

Rule-based Generation

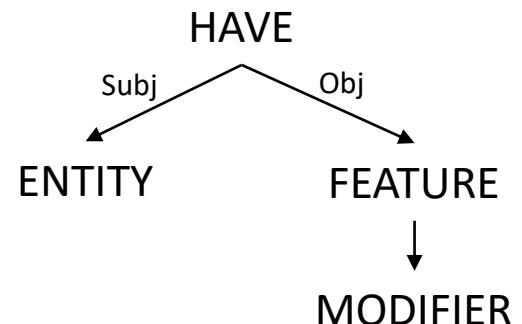
Assume that the dialog system need to tell the user about the restaurant

Content Planning

- Select Information ordering
 - has(sushitrain, crusine(bad))
 - has(sushitrain, decor(good))

Sentence Planning

- Choose *syntactic templates*
- Choose lexicon
 - Bad → awful; crusine → food quality
 - Good → excellent; decor → décor
- Generate expressions
 - Entity → this restaurant



Realisation

- Choose correct verb: HAVE → has
- No article needed for feature names

“This restaurant has awful food quality but excellent décor”

Summary

Goal-oriented Conversational Agent:

- Ontology + hand-written rules for slot fillers
- Machine learning classifiers to fill slots

Chatbots:

- Simple rule-based systems
- IR-based: mine datasets of conversations.
- Neural net models with more data

The future...

- Need to acquire that data
- Integrate goal-based and chatbot-based systems

Summarisation: two strategies

Extractive Summarisation

- Select parts (typically sentences) of the original text to form a summary.



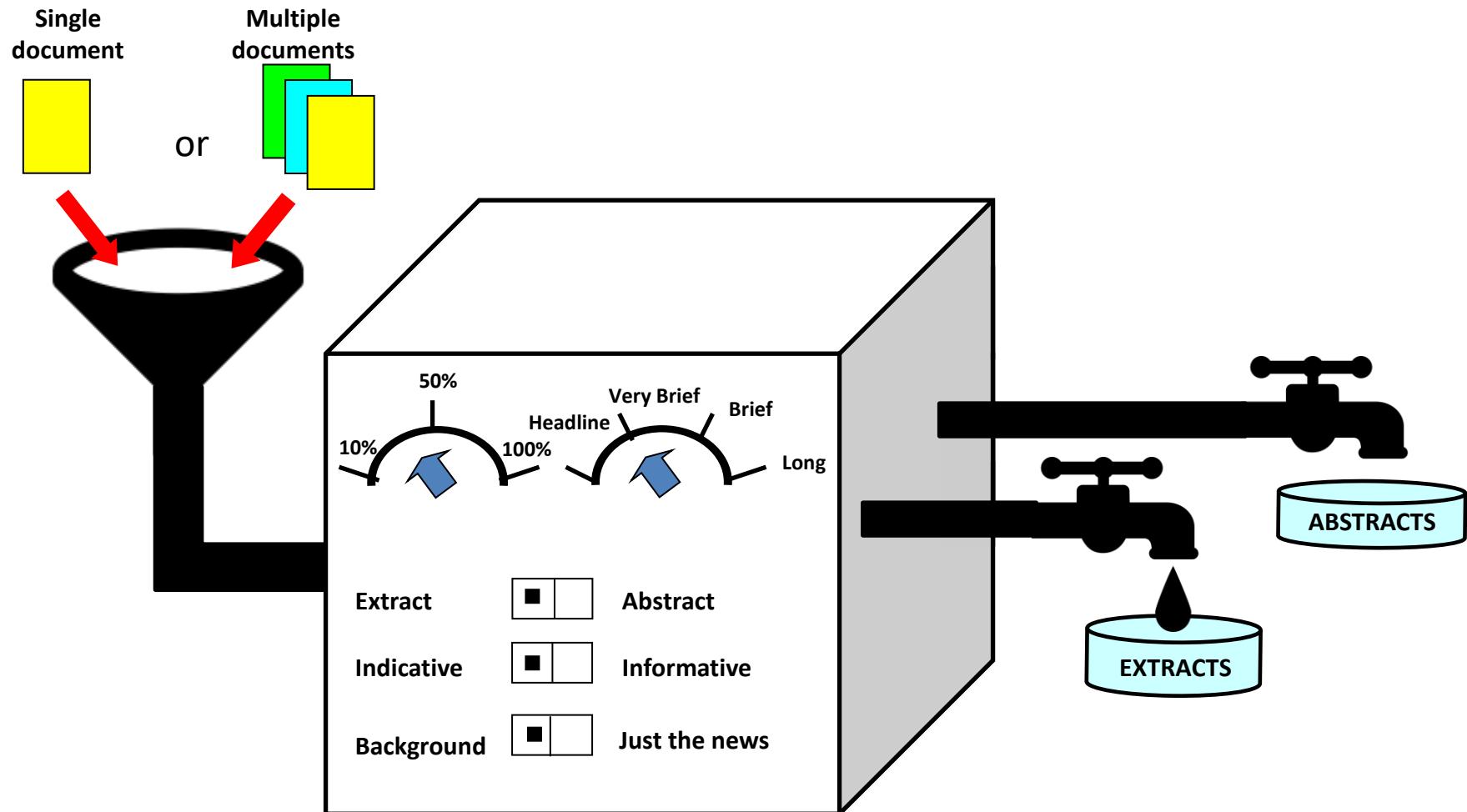
Abstractive Summarisation

- Generate new text using natural language generation techniques.

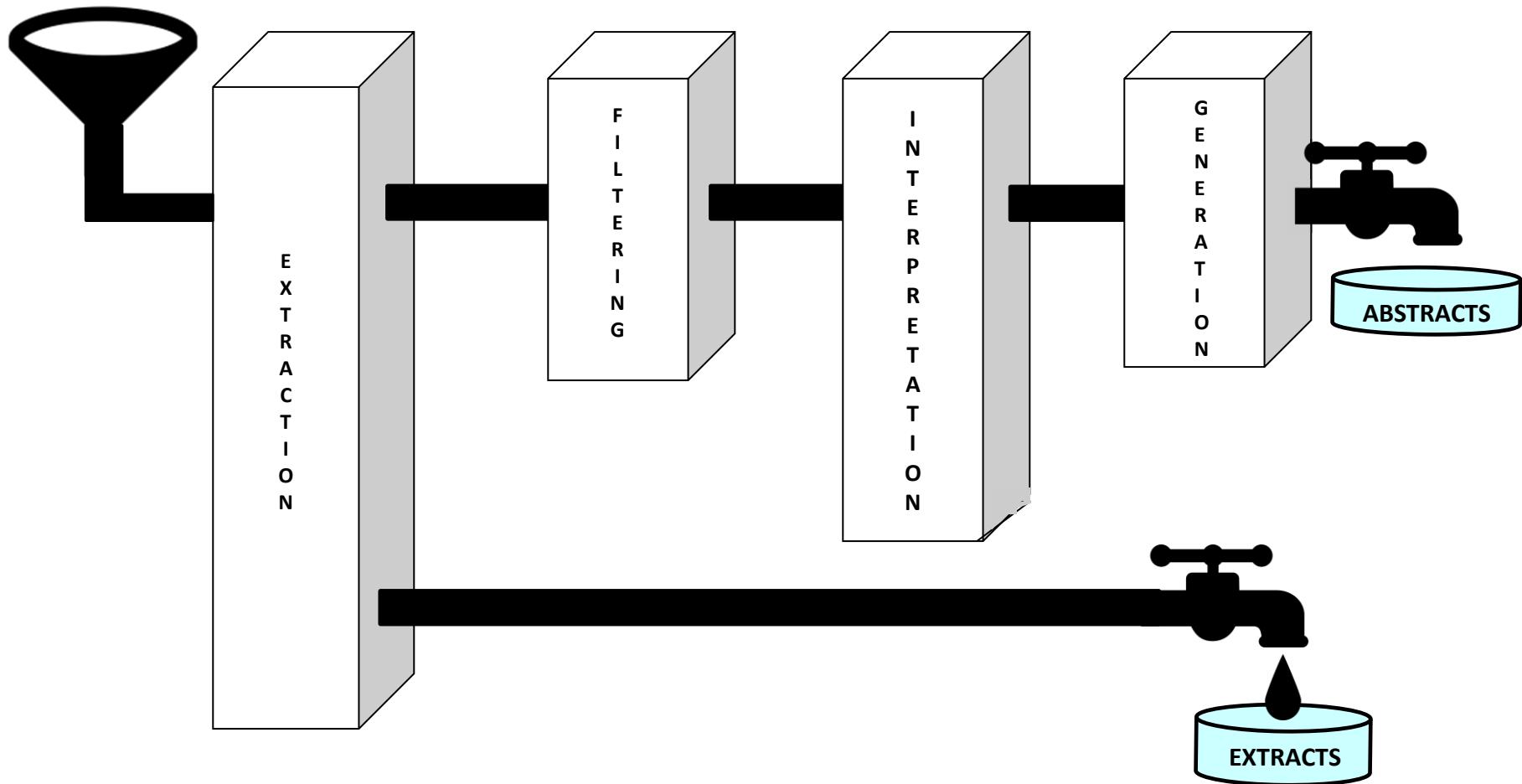


5 NLG Tasks

Summarisation: two strategies



Summarisation: two strategies



Other NLG Tasks: Visual StoryTelling (Kim et al., NAACL 2018)



the bride and groom are ready for their big day .

they are getting married .

the brides maids pose together .

the ceremony is a big moment .

the flower girl is having a blast .

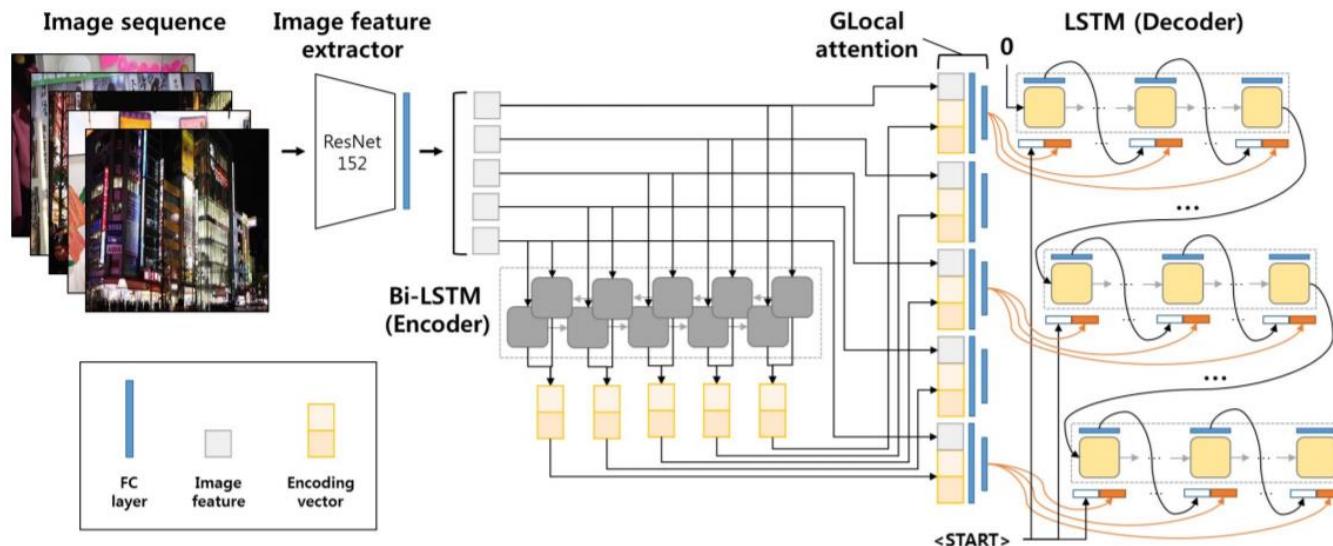


Figure 2: The global-local attention cascading (GLAC) network model for visual story generation. Note: activation function (ReLU), dropout, batch normalization, and softmax layer are omitted for readability.

0 LECTURE PLAN

Lecture 8: Language Model and Natural Language Generation

1. Language Model
2. Traditional Language Model
3. Neural Language Model
4. Natural Language Generation
5. NLG Tasks
6. **Language Model and NLG Evaluation**

How to evaluate the Language Model?

*The standard evaluation metric for Language Models is **perplexity**.*



Perplexed

perplex

/pə'plɛks/

verb

past tense: **perplexed**; past participle: **perplexed**

make (someone) feel completely baffled.
"she was perplexed by her husband's moodiness"

Similar: [puzzle](#) [baffle](#) [mystify](#) [bewuse](#) [bewilder](#) [confound](#) [confuse](#) ▾

• **DATED**
complicate or confuse (a matter).
"they were perplexing a subject plain in itself"

So, Lower Perplexity is better!

How to evaluate the Language Model?

*The standard evaluation metric for Language Models is **perplexity**.*



Perplexed

perplex
 /pə'plɛks/
 verb
 past tense: **perplexed**; past participle: **perplexed**
 make (someone) feel completely baffled.
 "she was perplexed by her husband's moodiness"
 Similar: [puzzle](#) [baffle](#) [mystify](#) [bewuse](#) [bewilder](#) [confound](#) [confuse](#) ▾
 • **DATED**
 complicate or confuse (a matter).
 "they were perplexing a subject plain in itself"

$$\text{perplexity} = \prod_{t=1}^T \left(\frac{1}{P_{\text{LM}}(\mathbf{x}^{(t+1)} | \mathbf{x}^{(t)}, \dots, \mathbf{x}^{(1)})} \right)^{1/T}$$

*Normalized by
number of words*

Inverse probability of corpus, according to Language Model

6 Language Model and NLG Evaluation

How to evaluate the Language Model?

Language Model Approaches Performance Evaluated by Facebook Research

Model	Perplexity	
Interpolated Kneser-Ney 5-gram (Chelba et al., 2013)	67.6	<i>N-gram model</i>
RNN-1		
RNN-2		
Sparse Non-negative Matrix Factorization (Shazeer et al., 2015)	52.9	
LSTM-2048 (Jozefowicz et al., 2016)	43.9	<i>Neural model</i>
2-layer LSTM-8		
Ours small (LSTM-2048)	43.9	
Ours large (2-layer LSTM-2048)	39.8	

Can we use the perplexity for NLG Evaluation?

*No. Captures how powerful your LM is,
but doesn't tell you anything about generation*

Table 2. Comparison on 1B word in perplexity (lower the better). Note that Jozefowicz et al., uses 32 GPUs for training. We only use 1 GPU.

How to evaluate the Natural Language Generation?

Unfortunately, No automatic metrics to adequately capture overall quality

There are some metrics to capture particular aspects of generated text:

- *Fluency (compute probability - well-trained Language model)*
- *Correct style (Language Model trained on target corpus)*
- *Diversity (rare word usage, uniqueness of n-grams)*
- *Relevance to input (semantic similarity measures)*
- *Simple things like length and repetition*
- *Task-specific metrics e.g. compression rate for summarization*
- *Though these don't measure overall quality, they can help us track some important qualities that we care about.*

How to evaluate the Natural Language Generation?

Human Evaluation

- *Human judgments are regarded as the gold standard*
- *Of course, we know that human eval is slow and expensive*
- *Supposing you do have access to human evaluation: Does human evaluation solve all of your problems?*

Humans ...

- *are inconsistent*
- *can be illogical*
- *lose concentration*
- *misinterpret your question*
- *can't always explain why they feel the way they do*

Language Model and NLG Evaluation

Natural Language Generation: Long way to go



/ Reference

Reference for this lecture

- Deng, L., & Liu, Y. (Eds.). (2018). Deep Learning in Natural Language Processing. Springer.
- Rao, D., & McMahan, B. (2019). Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. " O'Reilly Media, Inc.".
- Manning, C. D., Manning, C. D., & Schütze, H. (1999). Foundations of statistical natural language processing. MIT press.
- Manning, C 2018, Natural Language Processing with Deep Learning, lecture notes, Stanford University
- Li, J., Galley, M., Brockett, C., Gao, J., & Dolan, B. (2015). A diversity-promoting objective function for neural conversation models. arXiv preprint arXiv:1510.03055.
- Jiang, S., & de Rijke, M. (2018). Why are Sequence-to-Sequence Models So Dull? Understanding the Low-Diversity Problem of Chatbots. arXiv preprint arXiv:1809.01941.
- Liu, C. W., Lowe, R., Serban, I. V., Noseworthy, M., Charlin, L., & Pineau, J. (2016). How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. arXiv preprint arXiv:1603.08023.

COMP5046

Natural Language Processing

*Lecture 10: Attention and Question Answering
(Reading Comprehension)*

Dr. Caren Han

Semester 1, 2022

School of Computer Science,
University of Sydney



0 LECTURE PLAN

Lecture 10: Attention and Question Answering (Reading Comprehension)

1. Question Answering
2. Knowledge-based Question Answering
3. IR-based Question Answering (Reading Comprehension)
4. Attention
5. Reading Comprehension with Attention
6. Visual Question Answering

Question Answering

Question Answering

Question answering (QA) is a computer science discipline within the fields of information retrieval and natural language processing (NLP), which is concerned with building systems that **automatically answer questions posed by humans in a natural language**.

Different types of questions:

General questions, with Yes/No answers

- e.g. Are you a student?

Wh- Questions, start with: *who, what, where, when, why, how, how many*

- e.g. When did you get to this lecture?
- e.g. What is the weather like in London?



Question Answering

Question Answering

Question answering (QA) is a computer science discipline within the fields of information retrieval and natural language processing (NLP), which is concerned with building systems that **automatically answer questions posed by humans in a natural language**.

Different types of questions:

Choice Questions, where you have some options inside the question

Factoid questions, where the complete answer can be found inside a text. The answer to such questions consist of one or several words that go one after another



Question Answering

Question

Three Questions for building a QA System

- What do the answers look like?
- Where can I get the answers from?
- What does my training data look like?

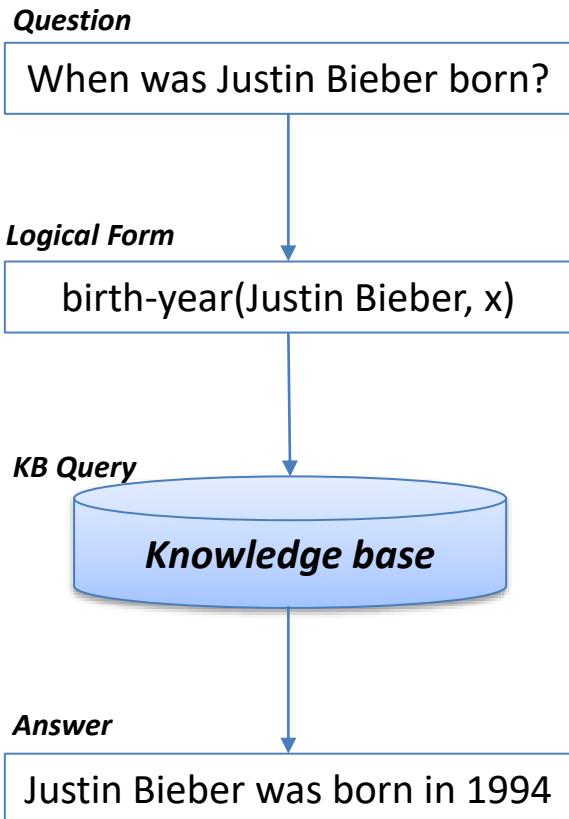
Question Answering Research Areas

Research Areas in Question Answering

Research Area	Details
Knowledge-based QA (Semantic Parsing)	<ul style="list-style-type: none">• Answer is a logical form, possibly executed against a Knowledge Base• Context is a Knowledge Base
Information Retrieval-based QA <ul style="list-style-type: none">• Answer sentence selection• Reading Comprehension	<ul style="list-style-type: none">• Answer is a document, paragraph, sentence• Context is a corpus of documents or a specific document
Visual QA	<ul style="list-style-type: none">• Answer is simple and factual• Context is one/multiple image(s)
Library Reference	<ul style="list-style-type: none">• Answer is another question• Context is the structured knowledge available in the library and the librarians' view of it.

Semantic Parsing

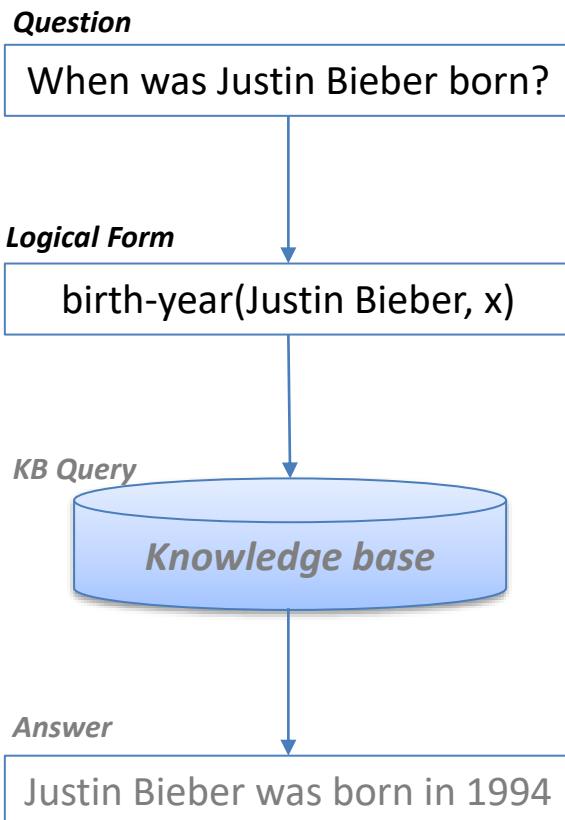
Answering a natural language question by mapping it to a query over a structured database (formal representation of its meaning).



Knowledge-based Question Answering

Semantic Parsing

Answering a natural language question by mapping it to a query over a structured database (formal representation of its meaning).



Mapping from a text string to any logical form

Question	Logical Form
When was Justin Bieber born?	birth-year(Justin Bieber, x)
What is the largest state?	$\text{argmax}(\lambda x.\text{state}(x), \lambda x.\text{size}(x))$

How to map? Map either to some version of predicate calculus or a query language like SQL or SPARQL
<https://query.wikidata.org/>

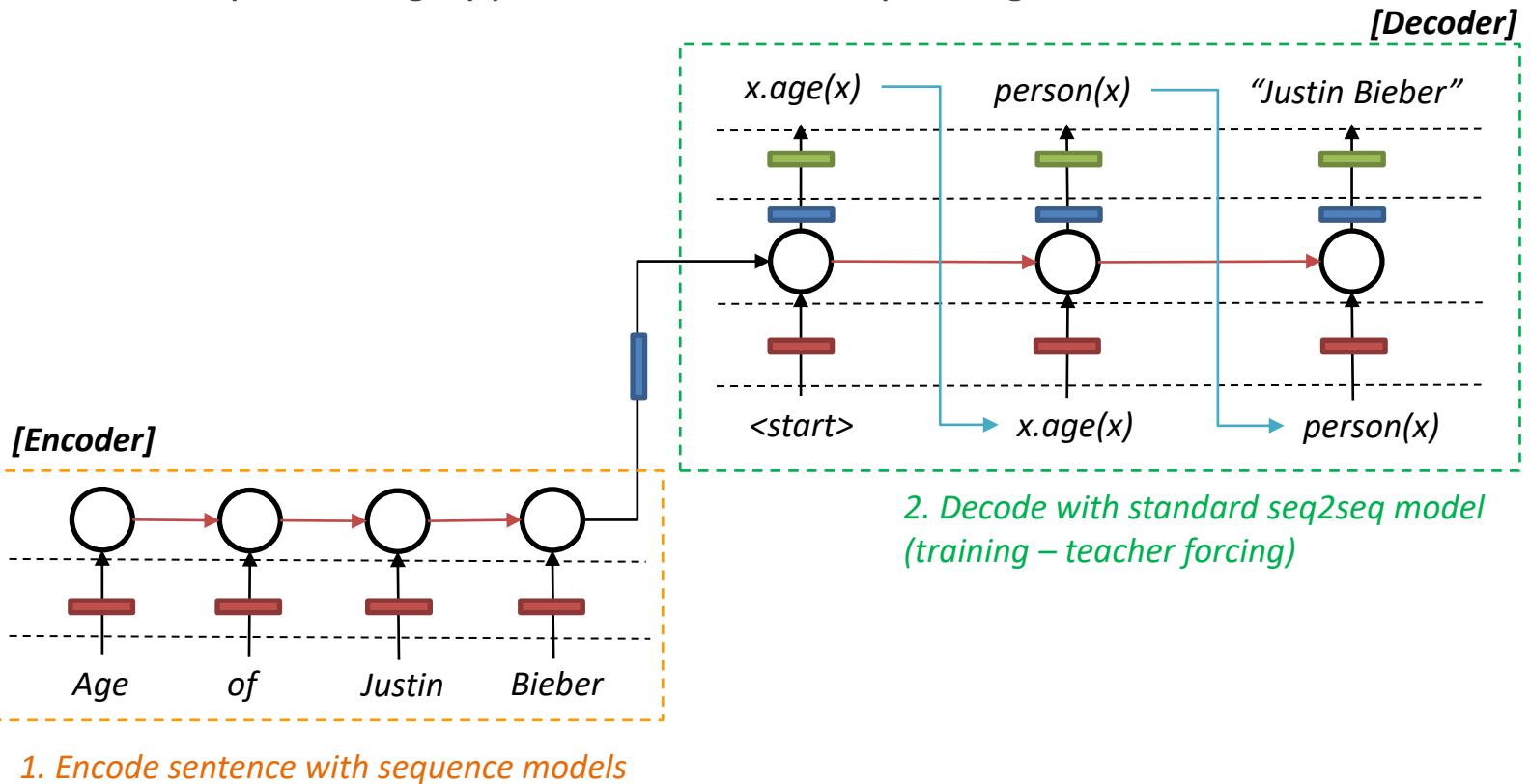
Use expensive supervised data?

Require experts for the manual annotation process....

Seq2Seq model for semantic parser

How to transfer the text to the logical form?

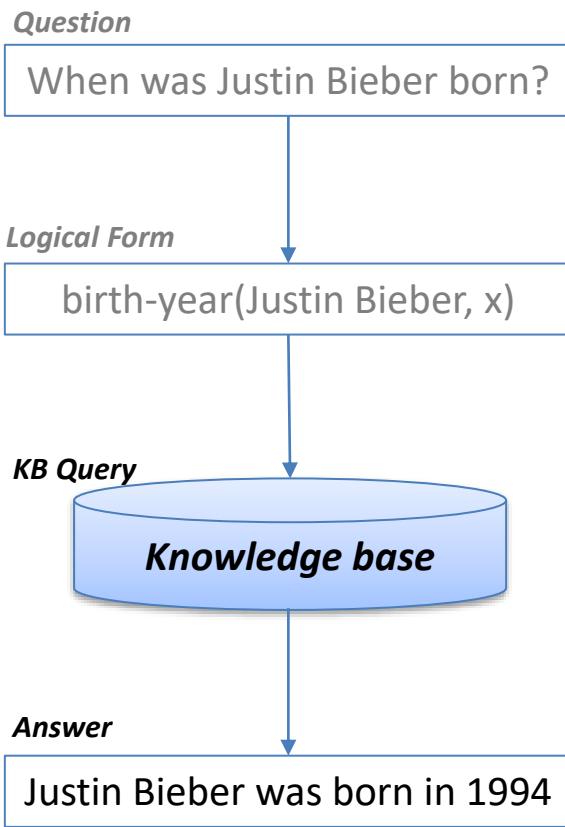
A basic deep learning approach to semantic parsing



Knowledge-based Question Answering

Semantic Parsing

Answering a natural language question by mapping it to a query over a structured database (formal representation of its meaning).



Answer questions that ask about one of the missing arguments in a triple

Subject	Predicate (relation)	Object
Justin Bieber	birth-year	1994
Frédéric Chopin	birth-year	1810
...

- DBpedia
- Freebase

How to produce the answer?

- Seq2seq
- Template based generation

Knowledge-based Question Answering

Pros and Cons of Knowledge-based QA

- *Logical Form instead of (direct) answer makes system robust*
- *Answer independent of question and parsing mechanism*
- *Constrained to queriable questions in Database Schema*
- *Difficult to find the well-structured training dataset*

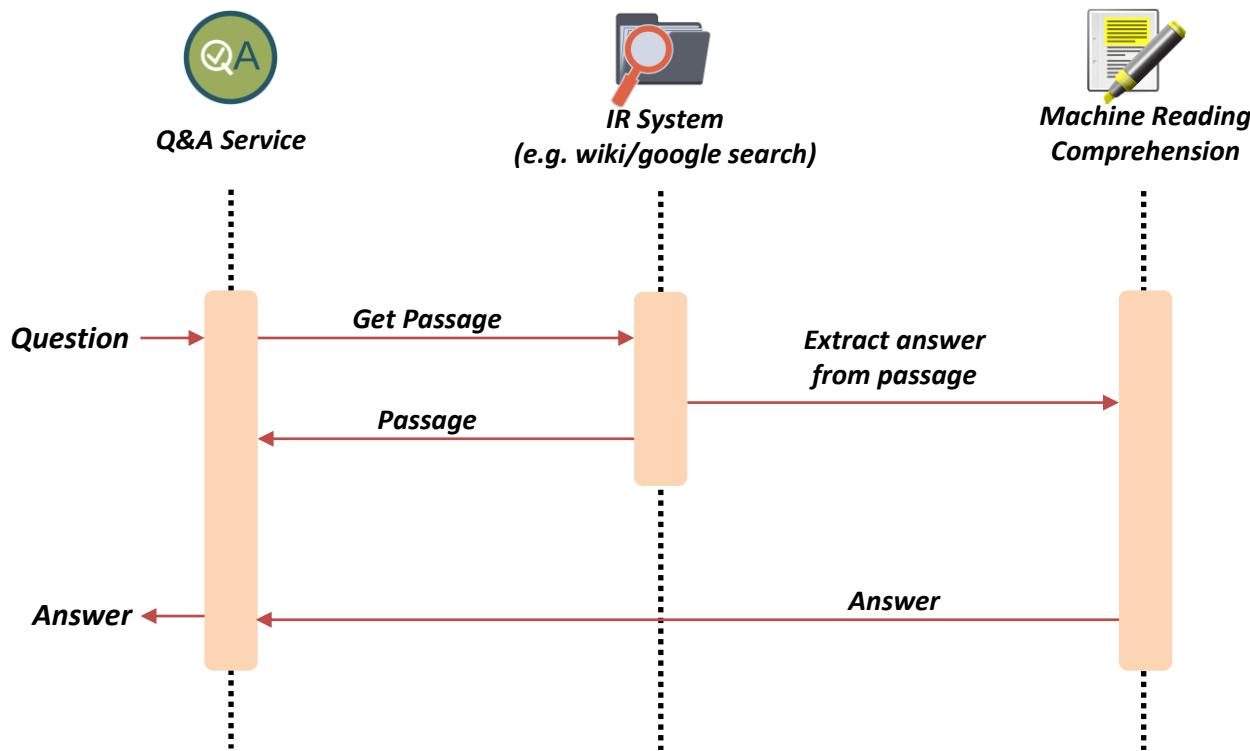
0 Question Answering

Research Areas in Question Answering

Research Area	Details
Knowledge-based QA (Semantic Parsing)	<ul style="list-style-type: none">• Answer is a logical form, possibly executed against a Knowledge Base• Context is a Knowledge Base
Information Retrieval-based QA <ul style="list-style-type: none">• Answer sentence selection• Reading Comprehension	<ul style="list-style-type: none">• Answer is a document, paragraph, sentence• Context is a corpus of documents or a specific document
Visual QA	<ul style="list-style-type: none">• Answer is simple and factual• Context is one/multiple image(s)
Library Reference	<ul style="list-style-type: none">• Answer is another question• Context is the structured knowledge available in the library and the librarians' view of it.

Information Retrieval-based Question Answering

Answering a user's question by finding short text segments, sentences, or documents on the web or collection of document



3 Reading Comprehension

Information Retrieval-based Question Answering

*Answering a user's question by **finding short text segments, sentences, or documents** on the web or collection of document*

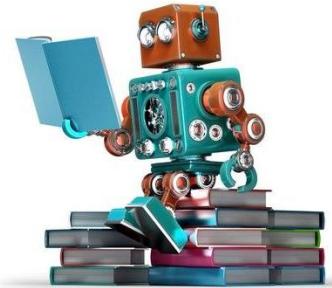
- ***Reading Comprehension and Answer Sentence Selection:***
 - *Finding an answer in a paragraph or a document*
 - *Picking a suitable sentence from a corpus that can be used to answer a question*

3 Reading Comprehension

Reading Comprehension

To answer these questions, you need to first gather information by collecting answer-related sentences from the article.

Can we teach this to machine?



Yes, we can!

Machine Comprehension of Text
(Burges 2013)

THE BOAT PARADE

The boats are floating along the lakeshore. It is the summer boat parade.

There are motor boats, rowboats and sailboats.

Jessica's favorite is the yellow motor boat with the flag. The rowboat decorated with flowers is Lisa's favorite. Tony likes the purple sailboat.

The boats float by one at a time. The people on the boats waive at the crowds. The crowds cheer the boats.

The boat parade is so much fun to watch. It is the best part of the summer.

Answer the Questions:

1. Where are the boats floating?
2. What kind of boats are there?
3. What is Lisa's favorite boat?

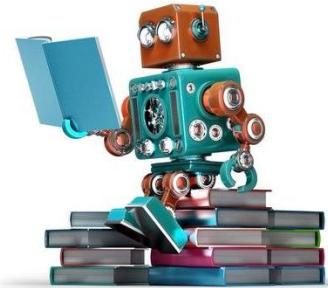


3 Reading Comprehension

Reading Comprehension

To answer these questions, you need to first gather information by collecting answer-related sentences from the article.

Can we teach this to machine?



A machine comprehends a passage of text if, for any question regarding that text that can be answered correctly by a majority of native speakers

THE BOAT PARADE

The boats are floating along the lakeshore. It is the summer boat parade.

There are motor boats, rowboats and sailboats.

Jessica's favorite is the yellow motor boat with the flag. The rowboat decorated with flowers is Lisa's favorite. Tony likes the purple sailboat.

The boats float by one at a time. The people on the boats waive at the crowds. The crowds cheer the boats.

The boat parade is so much fun to watch. It is the best part of the summer.

Answer the Questions:

1. Where are the boats floating?
2. What kind of boats are there?
3. What is Lisa's favorite boat?

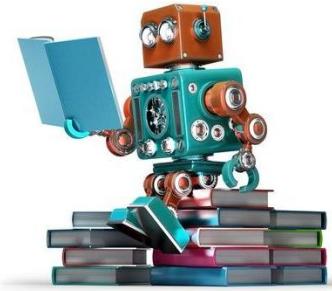


3 Reading Comprehension

Reading Comprehension

To answer these questions, you need to first gather information by collecting answer-related sentences from the article.

Why do we need to teach this?



The ability to comprehend text will lead us to a better search and solve lots of NLP problems!

THE BOAT PARADE

The boats are floating along the lakeshore. It is the summer boat parade.

There are motor boats, rowboats and sailboats.

Jessica's favorite is the yellow motor boat with the flag. The rowboat decorated with flowers is Lisa's favorite. Tony likes the purple sailboat.

The boats float by one at a time. The people on the boats waive at the crowds. The crowds cheer the boats.

The boat parade is so much fun to watch. It is the best part of the summer.

Answer the Questions:

1. Where are the boats floating?
2. What kind of boats are there?
3. What is Lisa's favorite boat?



3 Reading Comprehension

Corpora for Reading Comprehension

Dataset	Answer Type	Domain
MCTest (Richardson et al. 2013)	Multiple choice	Children's stories
CNN/Daily Mail (Hermann et al. 2015)	Spans	News
Children's book test (Hill et al. 2016)	Multiple choice	Children's stories
SQuAD (Rajpurkar et al., 2016)	Spans	Wikipedia
MS MARCO (Nguyen et al., 2016)	Free-from text, Unanswerable	Web Search
NewsQA (Trischler et al., 2017)	Spans	News
SearchQA (Dunn et al., 2017)	Spans	Jeopardy
TriviaQA (Joshi et al., 2017)	Spans	Trivia
RACE (Lai et al., 2017)	Multiple choice	Mid/High School Exams
Narrative QA (Kočiský et al., 2018)	Free-form text	Movie Scripts, Literature
SQuAD 2.0 (Rajpurkar et al., 2018)	Spans, Unanswerable	Wikipedia

3 Reading Comprehension

TriviaQA: A Large Scale Dataset for Reading Comprehension

TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension

The full dataset is coming soon. Here's a sneak peek! The evidence documents come from two domains -- Wikipedia and the web. Click on the "Evidence" button to see the document for each question.

QuestionId	Question	Answer	Web	Wikipedia
qw_3199	Miami Beach in Florida borders which ocean?	Atlantic	Evidence	Evidence
bt_1255	What was the occupation of Lovely Rita according to the song by the Beatles	Traffic Warden	Evidence	Evidence
qg_77	Who was Poopdeck Pappys most famous son?	Popeye	Evidence	Evidence
wh_1026	The Nazi regime was Germany's Third Reich; which was the first Reich?	HOLY ROMAN EMPIRE	Evidence	Evidence
bb_1342	At which English racecourse did two horses collapse and die in the parade ring due to electrocution, in February 2011?	Newbury	Evidence	Evidence
wh_2759	Which type of hat takes its name from an 1894 novel by George Du Maurier where the title character has the surname O'Ferrall ?	TRILBY	Evidence	Evidence
sfq_8522	What was the Elephant Man's real name?	Joseph Merrick	Evidence	Evidence

3 Reading Comprehension

TriviaQA: A Large Scale Dataset for Reading Comprehension

Our UsydNLP achieved the **No.1** in the TriviaQA Leaderboard (Web Setting)!

		Phase description							
		Wikipedia	Web						
		This phase refers to the Web domain described in the paper.							
		Max submissions per day: 3							
		Max submissions total: 100							
		 Download CSV							
Results									
#	User	Entries	Date of Last Entry	Team Name	full-em ▲	full-f1 ▲	verified-em ▲	verified-f1 ▲	
1	mandarjoshi	10	12/02/19	Dataset Author (Oracle Result)	82.99 (1)	87.18 (1)	90.38 (1)	92.96 (1)	
2	usydnlp	11	01/11/21		72.17 (2)	77.36 (2)	84.40 (2)	87.11 (2)	
3	NEUKG	9	08/12/19		69.64 (3)	73.80 (3)	83.36 (3)	85.66 (4)	
4	mingyan	1	02/26/18	SLQA	68.65 (4)	73.07 (5)	82.44 (5)	85.35 (5)	
5	scv.back	4	06/07/18	S3R	68.21 (5)	73.26 (4)	82.57 (4)	86.05 (3)	
6	dirkweissenborn	7	01/15/18		67.46 (6)	72.80 (6)	77.63 (9)	82.01 (8)	
7	S3R	5	02/28/18		66.82 (7)	71.91 (7)	81.01 (6)	84.12 (6)	
8	chrisc	3	09/24/17		66.37 (8)	71.32 (8)	79.97 (7)	83.70 (7)	
9	Mary	5	05/26/20		66.09 (9)	69.78 (9)	79.71 (8)	81.88 (9)	
10	shuohang	3	11/13/17		63.04 (10)	68.53 (10)	69.70 (10)	74.57 (10)	
11	swabha	1	10/27/17	swabha_ankur_tom	53.75 (11)	58.57 (11)	63.20 (11)	66.88 (11)	
12	hux444	2	07/25/17		46.65 (12)	52.89 (12)	56.96 (12)	61.48 (12)	

3 Reading Comprehension

SQuAD: Stanford Question Answering Dataset

Victoria_(Australia)

The Stanford Question Answering Dataset

The economy of Victoria is highly diversified: service sectors including financial and property services, health, education, wholesale, retail, hospitality and manufacturing constitute the majority of employment. Victoria's total gross state product (GSP) is ranked second in Australia, although Victoria is ranked fourth in terms of GSP per capita because of its limited mining activity. Culturally, Melbourne is home to a number of museums, art galleries and theatres and is also described as the "sporting capital of Australia". The Melbourne Cricket Ground is the largest stadium in Australia, and the host of the 1956 Summer Olympics and the 2006 Commonwealth Games. The ground is also considered the "spiritual home" of Australian cricket and Australian rules football, and hosts the grand final of the Australian Football League (AFL) each year, usually drawing crowds of over 95,000 people. Victoria includes eight public universities, with the oldest, the University of Melbourne, having been founded in 1853.

What kind of economy does Victoria have?

Ground Truth Answers: diversified | highly diversified | highly diversified
Prediction: highly diversified

Where according to gross state product does Victoria rank in Australia?

Ground Truth Answers: second | second | second
Prediction: second

At what rank does GPS per capita set Victoria?

Ground Truth Answers: fourth | fourth | fourth
Prediction: fourth

What city in Victoria is called the sporting capital of Australia?

Ground Truth Answers:
Melbourne | Melbourne | Melbourne
Prediction: Melbourne

3 Reading Comprehension

A Generic Neural Model for Reading Comprehension

Step 1: For both documents and questions, convert words to word vectors



Document (D)

A partly submerged glacier cave on Perito Moreno Glacier. The ice facade is approximately 60 m high. Ice formations in the Titlis glacier cave. A glacier cave is a cave formed within the ice of a glacier. Glacier caves are often called ice caves, but the latter term is properly used to describe bedrock caves that contain year-round ice

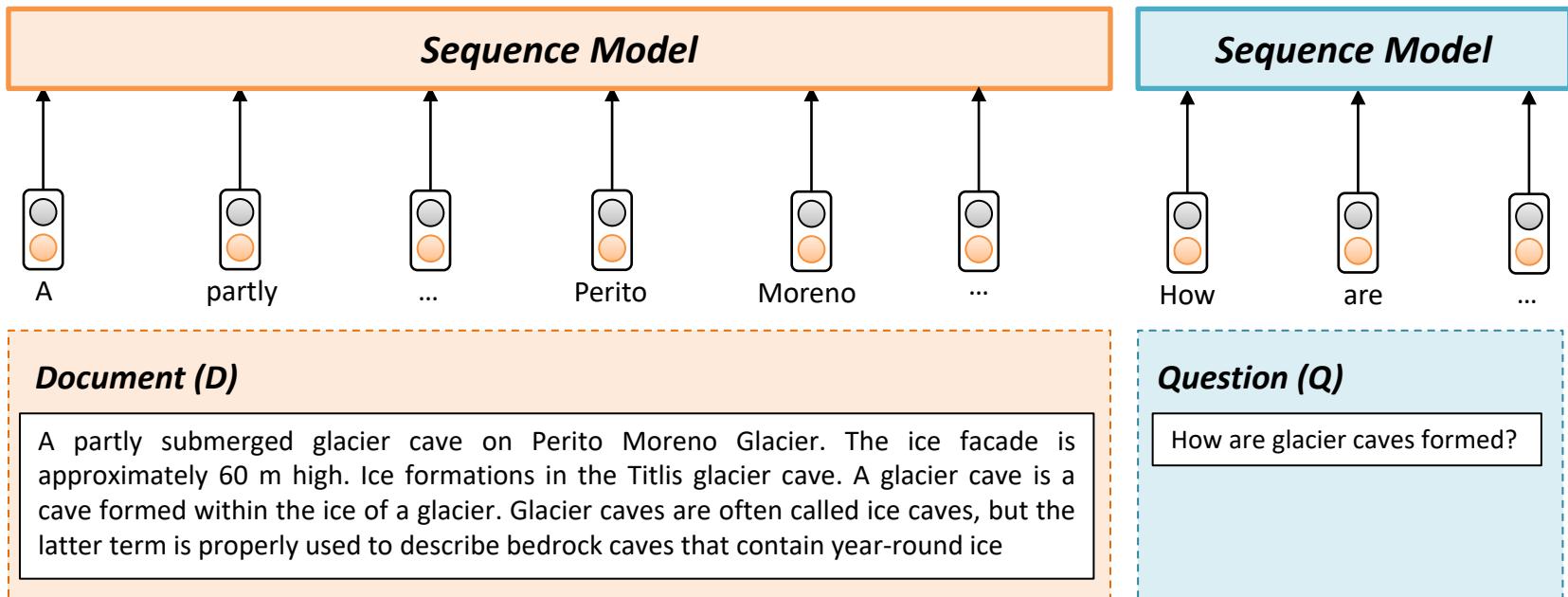
Question (Q)

How are glacier caves formed?

3 Reading Comprehension

A Generic Neural Model for Reading Comprehension

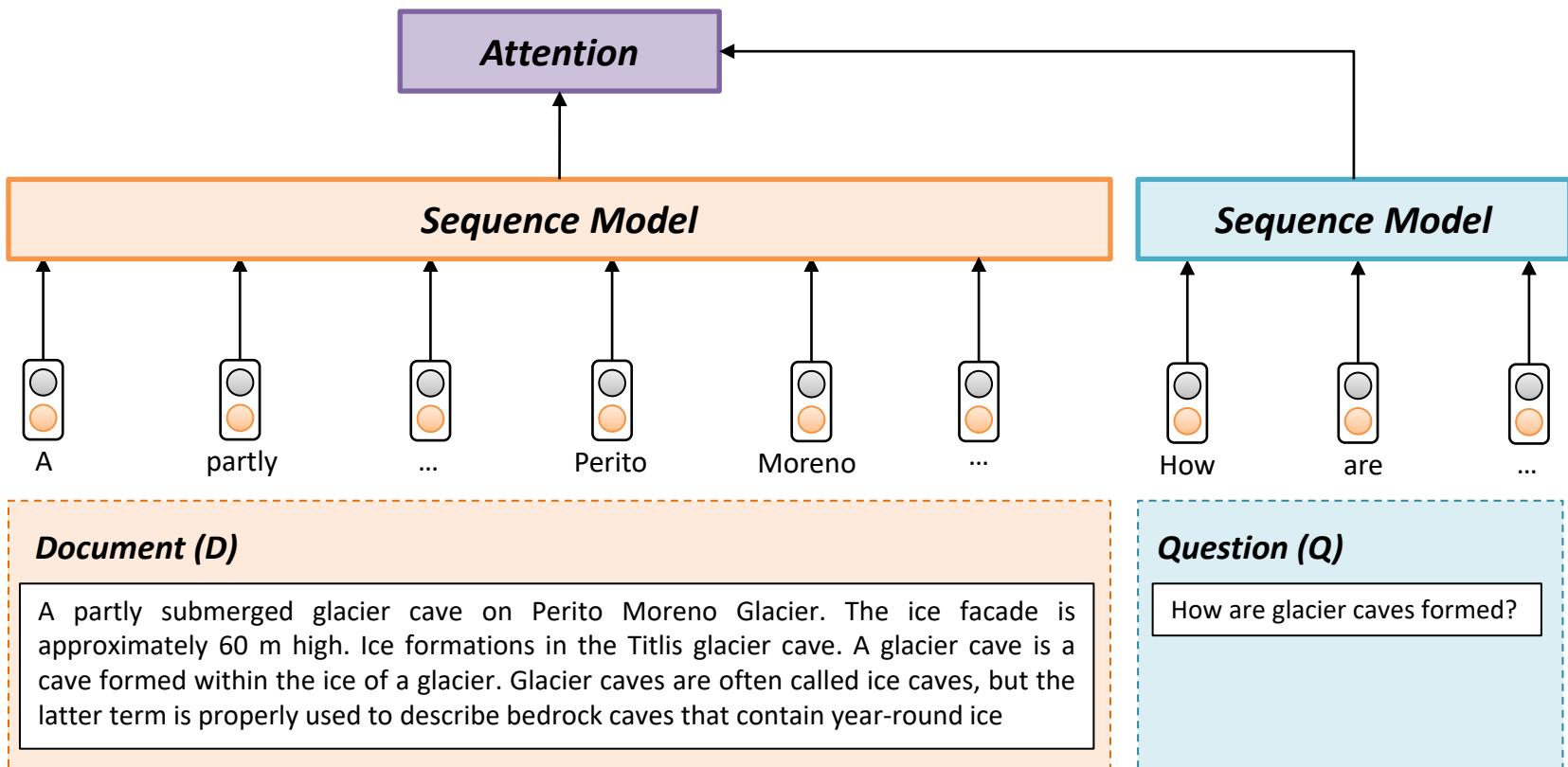
Step2: Encode context (documents) and question with sequence models



3 Reading Comprehension

A Generic Neural Model for Reading Comprehension

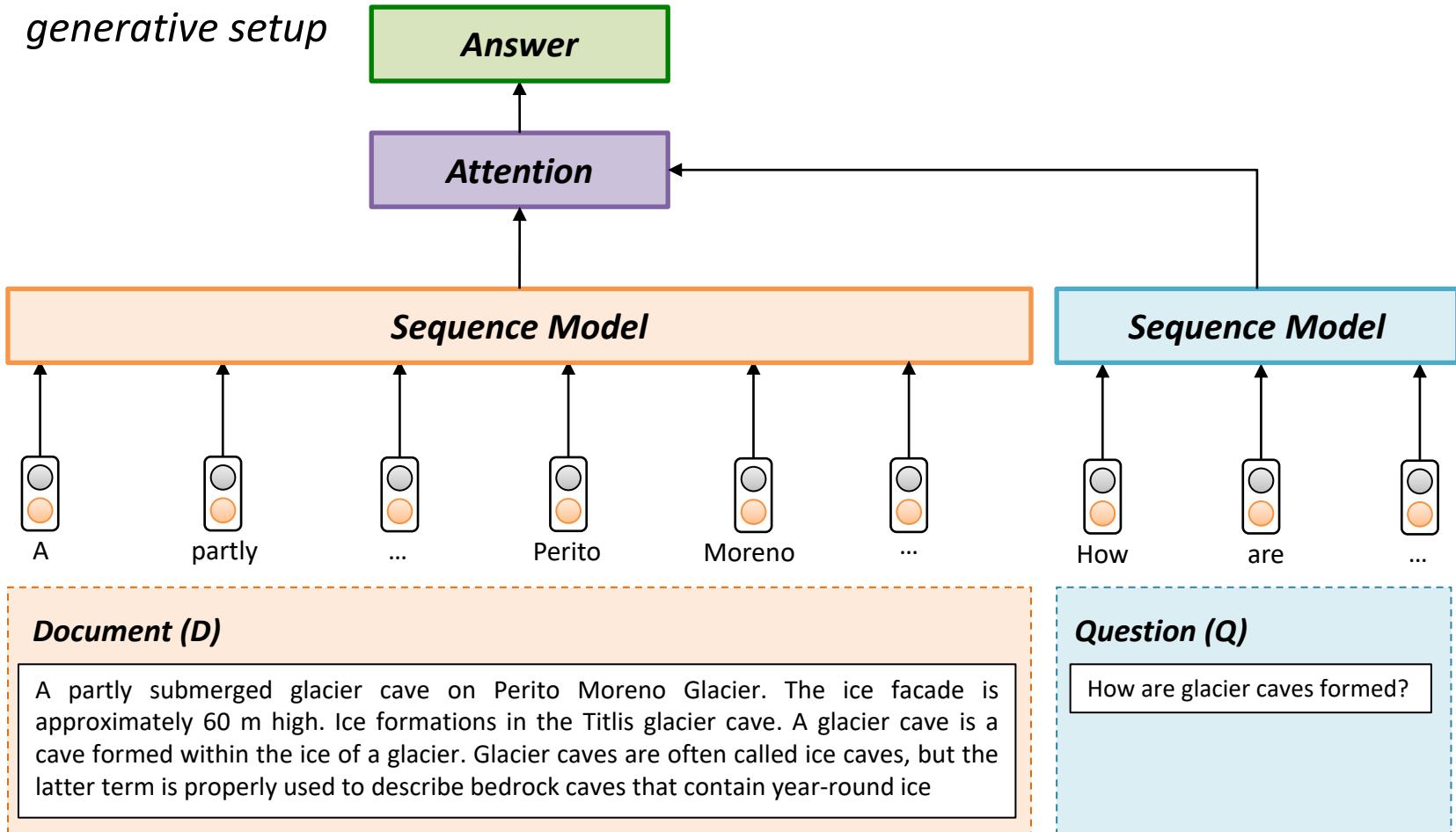
Step 3: Combine *context (documents)* and *question* with an attention



3 Reading Comprehension

A Generic Neural Model for Reading Comprehension

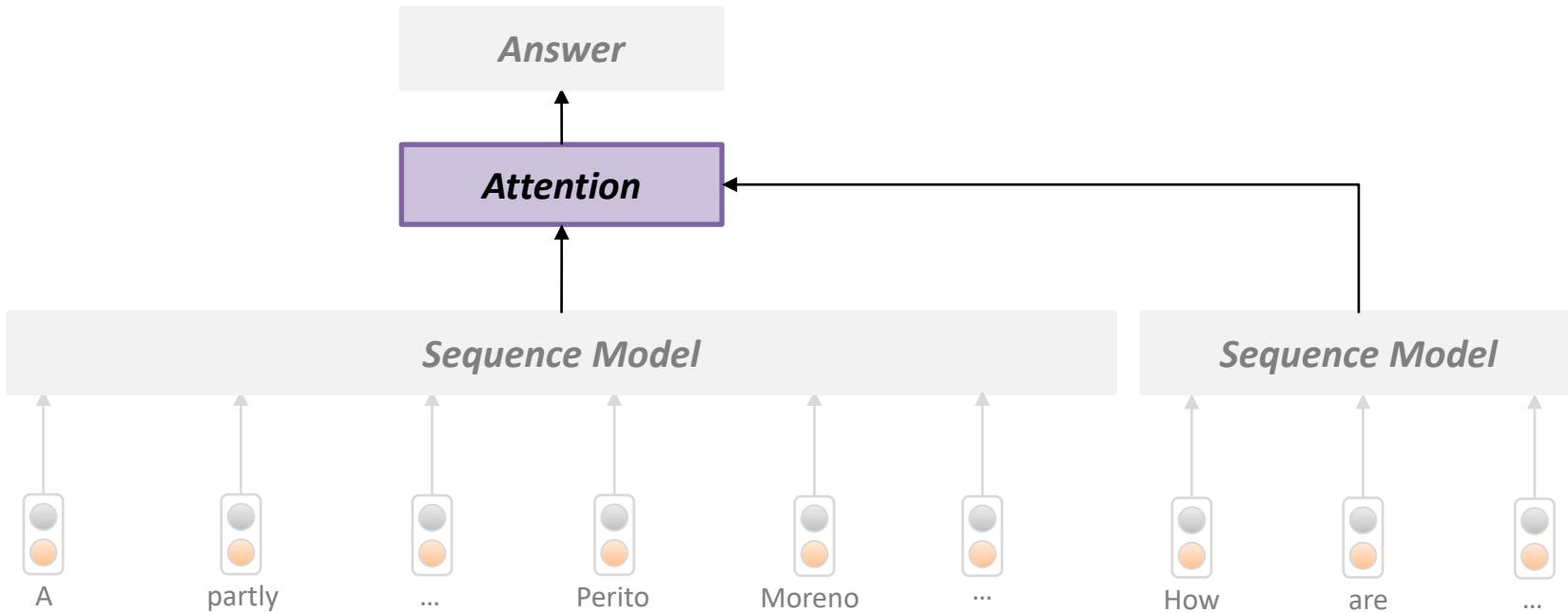
Step4: Select *answer* from attention map by using a classifier or with generative setup



3 Reading Comprehension

A Generic Neural Model for Reading Comprehension

*What is the **Attention**? Why we need this?*



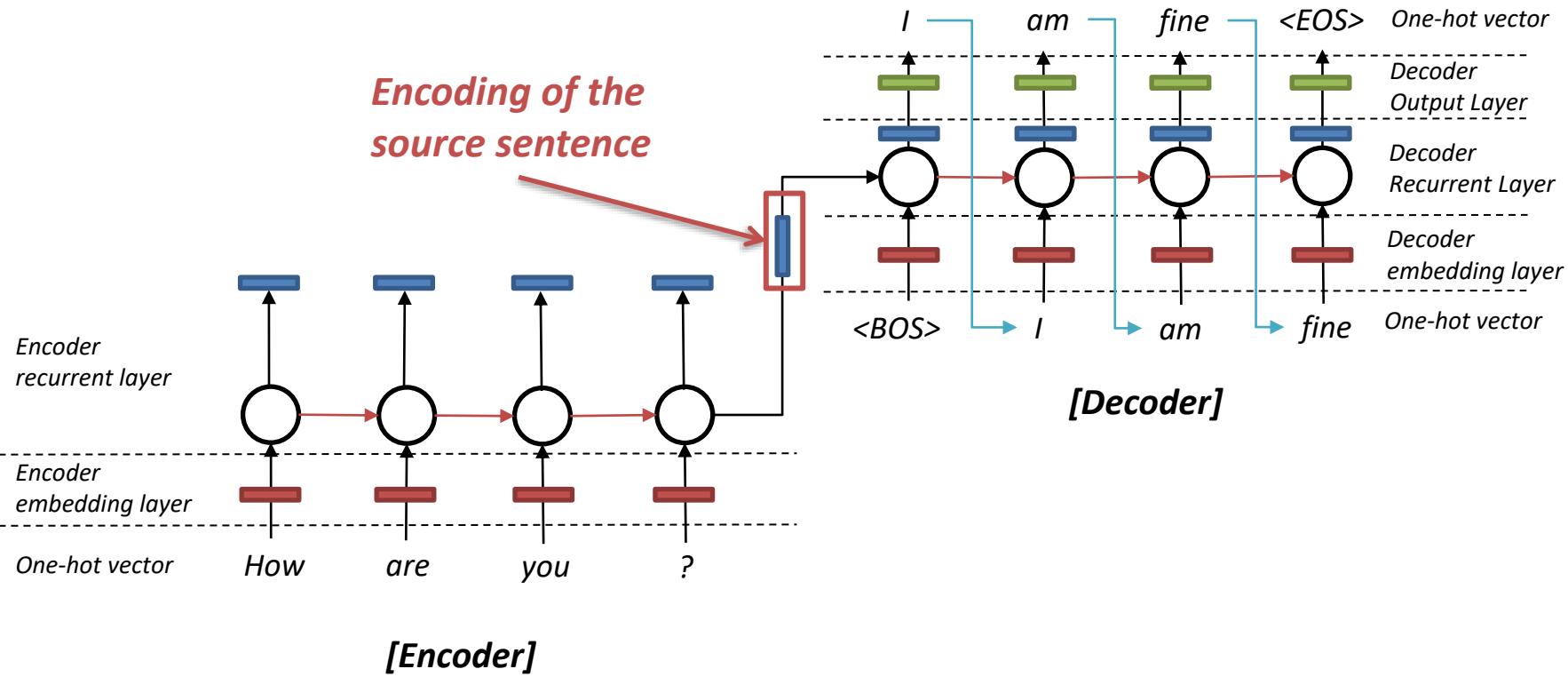
Document (D)

A partly submerged glacier cave on Perito Moreno Glacier. The ice facade is approximately 60 m high. Ice formations in the Titlis glacier cave. A glacier cave is a cave formed within the ice of a glacier. Glacier caves are often called ice caves, but the latter term is properly used to describe bedrock caves that contain year-round ice

Question (Q)

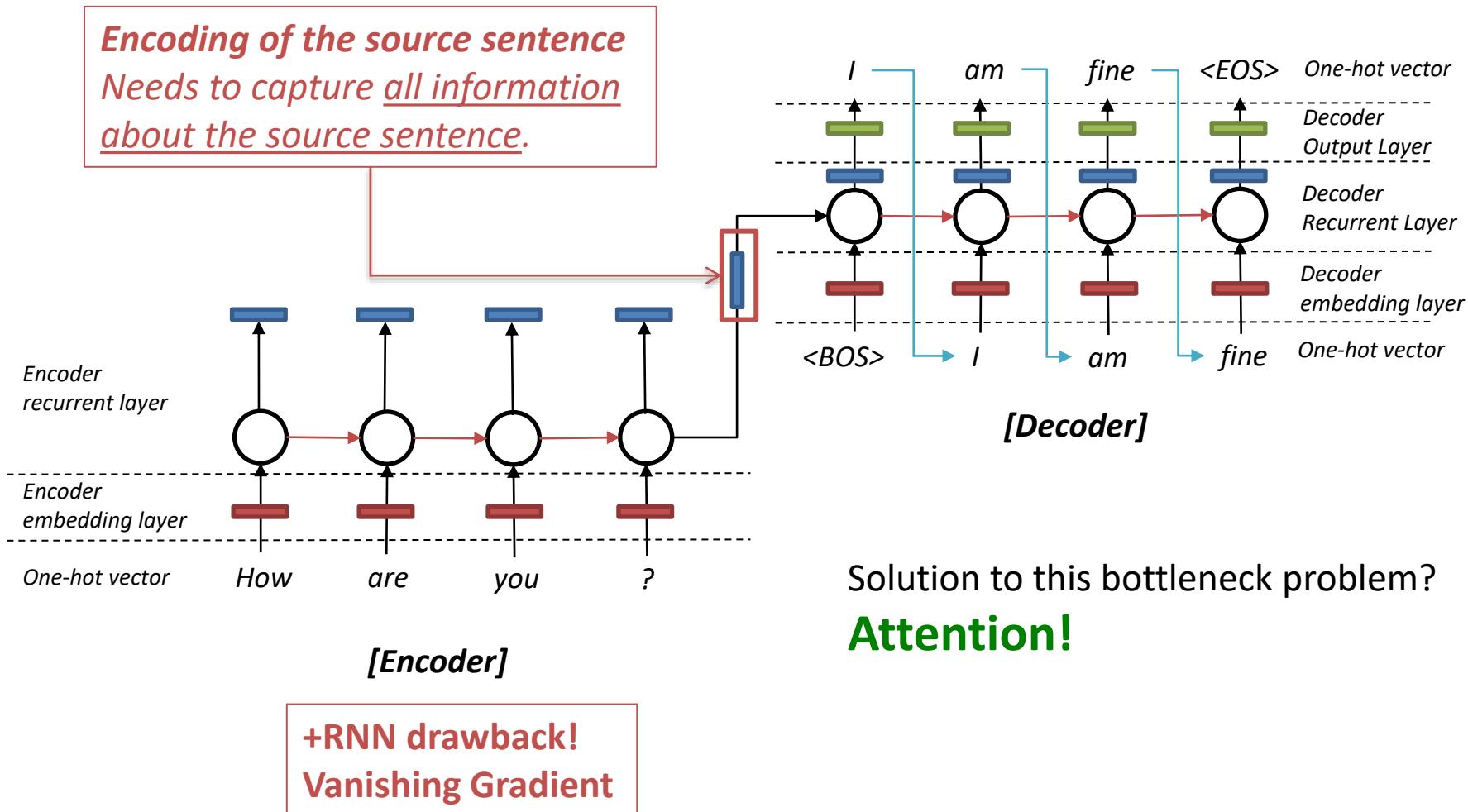
How are glacier caves formed?

Seq2Seq Model: Recap

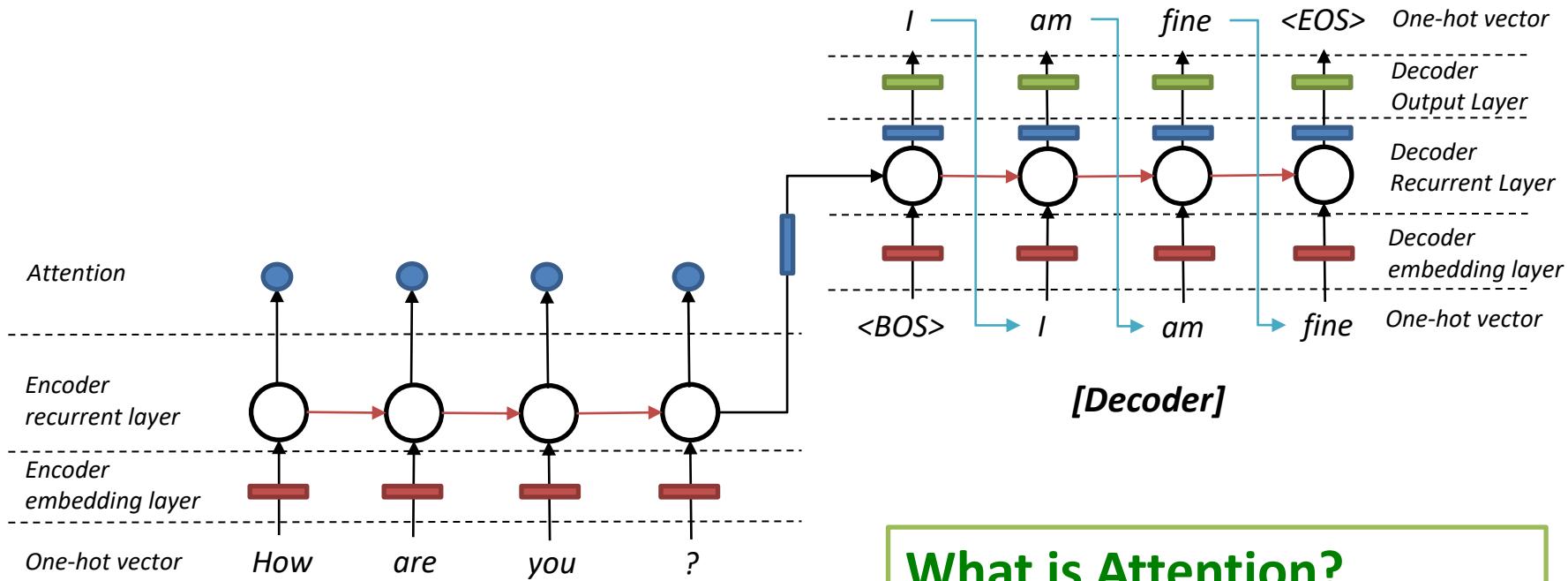


Attention

Seq2Seq Model: the bottleneck problem



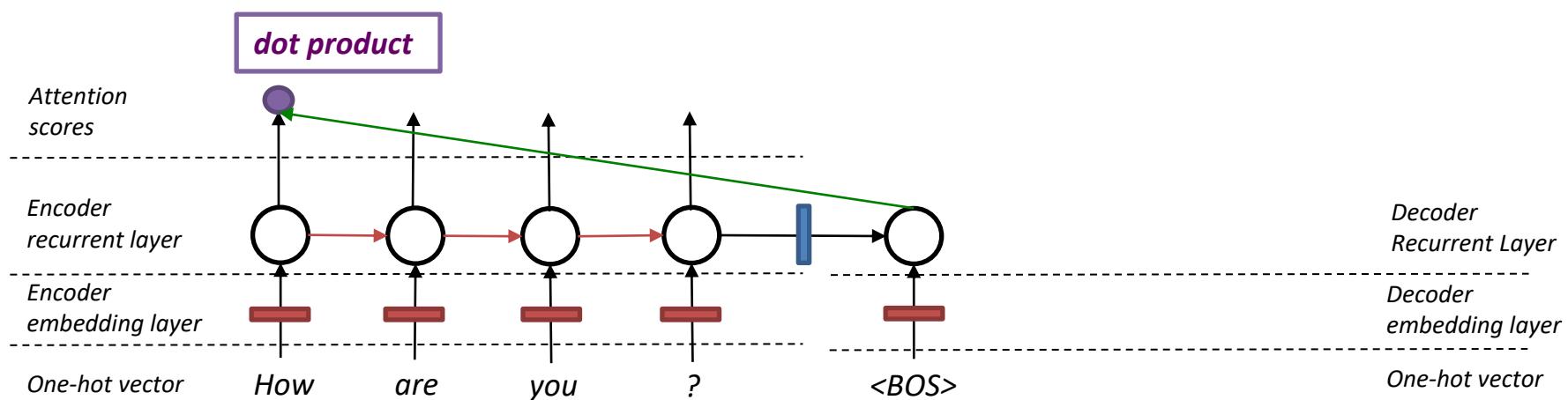
Seq2Seq with Attention



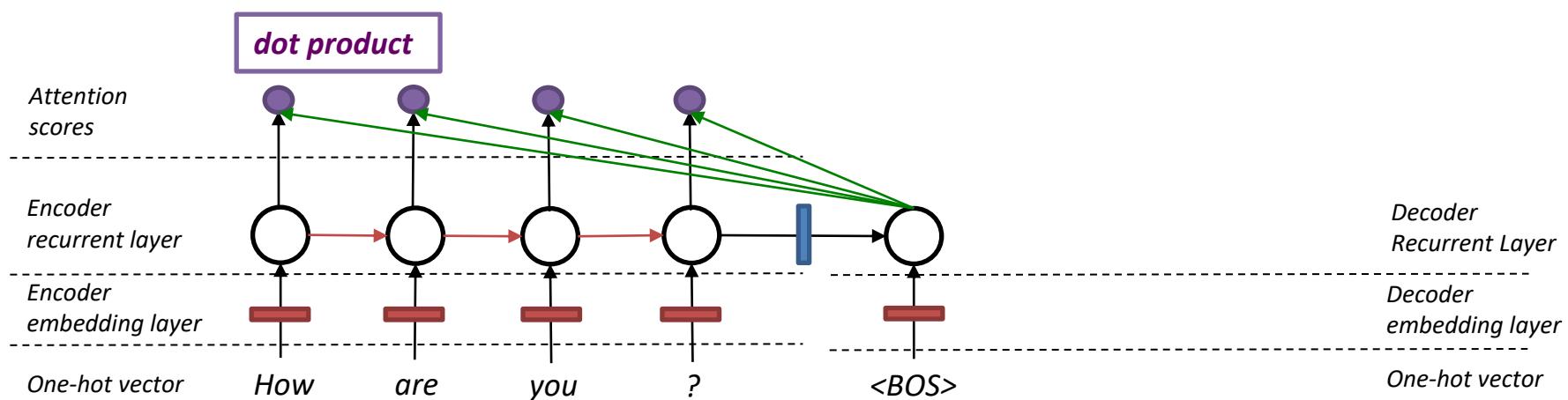
What is Attention?

On each step of the decoder, use direct connection to the encoder to focus on a particular part of the input sequence

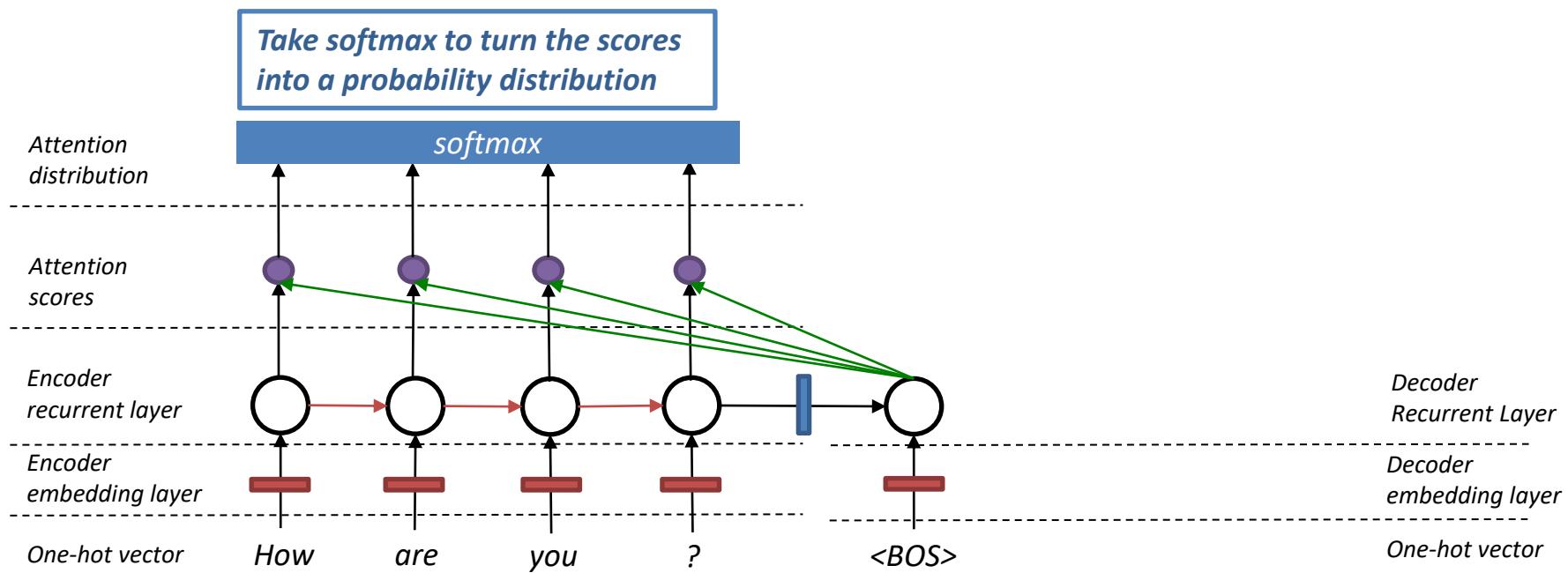
Seq2Seq with Attention



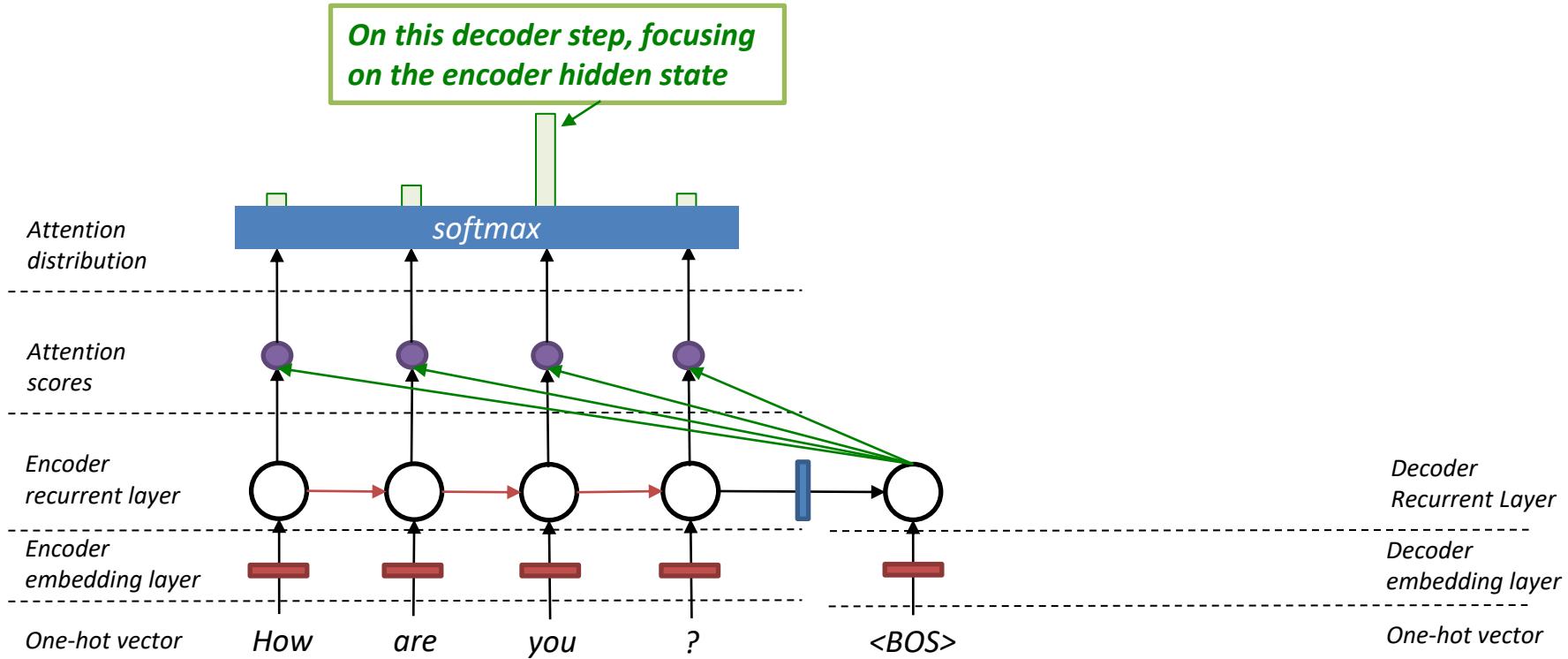
Seq2Seq with Attention



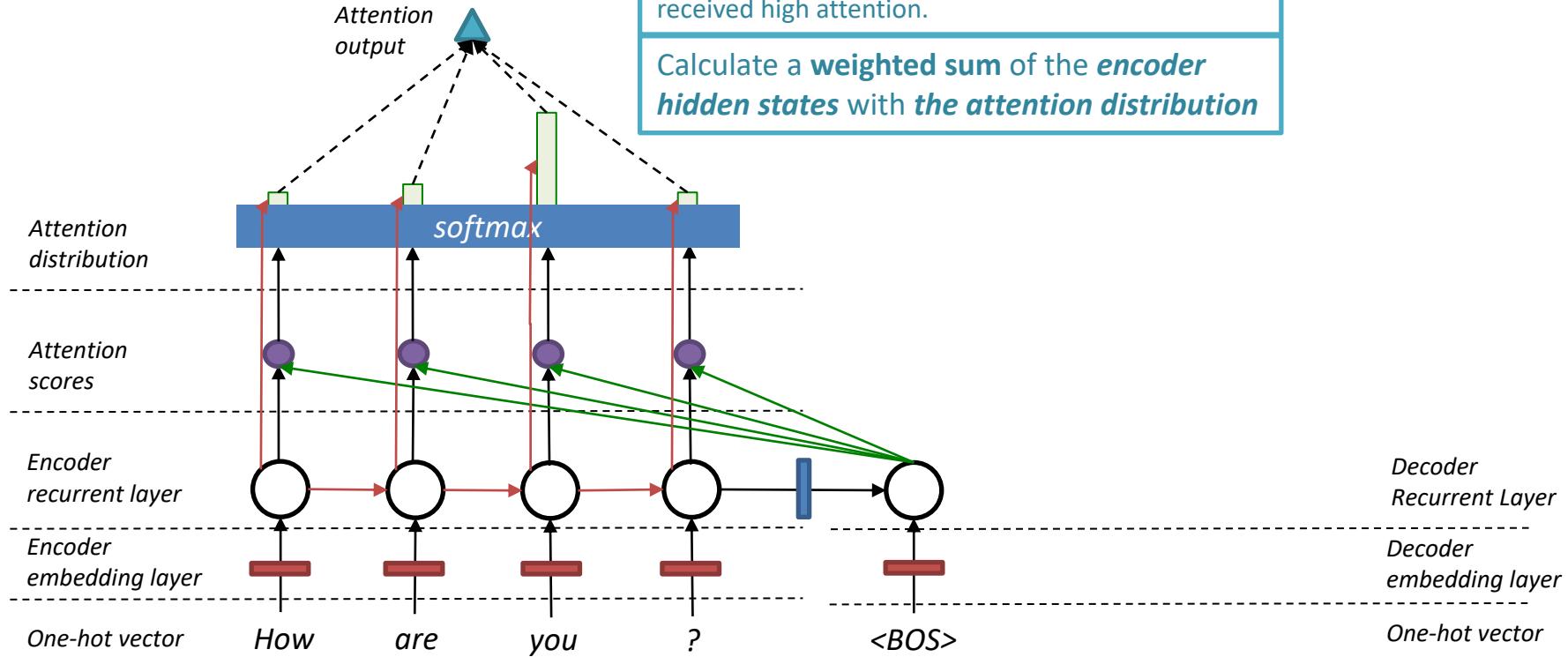
Seq2Seq with Attention



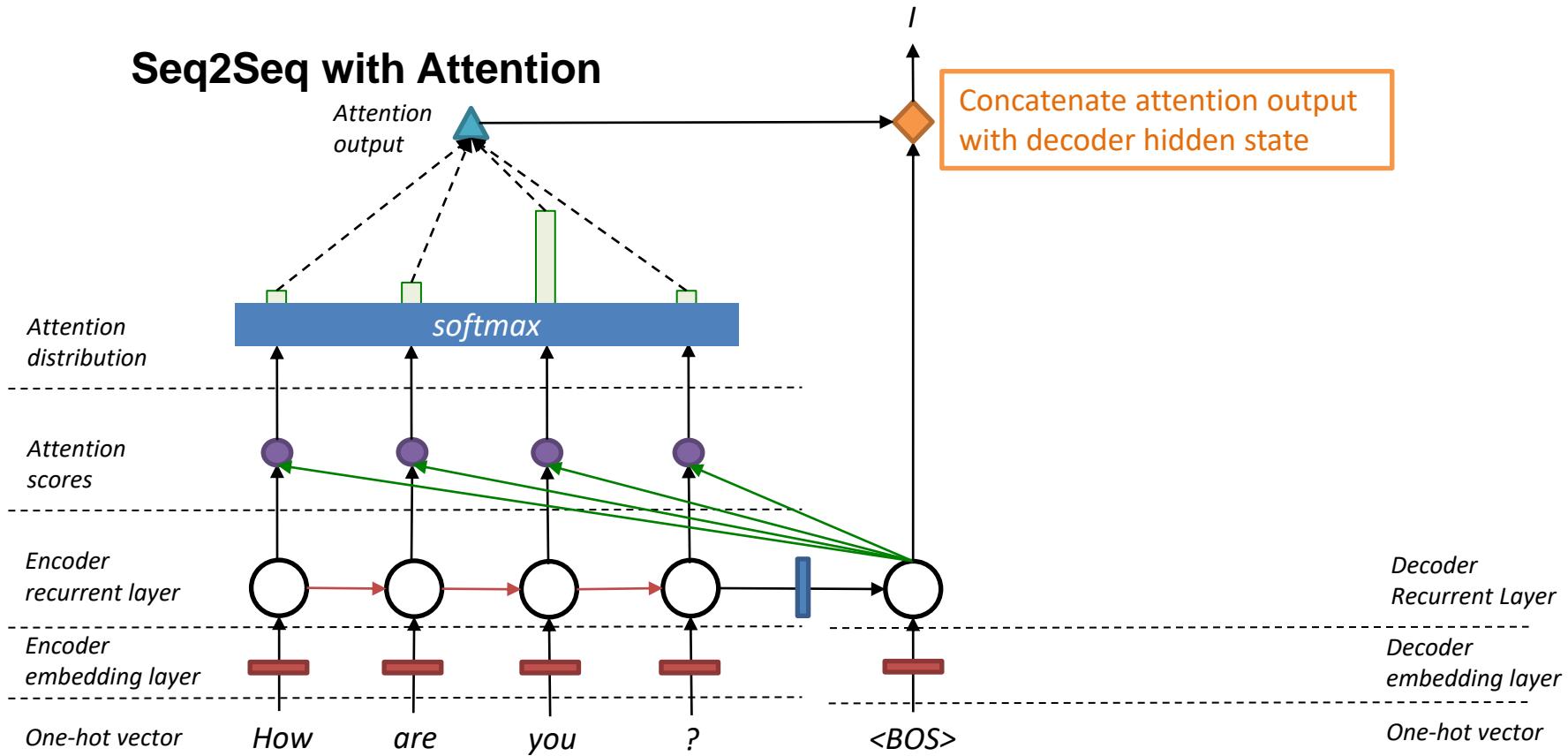
Seq2Seq with Attention

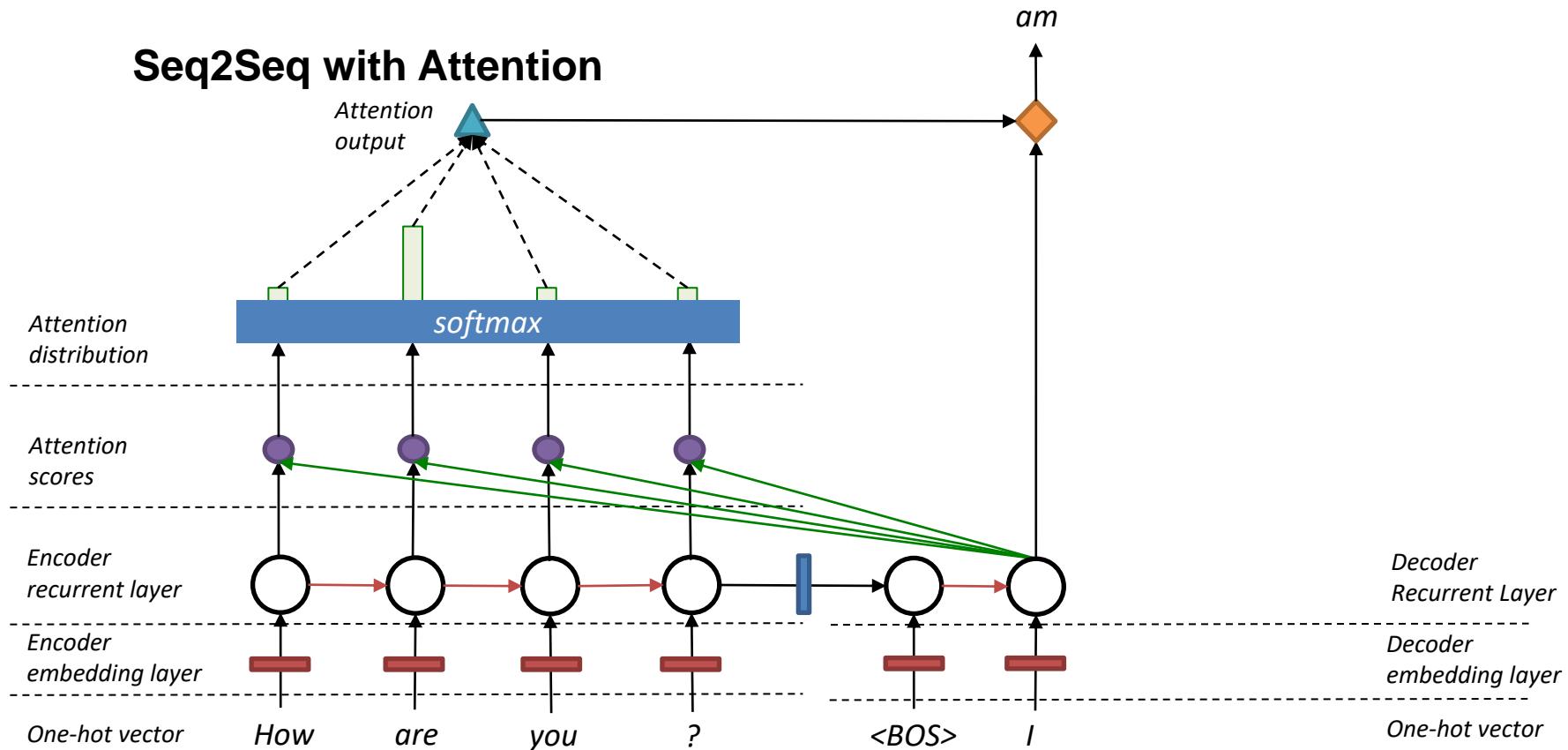


Seq2Seq with Attention

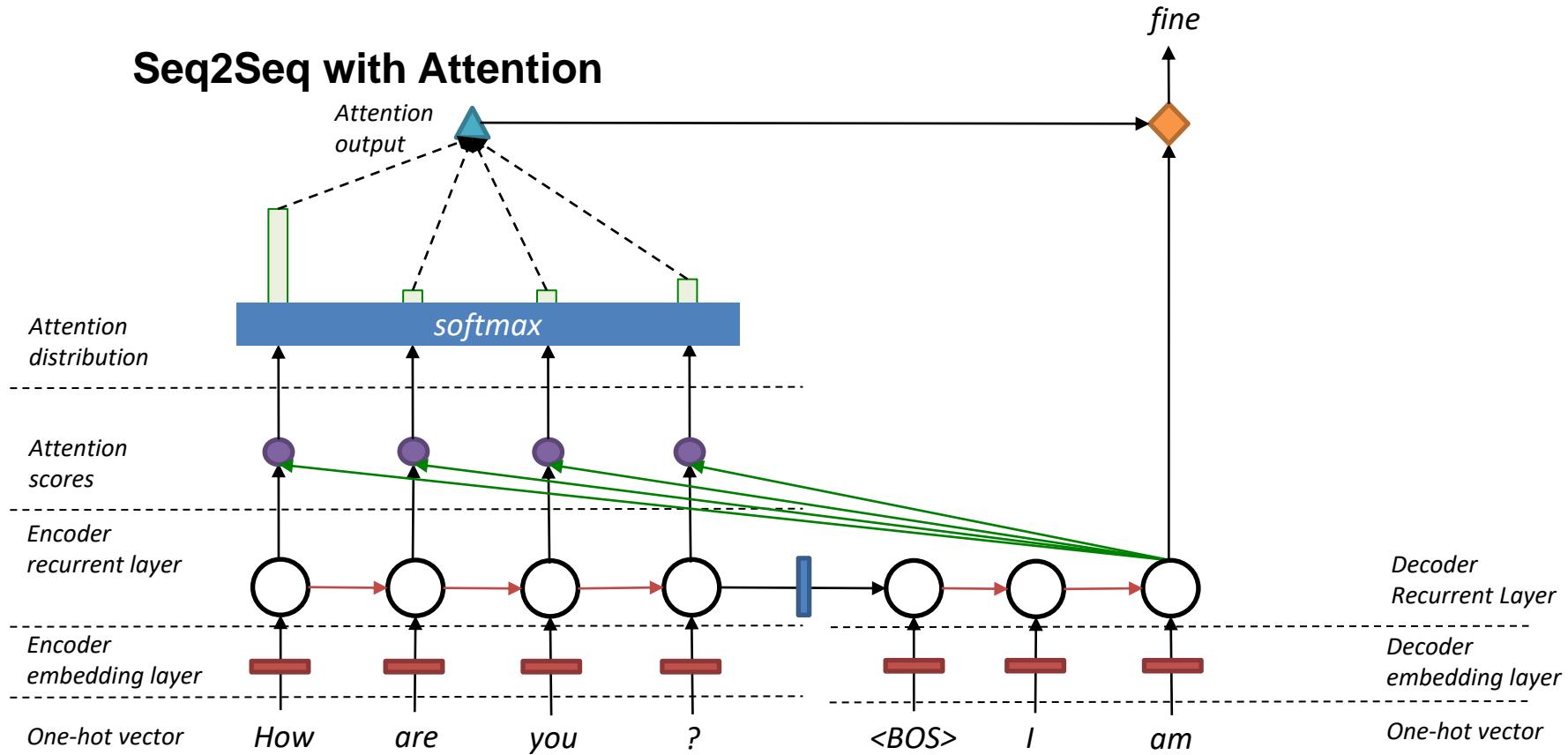


Seq2Seq with Attention

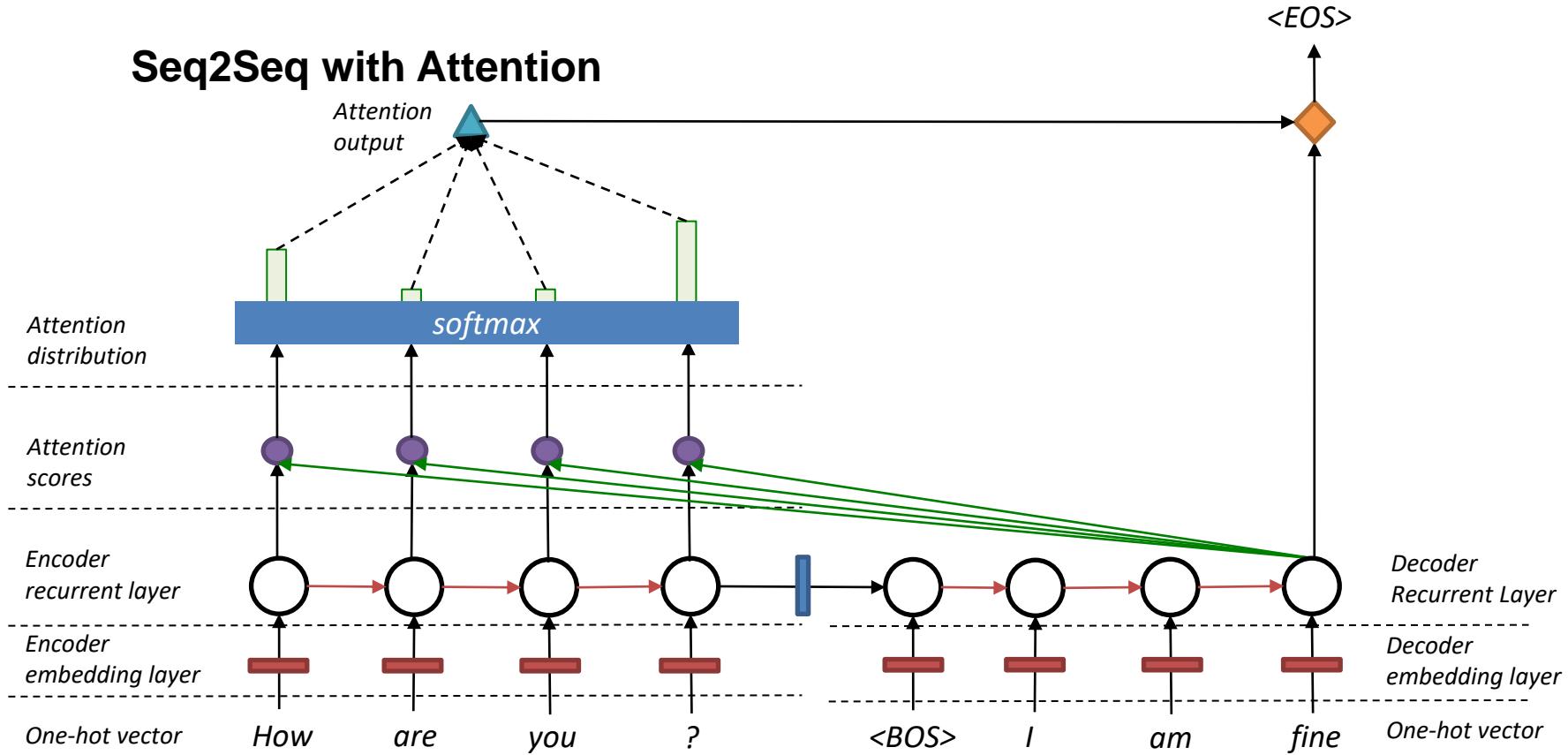




Seq2Seq with Attention

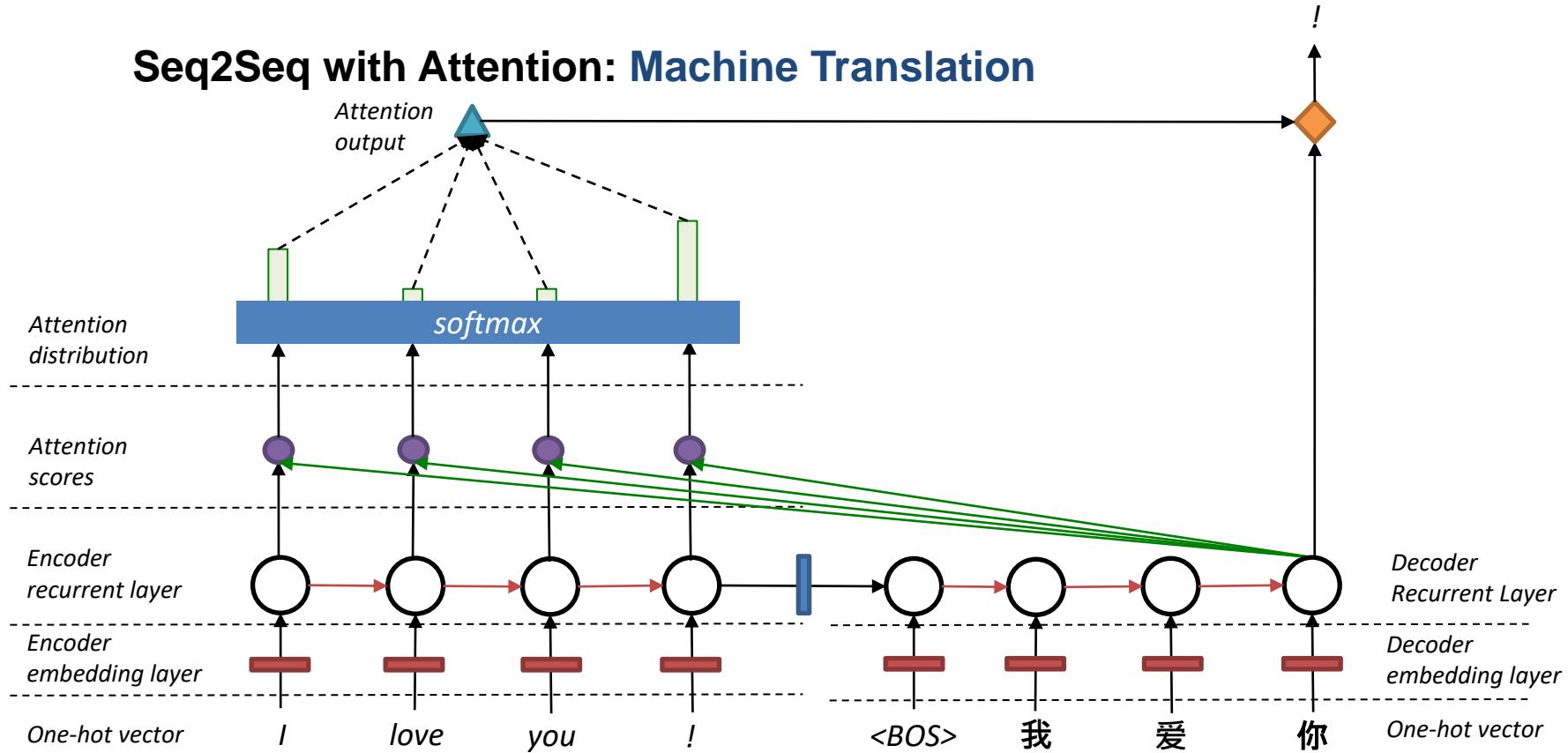


Seq2Seq with Attention



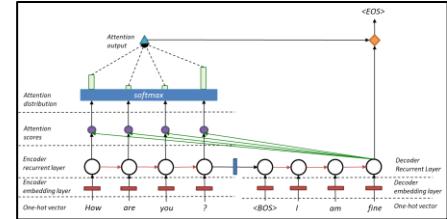
Attention

Seq2Seq with Attention: Machine Translation



Seq2Seq with Attention (Equations)

- *Encoder hidden states:* $h_1, \dots, h_N \in \mathbb{R}^h$
- *Decoder hidden state:* $s_t \in \mathbb{R}^h$ (on timestep t)



1. *Attention score e^t :* $e^t = [s_t^T h_1, \dots, s_t^T h_N] \in \mathbb{R}^N$ (for timestep t)
2. *Use softmax to get the attention distribution, α^t (for timestep t)*
(this is a probability distribution and sums to 1)

$$\alpha^t = \text{softmax}(e^t) \in \mathbb{R}^N$$

3. *Attention Output: Use α^t to take a weighted sum of the encoder hidden states*

$$a_t = \sum_{i=1}^N \alpha_i^t h_i \in \mathbb{R}^h$$

4. *Then, concatenate the attention output a_t with the decoder hidden state s_t and proceed as in the non-attention seq2seq model*

$$[a_t; s_t] \in \mathbb{R}^{2h}$$

Attention

Why we use Attention? The benefit!

Improve performance

- *Allow decoder to focus on certain parts of the source*

Solving the bottleneck problem

- *Allow decoder to directly look at the source (input)*

Reducing vanishing gradient problem

- *Provide shortcut to faraway states*

Providing some interpretability

- *Inspect attention distribution, and show what the decoder was focusing on*

Attention

Attention is now a general component in Deep Learning NLP

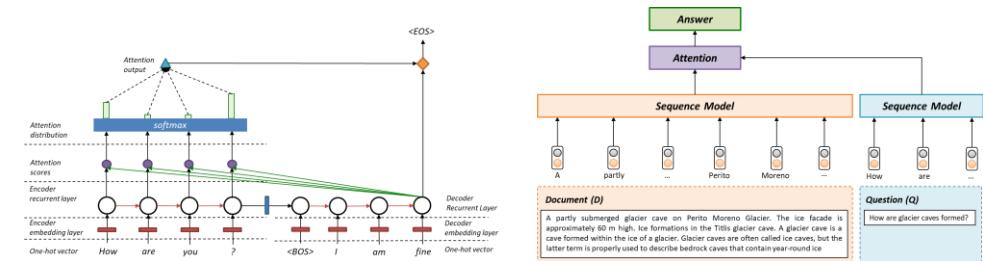
Attention is great way to improve the sequence to sequence model.

You can use attention in many architecture (not just seq2seq) and many NLP tasks (not just dialog system/NLG, Translation)

More general definition of attention:

Given a set of vector values, and a vector query, attention is a technique to compute a weighted sum of the values, dependent on the query.

For example, in the seq2seq + attention model, each decoder hidden state (query) attends to all the encoder hidden states (values).



Attention variants

There are several ways to compute attention score.

- Encoder hidden states: $h_1, \dots, h_N \in \mathbb{R}^h$
- Decoder hidden state: $s_t \in \mathbb{R}^h$ (on timestep t)

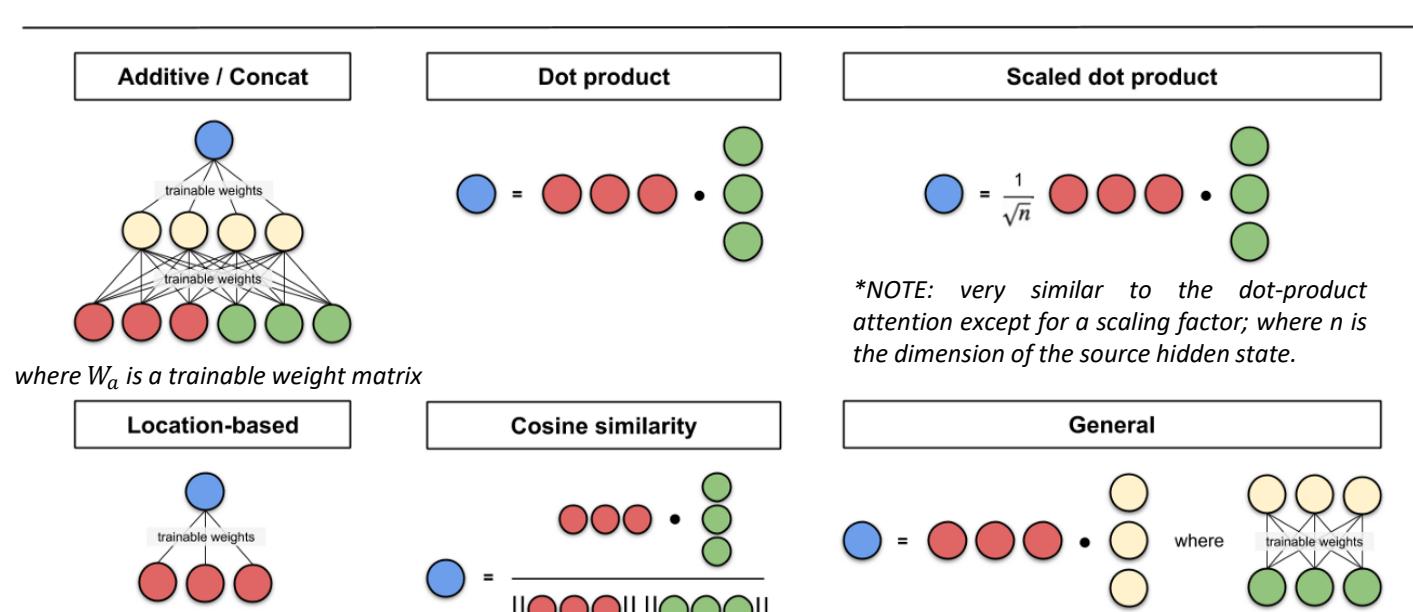
Attention Name	Attention score function	Reference
Content-base	$score(s_t, h_i) = cosine[s_t, h_i]$	Graves 2014
Dot-product	$score(s_t, h_i) = s_t^\top h_i$	Luong 2015
Scaled Dot-product	$score(s_t, h_i) = \frac{s_t^\top h_i}{\sqrt{n}}$ *NOTE: very similar to the dot-product attention except for a scaling factor; where n is the dimension of the source hidden state.	Vaswani 2017
Additive	$score(s_t, h_i) = v_a^\top \tanh(W_a[s_t; h_i])$	Vaswani 2017
General	$score(s_t, h_i) = s_t^\top W_a h_i$ *NOTE: where W_a is a trainable weight matrix in the attention layer.	Luong 2015
Location-based	$a_{t,i} = softmax(W_a s_t)$ *Note: This simplifies the softmax alignment to only depend on the target position.	Luong 2015

*The papers (Luong 2015 and Vaswani 2017) can be found in the canvas content page

Attention variants

There are several ways to compute attention score.

- Encoder hidden states: $h_1, \dots, h_N \in \mathbb{R}^h$
- Decoder hidden state: $s_t \in \mathbb{R}^h$ (on timestep t)



*Note: This simplifies the softmax alignment to only depend on the target position.

*NOTE: very similar to the dot-product attention except for a scaling factor; where n is the dimension of the source hidden state.

Categories of Attention Mechanism

A summary of broader categories of attention mechanisms

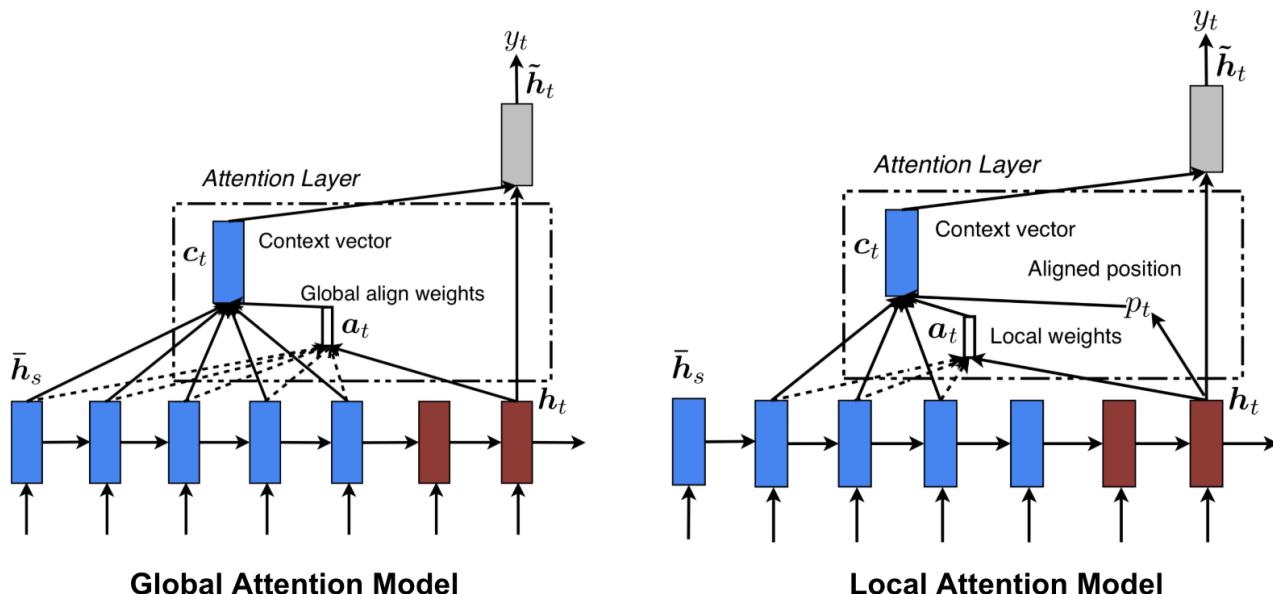
Name	Definition	Citation
Global or Local	<ul style="list-style-type: none">• Global: Attending to the entire input state space.• Local: Attending to the part of input state space (i.e. a patch of the input image.)	Luong 2015
Self-Attention	Relating different positions of the same input sequence. Theoretically the self-attention can adopt any attention score functions, but just replace the target sequence with the same input sequence.	Cheng 2016

*The papers (Luong 2015 and Cheng 2016) can be found in the canvas content page

Categories of Attention Mechanism (1)

Global/Local Attention

- *Global: Attending to the entire input state space.*
- *Local: Attending to the part of input state space*



Categories of Attention Mechanism (2)

Self-Attention

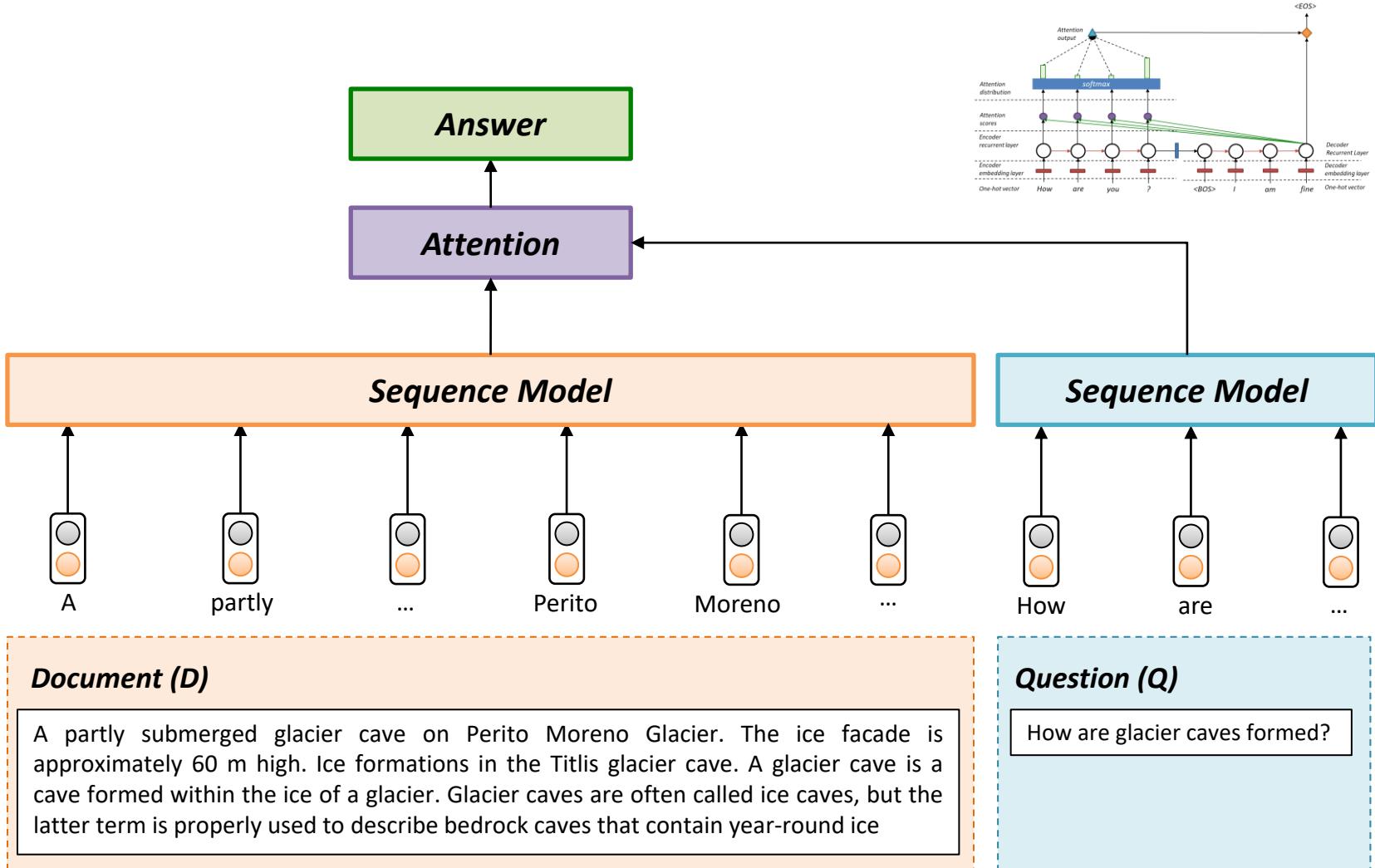
The long short-term memory network (Cheng et al., 2016) paper used self-attention to do machine reading. In the example below, the self-attention mechanism enables us to learn the correlation between the current words and the previous part of the sentence.

The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .

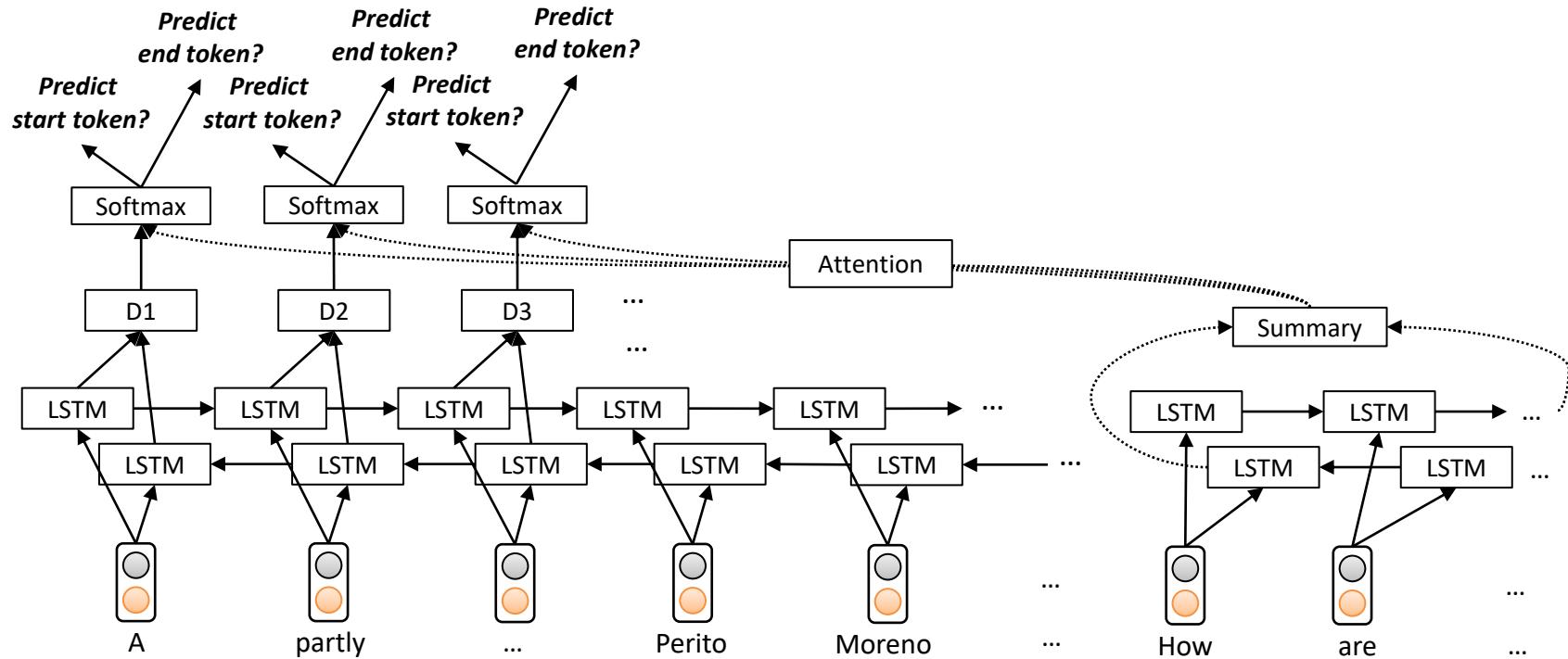
The current word is in red and the size of the blue shade indicates the activation level.

Reading Comprehension with Attention

A Generic Neural Model for Reading Comprehension



Bi-LSTM for Reading Comprehension with Attention



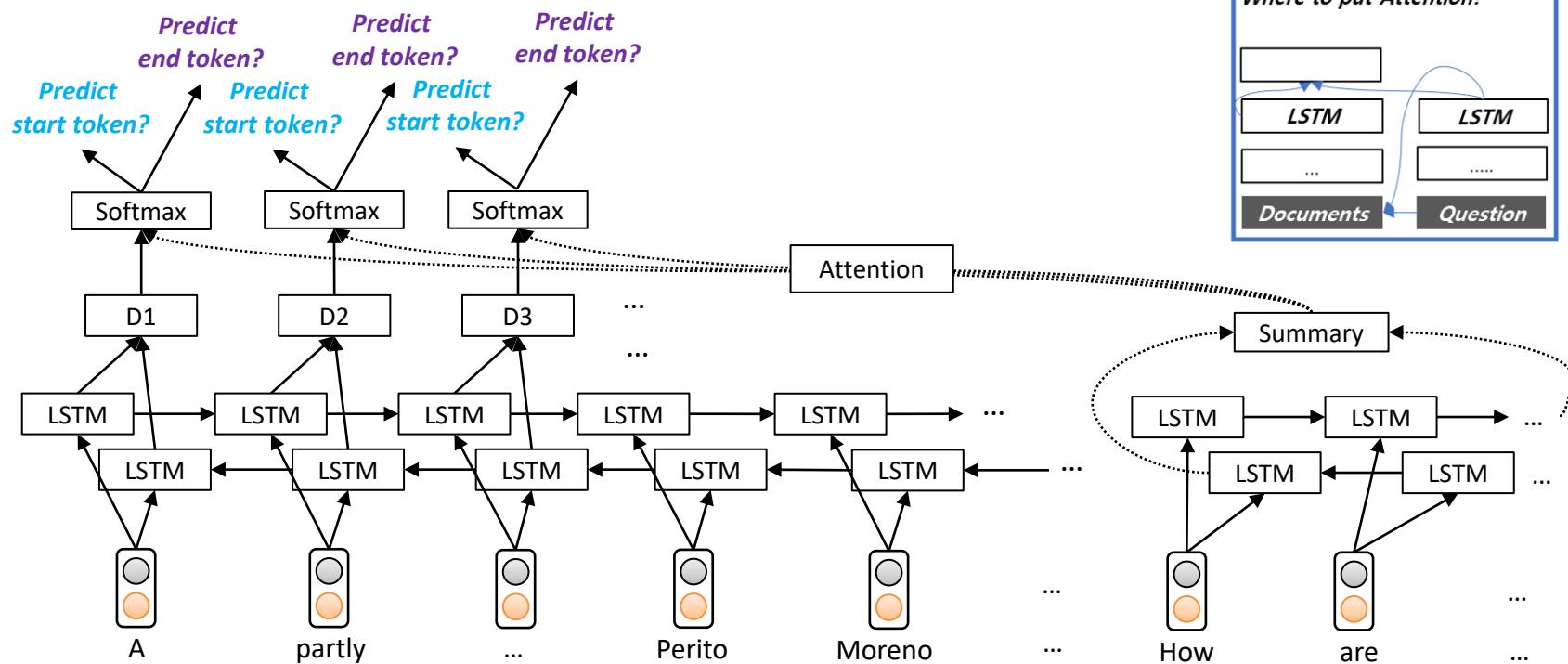
Document (D)

A partly submerged glacier cave on Perito Moreno Glacier. The ice facade is approximately 60 m high. Ice formations in the Titlis glacier cave. A glacier cave is a cave formed within the ice of a glacier. Glacier caves are often called ice caves, but the latter term is properly used to describe bedrock caves that contain year-round ice

Question (Q)

How are glacier caves formed?

Bi-LSTM for Reading Comprehension with Attention



Document (D)

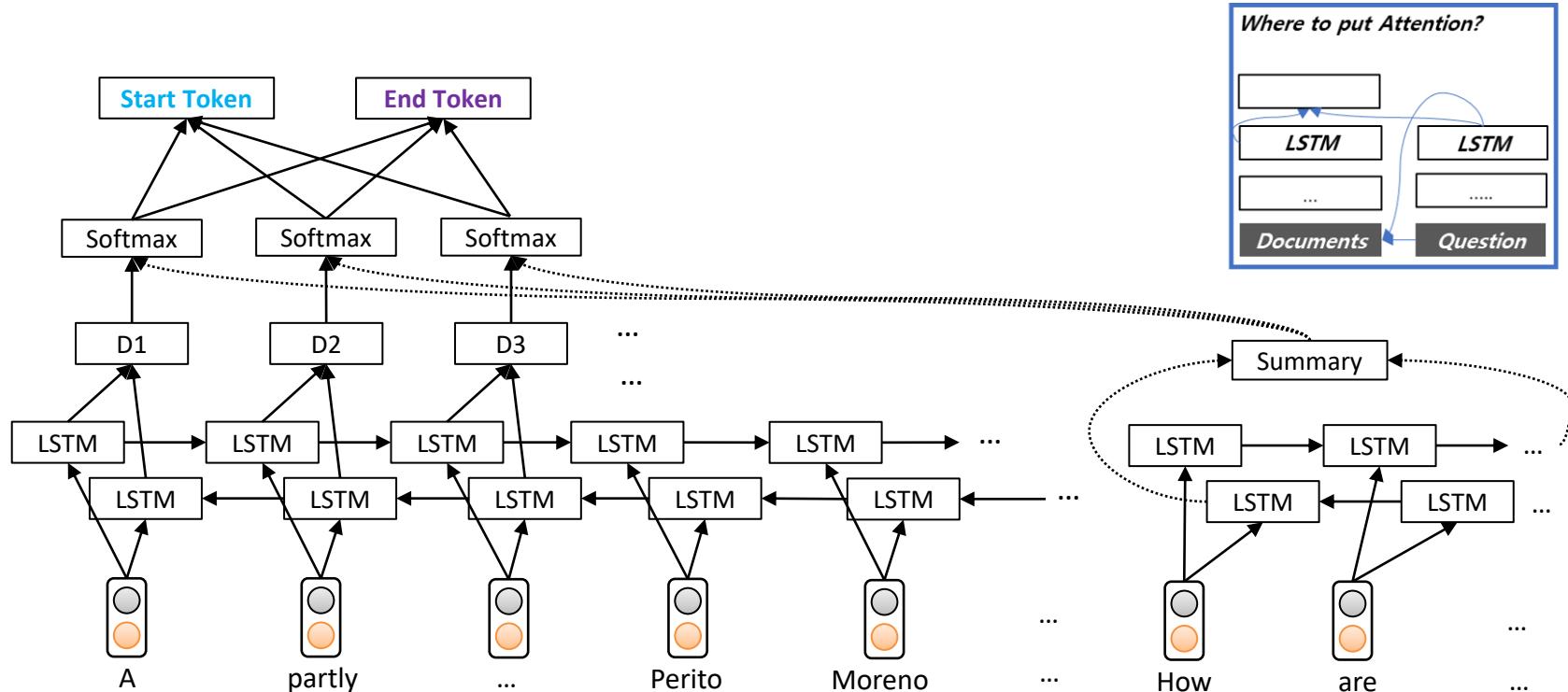
A partly submerged glacier cave on Perito Moreno Glacier. The ice facade is approximately 60 m high. Ice formations in the Titlis glacier cave. A glacier cave is a cave formed within the ice of a glacier. Glacier caves are often called ice caves, but the latter term is properly used to describe bedrock caves that contain year-round ice

Question (Q)

How are glacier caves formed?

Reading Comprehension with Attention

Bi-LSTM for Reading Comprehension with Attention



Document (D)

A partly submerged glacier cave on Perito Moreno Glacier. The ice facade is approximately 60 m high. Ice formations in the Titlis glacier cave. A glacier cave is a cave formed within the ice of a glacier. Glacier caves are often called ice caves, but the latter term is properly used to describe bedrock caves that contain year-round ice

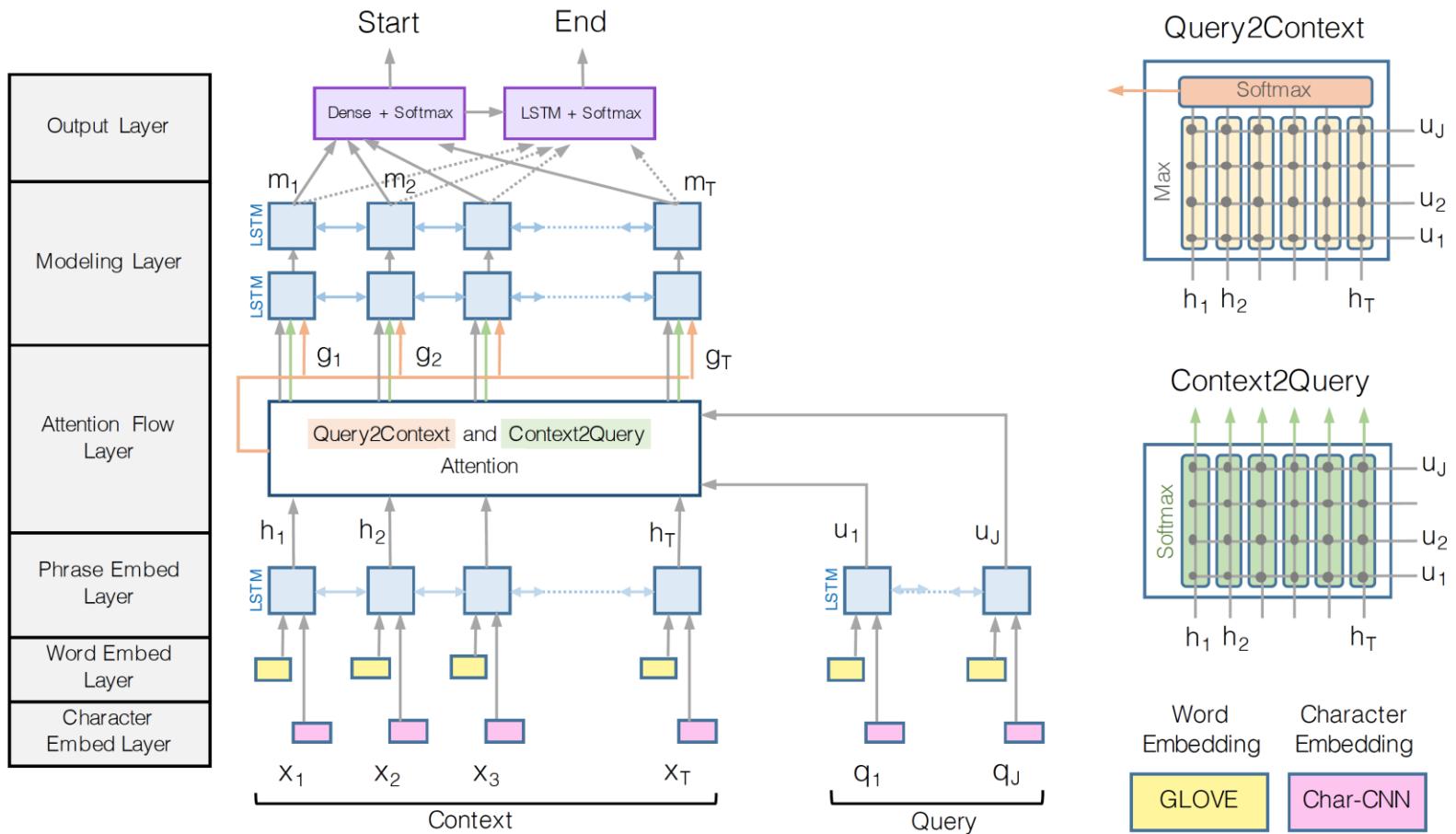
Question (Q)

How are glacier caves formed?

Reading Comprehension with Attention

Bi-Directional Attention Flow (Bi-DAF)

Bi-Directional Attention Flow for Machine Comprehension (Seo et al. 2017)



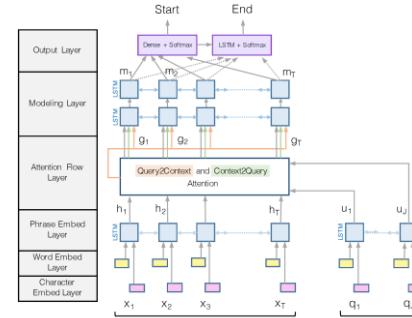
Bi-Directional Attention Flow (Bi-DAF)

Attention Flow layer is the core idea!

- *Variants and improvements to the Bi-DAF architecture over the years*

Attention should flow both ways:

- 1) *the context → the question (C2Q)*
- 2) *the question → the context (Q2C)*



*Both attentions are derived from a **shared similarity matrix** between the context (H) and the query (U), where S_{tj} indicates the similarity between t-th context word and j-th query word*

$$\mathbf{S}_{tj} = \alpha(\mathbf{H}_{:t}, \mathbf{U}_{:j}) \in \mathbb{R}$$

Bi-Directional Attention Flow (Bi-DAF)

Attention Flow layer is the core idea!

- *Variants and improvements to the Bi-DAF architecture over the years*

Attention should flow both ways:

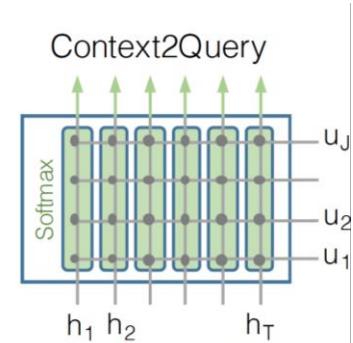
- 1) *the context → the question (C2Q)*
- 2) *the question → the context (Q2C)*

1. Context-to-Question (C2Q) attention:

- *which query words are most relevant to each context word*

$$\alpha^i = \text{softmax}(\mathbf{S}_{i,:}) \in \mathbb{R}^M \quad \forall i \in \{1, \dots, N\}$$

$$\mathbf{a}_i = \sum_{j=1}^M \alpha_j^i \mathbf{q}_j \in \mathbb{R}^{2h} \quad \forall i \in \{1, \dots, N\}$$



Bi-Directional Attention Flow (Bi-DAF)

Attention Flow layer is the core idea!

- *Variants and improvements to the Bi-DAF architecture over the years*

Attention should flow both ways:

- 1) *the context → the question (C2Q)*
- 2) *the question → the context (Q2C)*

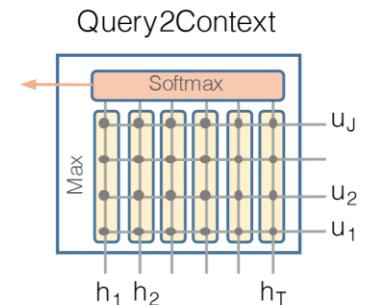
2. Question-to-Context (Q2C) attention:

- *the weighted sum of the most important words in the context with respect to the query – slight asymmetry through max*

$$\mathbf{m}_i = \max_j S_{ij} \in \mathbb{R} \quad \forall i \in \{1, \dots, N\}$$

$$\beta = \text{softmax}(\mathbf{m}) \in \mathbb{R}^N$$

$$\mathbf{c}' = \sum_{i=1}^N \beta_i \mathbf{c}_i \in \mathbb{R}^{2h}$$



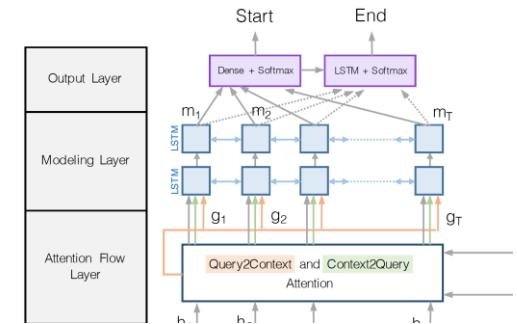
Bi-Directional Attention Flow (Bi-DAF)

A “*modelling*” layer:

- *Another deep (2-layer) Bi-LSTM over the passage*

And answer span selection is more complex:

- **Start:** Pass output of BiDAF and modelling layer concatenated to a dense FF layer and then a softmax
- **End:** Put output of modelling layer M through another BiLSTM to give M_2 and then concatenate with BiDAF layer and again put through dense FF layer and a softmax



/ More...?

More Advanced Architecture? Preview for following weeks

The transformer, based solely on attention mechanisms.

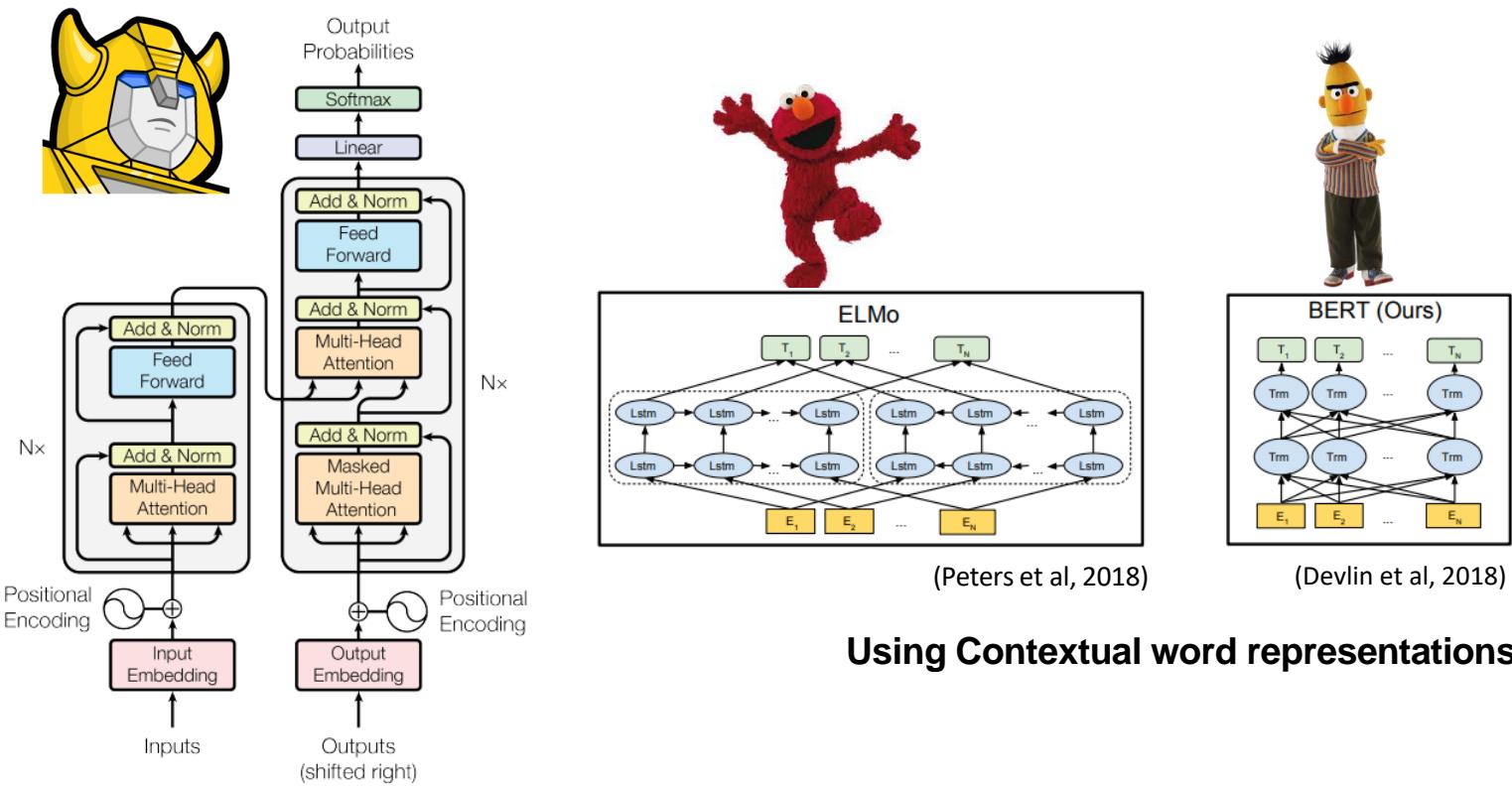


Figure 1: The Transformer - model architecture.

(Vaswani et al, 2017)

Using Contextual word representations

0 Question Answering

Research Areas in Question Answering

Research Area	Details
Knowledge-based QA (Semantic Parsing)	<ul style="list-style-type: none">• Answer is a logical form, possibly executed against a Knowledge Base• Context is a Knowledge Base
Information Retrieval-based QA <ul style="list-style-type: none">• Answer sentence selection• Reading Comprehension	<ul style="list-style-type: none">• Answer is a document, paragraph, sentence• Context is a corpus of documents or a specific document
Visual QA	<ul style="list-style-type: none">• Answer is simple and factual• Context is one/multiple image(s)
Library Reference	<ul style="list-style-type: none">• Answer is another question• Context is the structured knowledge available in the library and the librarians' view of it.

Visual Question Answering

Textual Question Answering: Recap

Answer questions by exploiting pure natural language.

Document / Passage

Caren watched TV last night. There was a guy playing tennis. Caren did not know who he is. He was wearing white shirts ...

Question

What was he doing?

Answer

Playing tennis

Visual Question Answering

Visual QA

Several questions require context outside of pure language.



Question

What was he doing?

Answer

Playing tennis

Visual Question Answering

Visual QA Datasets

Recently, there are a number of visual QA datasets have sprung up. Some of the more popular ones include:

Type #1: Real images

VQA v2.0 (Goyal et al. 2017)

Where is the child sitting?
fridge arms



How many children are in the bed?



Type #2: Semantic reasoning

CLEVER (Johnson et al. 2016)



Q: Are there an **equal number** of **large things** and **metal spheres**?

Q: **What size** is the **cylinder** that is **left of** the **brown metal thing** that is **left of** the **big sphere**?

Q: There is a **sphere** with the **same size** as the **metal cube**; is it **made of the same material** as the **small red sphere**?

Type #3: Combined

GQA (Hudson and Manning, 2019)



Is the **bowl** to the right of the **green apple**?

What type of **fruit** in the image is **round**?

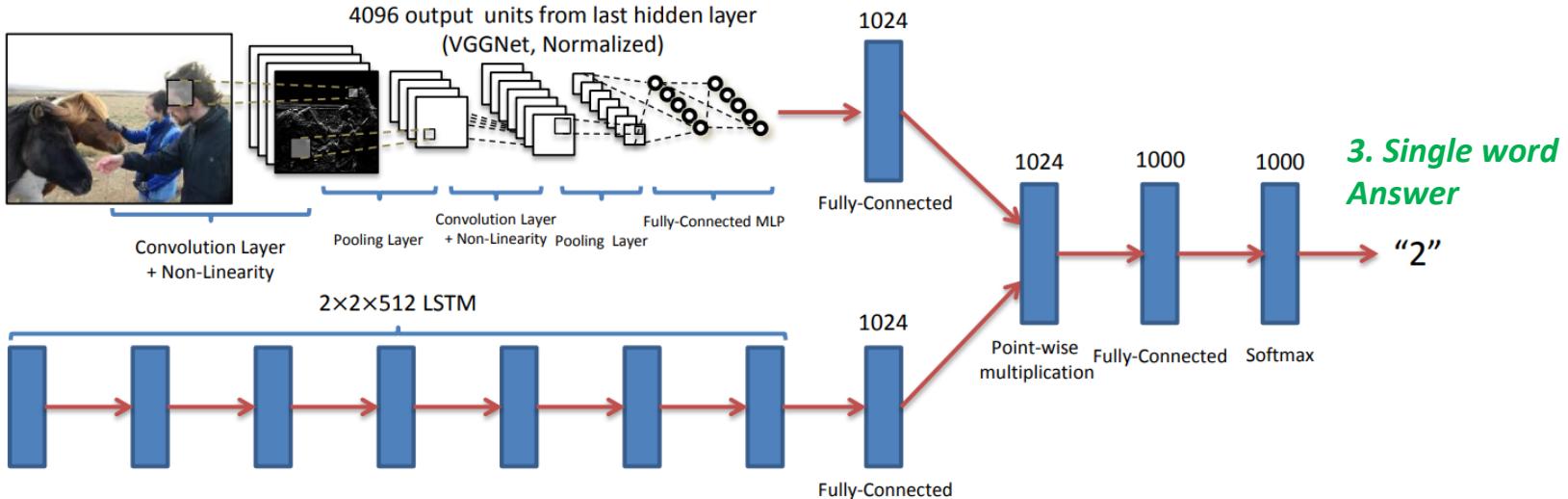
What color is the **fruit** on the right side, **red** or **green**?

Is there any **milk** in the **bowl** to the left of the **apple**?

Visual Question Answering

How does it work?

2. Context is a single picture using convolutional neural network (CNN)



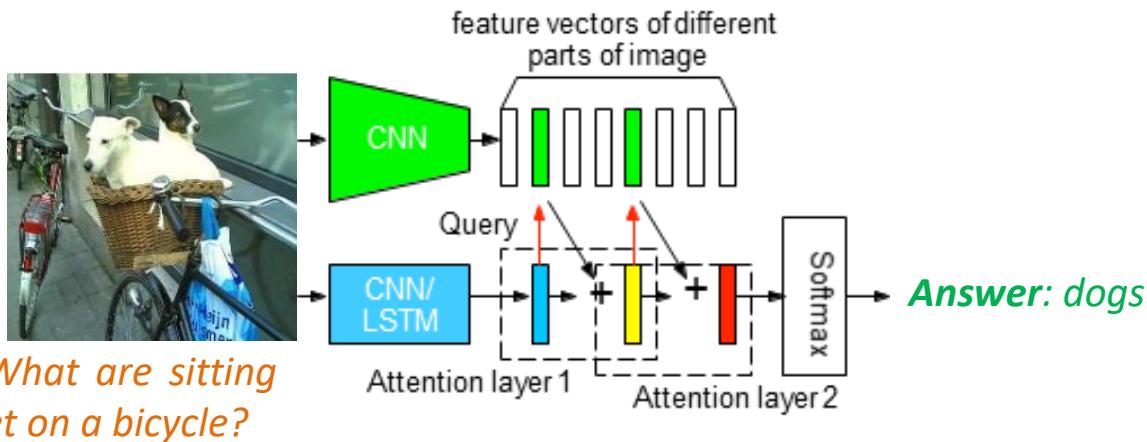
"How many horses are in this image?"

1. Encode sentence with sequence models

Visual Question Answering

How does it work?

*The idea of Visual QA is exactly same as reading comprehension-oriented.
Why can't we use **Attention** then? (Yang et al. 2015)*



Visual Question Answering

Visual QA with Attention

Let's try some example. VisualQA (<http://vqa.cloudcv.org/>)

(a) What are pulling a man on a wagon down on dirt road?
 Answer: horses Prediction: horses



(b) What is the color of the box ?
 Answer: red Prediction: red



(c) What next to the large umbrella attached to a table?
 Answer: trees Prediction: tree



(d) How many people are going up the mountain with walking sticks?
 Answer: four Prediction: four



(e) What is sitting on the handle bar of a bicycle?
 Answer: bird Prediction: bird

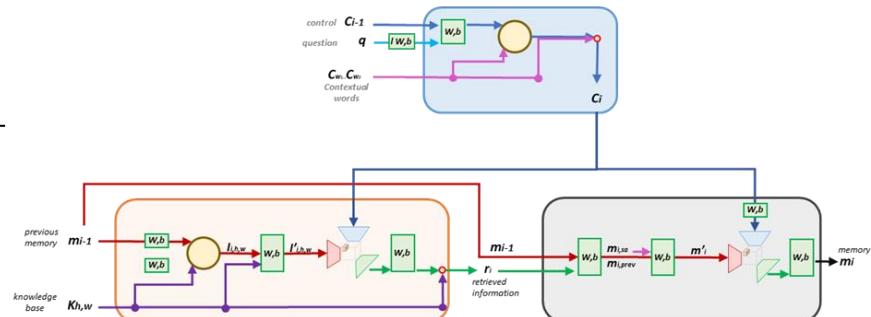
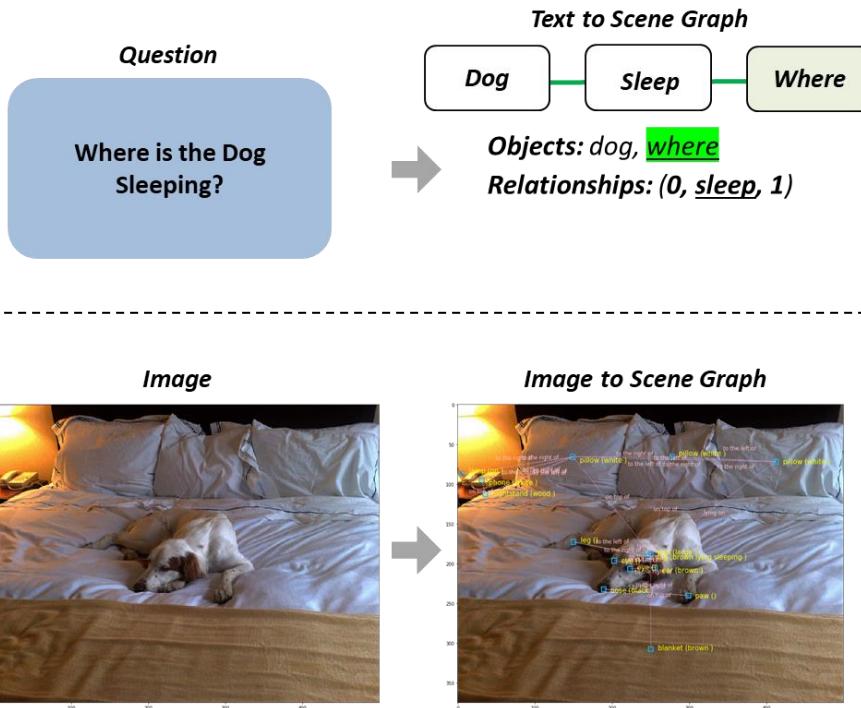


(f) What is the color of the horns?
 Answer: red Prediction: red



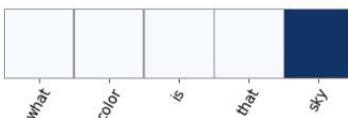
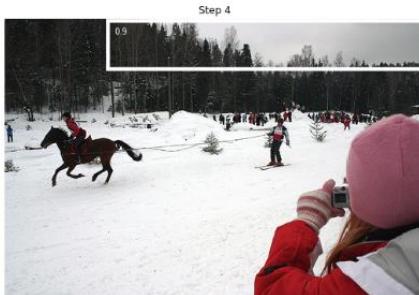
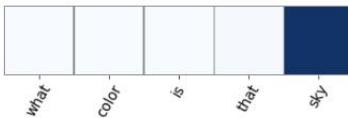
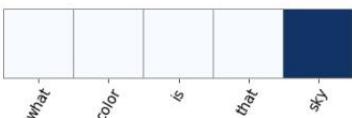
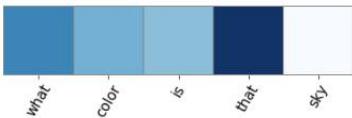
Visual Question Answering

Visual QA with Attention (Usyd NLP Group 2020)



Visual QA with Attention (Usyd NLP Group 2020)

Predictions #: 44
 Image ID: 2356967
 Object list index #: 62642
 Question: What color is that sky?
 Prediction: gray
 Answer: gray

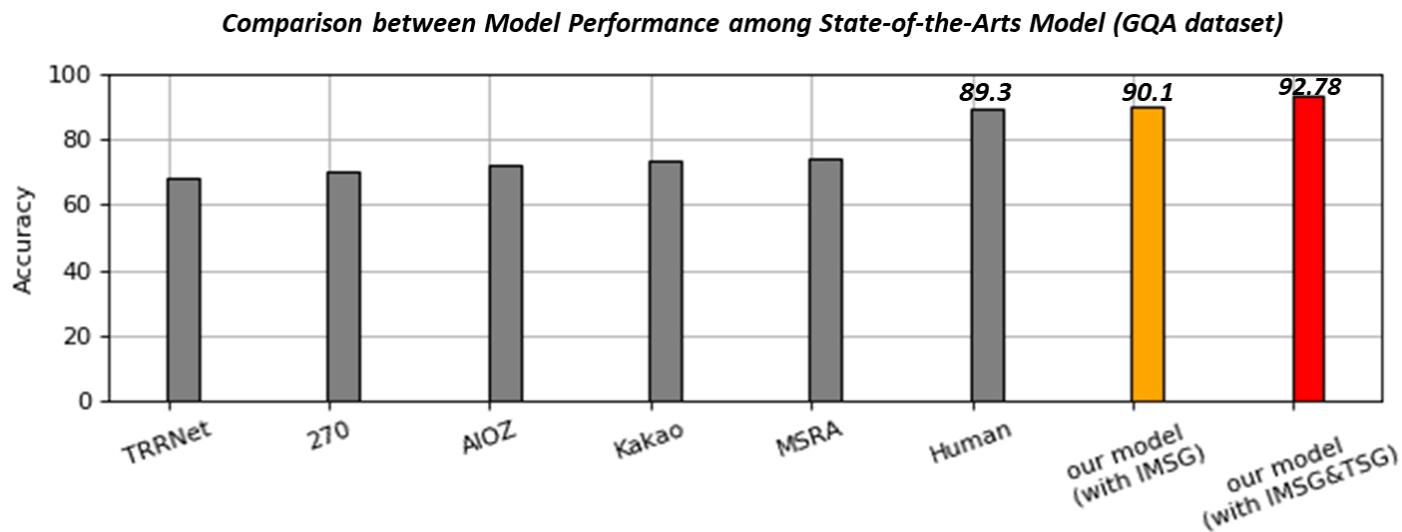


Let's have a look at the demo!!

Visual Question Answering

Visual QA with Attention (Usyd NLP Group 2020)

Testing Result



/ Additional QA

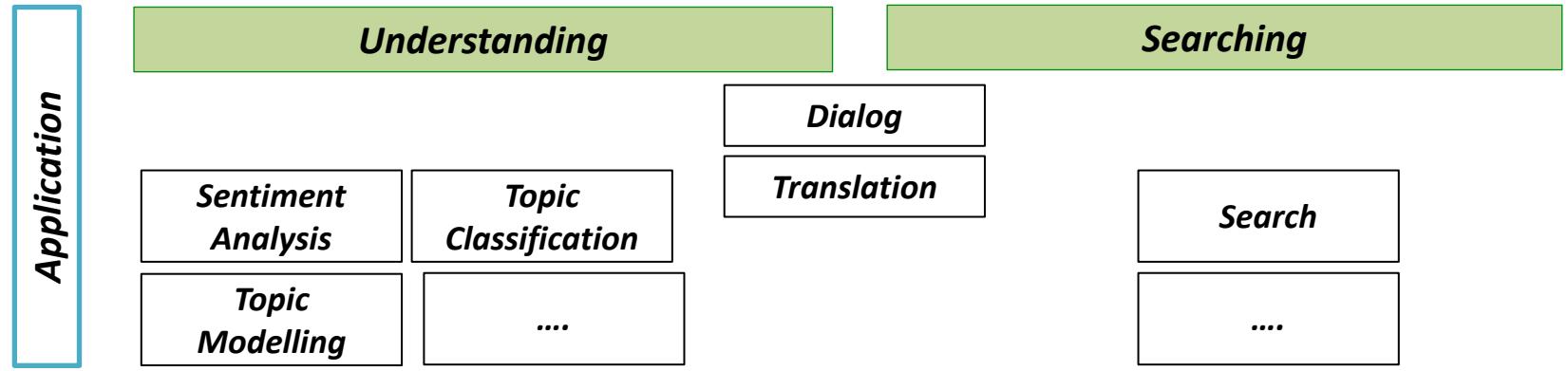
There is no reason to limit to just IR-based or Knowledge-based

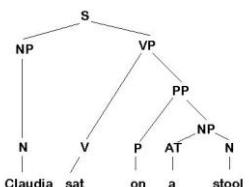
Using multiple information sources? IBM Watson!



/ The big picture of NLP

The purpose of Natural Language Processing: Overview



NLP Stack	Entity Extraction	When Sebastian Thrun ...	When Sebastian Thrun PERSON started at Google ORG in 2007 DATE
	Parsing	Claudia sat on a stool	 <pre> graph TD S --- NP S --- VP NP --- N1[Claudia] VP --- V1[sat] VP --- PP PP --- P1[on] PP --- NP2[NP] NP2 --- AT2[a] NP2 --- N2[stool] </pre>
	PoS Tagging	She sells seashells	[she/PRP] [sells/VBZ] [seashells/NNS]
	Stemming	Drinking, Drank, Drunk	Drink
	Tokenisation	How is the weather today	[How] [is] [the] [weather] [today]

/ Reference

Reference for this lecture

- Deng, L., & Liu, Y. (Eds.). (2018). Deep Learning in Natural Language Processing. Springer.
- Rao, D., & McMahan, B. (2019). Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. " O'Reilly Media, Inc.".
- Manning, C. D., Manning, C. D., & Schütze, H. (1999). Foundations of statistical natural language processing. MIT press.
- Manning, C 2018, Natural Language Processing with Deep Learning, lecture notes, Stanford University

- Berant, J., Chou, A., Frostig, R., & Liang, P. (2013). Semantic parsing on freebase from question-answer pairs. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (pp. 1533-1544).
- Chen, D., Bolton, J., & Manning, C. D. (2016). A thorough examination of the cnn/daily mail reading comprehension task. arXiv preprint arXiv:1606.02858.
- Chen, D., Fisch, A., Weston, J., & Bordes, A. (2017). Reading wikipedia to answer open-domain questions. arXiv preprint arXiv:1704.00051.
- Seo, M., Kembhavi, A., Farhadi, A., & Hajishirzi, H. (2016). Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603.
- Yang, Z., He, X., Gao, J., Deng, L., & Smola, A. (2016). Stacked attention networks for image question answering. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 21-29).
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).
- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

COMP5046

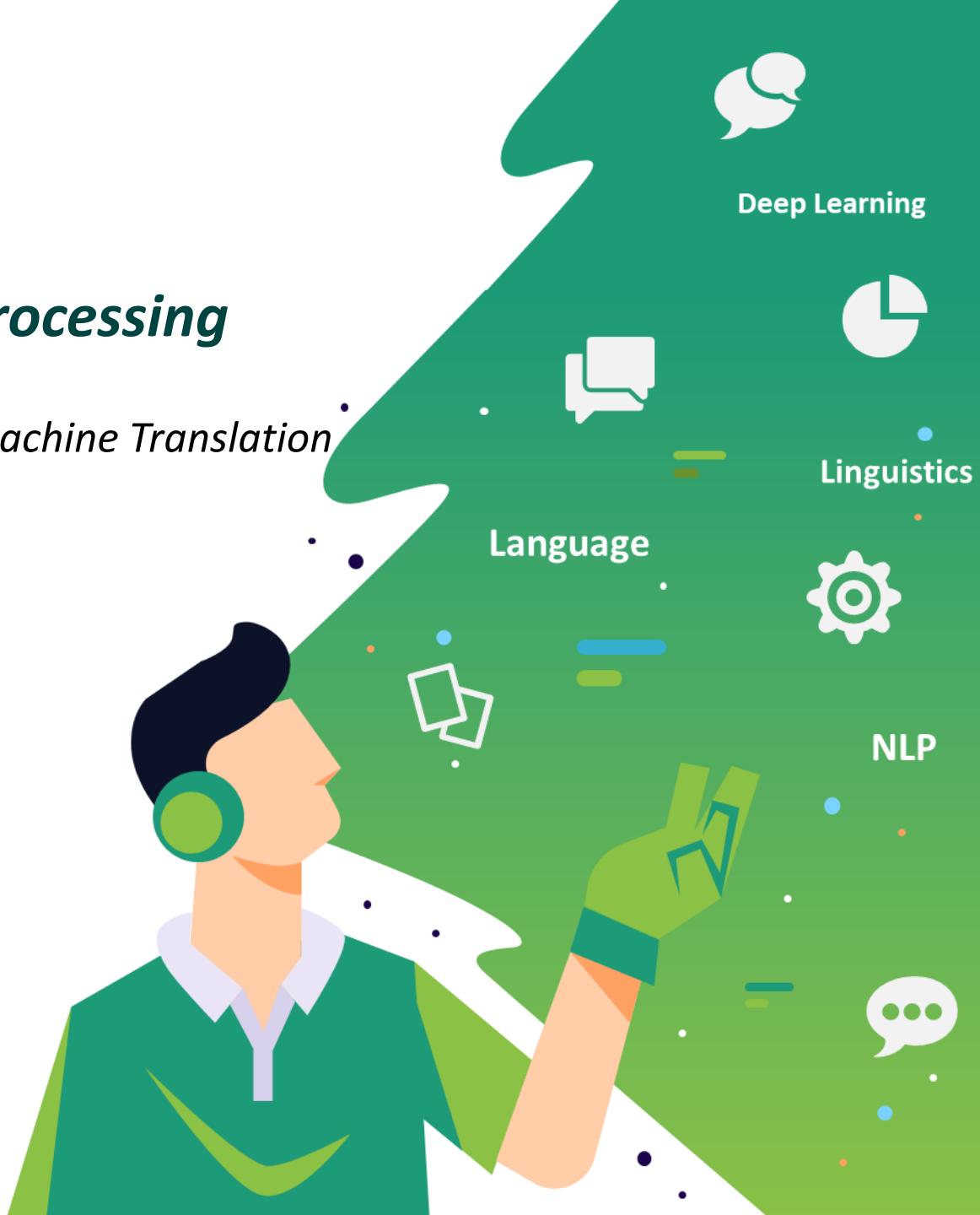
Natural Language Processing

*Lecture 11: Advanced NLP: Machine Translation
and Transformer*

Dr. Caren Han

Semester 1, 2022

*School of Computer Science,
University of Sydney*



0 LECTURE PLAN

Lecture 11: Machine Translation and Transformer

1. Machine Translation
2. Statistical Machine Translation
3. Neural Machine Translation
4. Attention and Transformer for MT
5. The Rise of the Pre-trained Model

0

Assignment 2 Specification

1

What is Machine Translation?

Machine Translation

Machine Translation

“translate a sentence x from one language (**the source language**) to a sentence y in another language (**the target language**).”

Source language

Sentence x

生命短暂

Target language

Sentence y

life is short



Machine

2

Statistical Machine Translation

Statistical Machine Translation

Statistical Machine Translation

“Learning a **probabilistic model** from data”

Source language (x)

Sentence x

生命短暂

Target language (y)

Sentence y

life is short

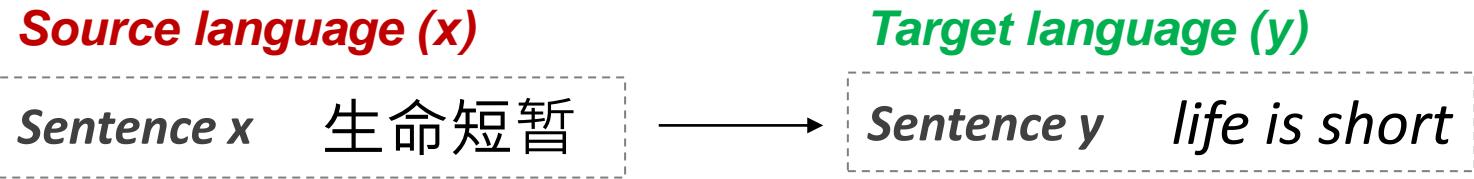
Best translation?

$$\operatorname{argmax}_y P(y|x)$$

How to learn translation model $P(x|y)$?

Statistical Machine Translation

“Learning a **probabilistic model** from data”



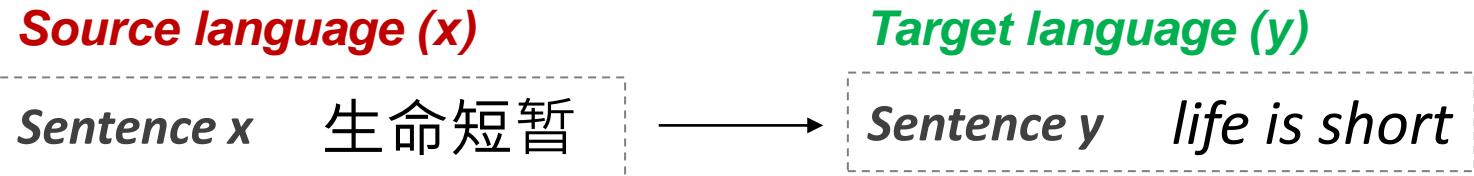
$$\begin{aligned}
 & \text{Best translation?} \\
 & \operatorname{argmax}_y P(y|x) \\
 & \downarrow \\
 & \text{Bayes Rule} \quad \operatorname{argmax}_y \underline{P(x|y)} \underline{P(y)}
 \end{aligned}$$

Translation Model (fidelity)
 Models how words and phrases should be translated

Language Model (fluency)
 Models to write good English

Statistical Machine Translation

“Learning a **probabilistic model** from data”



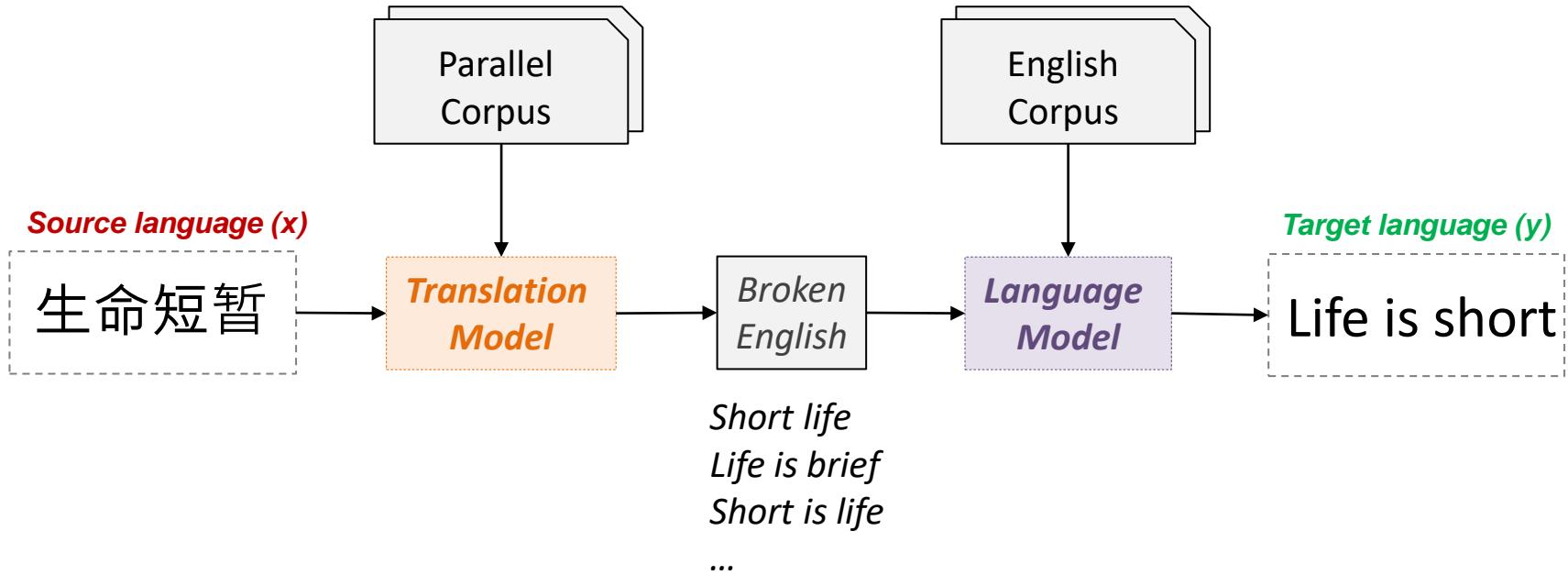
$$\begin{aligned}
 & \text{Best translation?} \\
 & \operatorname{argmax}_y P(y|x) \\
 & \downarrow \\
 & \text{Bayes Rule} \quad \operatorname{argmax}_y \underline{P(x|y)} \underline{P(y)}
 \end{aligned}$$

Translation Model (fidelity)
Learned from **parallel data**

Language Model (fluency)
Learned from **monolingual data**

Statistical Machine Translation

How to learn translation model with *parallel corpus*?



Bayes Rule

$$\operatorname{argmax}_y P(x|y) P(y)$$

Translation Model (fidelity)
 Learnt from **parallel data**

Language Model (fluency)
 Learnt from **monolingual data**

Parallel corpus and Alignment

How to learn translation model from the parallel corpus?

*i.e. pairs of human-translated
Chinese/English sentences*



OPUS is a growing collection of translated texts from the web. In the OPUS project we try to convert and align free online data, to add linguistic annotation, and to provide the community with a publicly available parallel corpus. OPUS is based on open source products and the corpus is also delivered as an open content package. We used several tools to compile the current collection. All pre-processing is done automatically. No manual corrections have been carried out.

The OPUS collection is growing! Check this page from time to time to see new data arriving ...

Contributions are very welcome! Please contact <jorg.tiedemann@helsinki.fi>

Search & download resources: en (English) ▾ zh (Chinese) ▾ >1M ▾

Language resources: click on [tmx | moses | xces | lang-id] to download the data! (raw = untokenized, ud = parsed with universal dependencies, alg = word alignments and phrase tables)

corpus	doc's	sent's	en tokens	zh tokens	XCES/XML	raw	TMX	Moses	mono	raw	ud	alg	dic	freq	other files
MultiUN v1	67167	10.5M	288.2M	80.0M	xces en zh	en zh	tmx	moses	en zh	en zh		alg		en zh	query sample
OpenSubtitles v2016	9829	10.3M		80.6M	71.7M	xces en zh	en zh	tmx	moses	en zh	en zh	alg	dic	en zh	sample
OpenSubtitles v2011	714	0.7M		6.1M	6.2M	xces en zh	en zh								sample
News-Commentary v11	7107	0.1M	6.6M	1.6M	xces en zh	en zh	tmx	moses	en zh	en zh	alg smt	dic	en zh	query	sample
Tanzil v1	30	0.2M	5.6M	1.7M	xces en zh	en zh	tmx	moses	en zh	en zh	alg smt	dic	en zh	query	sample
UN v20090831	1	74.1k	3.7M	1.2M	xces en zh	en zh	tmx	moses	en zh	en zh	alg smt		en zh	query	sample
News-Commentary v9.1	1	91.6k	3.4M	0.8M	xces en zh	en zh	tmx	moses	en zh	en zh	alg smt		en zh		sample
News-Commentary v9.0	1	91.6k	3.1M	0.8M	xces en zh	en zh	tmx	moses	en zh	en zh			en zh		sample
TED2013 v1.1	1	0.2M	3.1M	0.9M	xces en zh	en zh	tmx	moses	en zh	en zh	alg smt	dic	en zh	query	sample
<i>total</i>	84851	22.2M	400.4M	164.9M	22.2M		21.5M	21.5M							

Parallel corpus and Alignment

How to align these sentence (Open subtitles)

(trg)="1"> 片名 : 解放 的 潘 多 拉

(src)="1"> My name is Alice .

(trg)="2"> 我的 名字 是 阿 ? 丽 斯 。

(src)="2"> Alice Bonnard ...

(trg)="3"> 阿 ? 丽 斯 ...

(src)="3"> like my father and mother .

(trg)="4"> 象我 的 父母 。

(src)="4"> I hate people .

(trg)="5"> 我 恨 周 ? 围 的 人 。

(src)="5"> They oppress me .

(trg)="6"> 他 ?? 压 迫 我 。

(src)="6"> All year , I was away at school .

(trg)="7"> 整年 我 都 是 去 ? 学 校 。

(src)="7"> I only came home for end- of- term holidays

(trg)="8"> 我 只 有 ? 学 期 近 ? 结 束 ? 时 回 家

(src)="8"> Summer holidays were the worst .

(trg)="9"> 暑 假 最 麻 ? 烦 。

(src)="9"> They were endless .

(trg)="10"> ? 没 完 ? 没 了 。

(src)="10"> I ' m a little girl .

(trg)="11"> 我 是 一 ? 个 小 女 孩 。

(src)="11"> I don' t know , no , I don' t know .

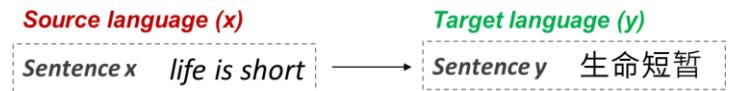
(trg)="12"> 我 不 知 道 , 不 , 我 不 知 道 。

How to learn translation model?

How to learn translation model from the parallel corpus?

$$\begin{array}{c} P(x|y) \\ \hline \downarrow \quad \downarrow \\ P(x, a|y) \end{array}$$

i.e. pairs of human-translated
Chinese/English sentences



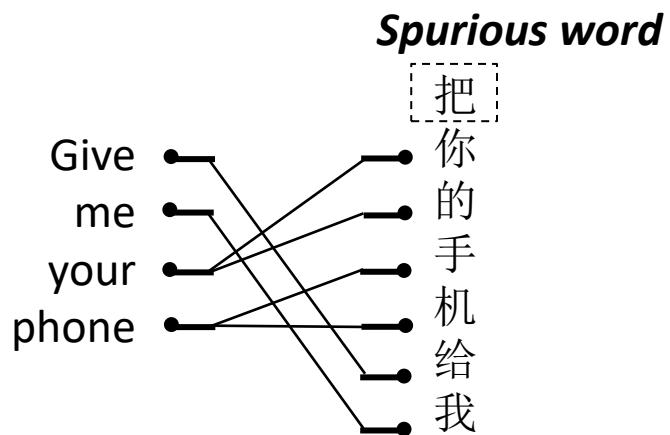
a is the **alignment**

Alignment is the correspondence between particular words in the translated sentence pair.
(i.e. word-level correspondence between **source sentence x** and **target sentence y**)

Statistical Machine Translation

What is Alignment **a**?

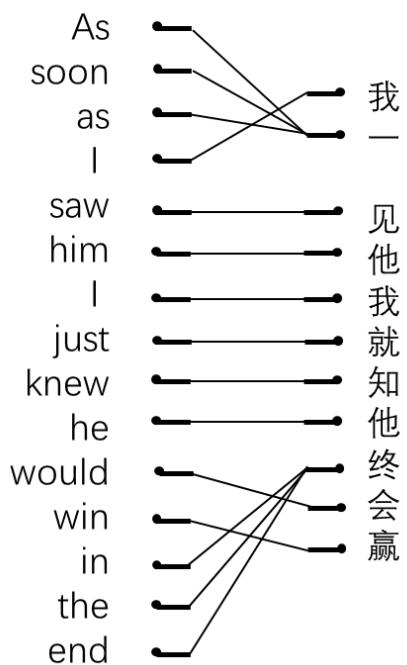
“The correspondence between particular words in the translated sentence pair”



	把	你	的	手	机	给	我
Give							
me							
your							
phone							

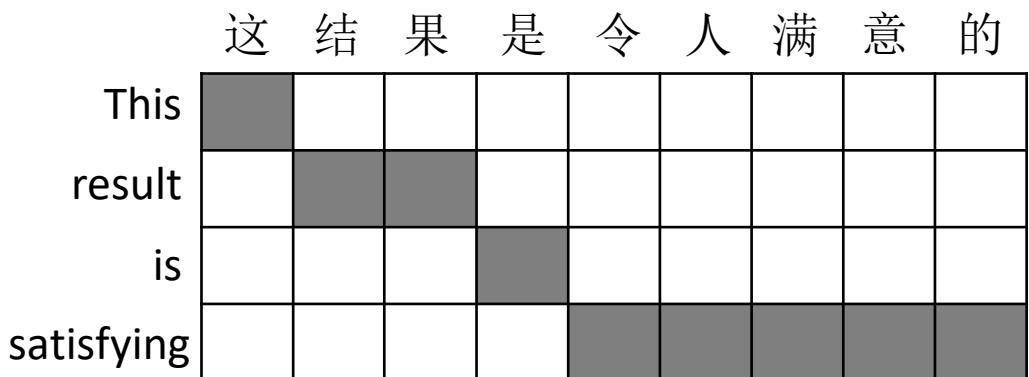
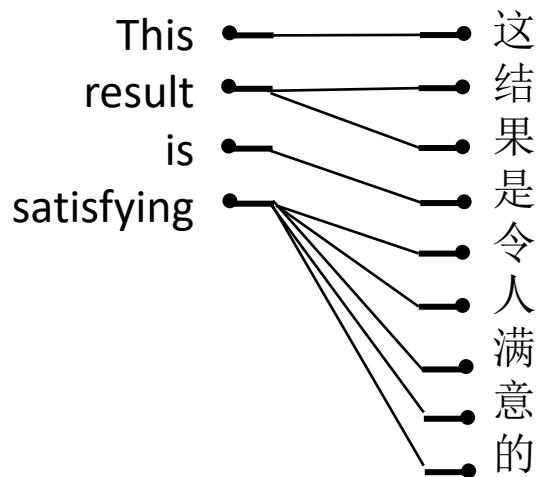
What is Alignment *a*?

Many-to-One Alignment



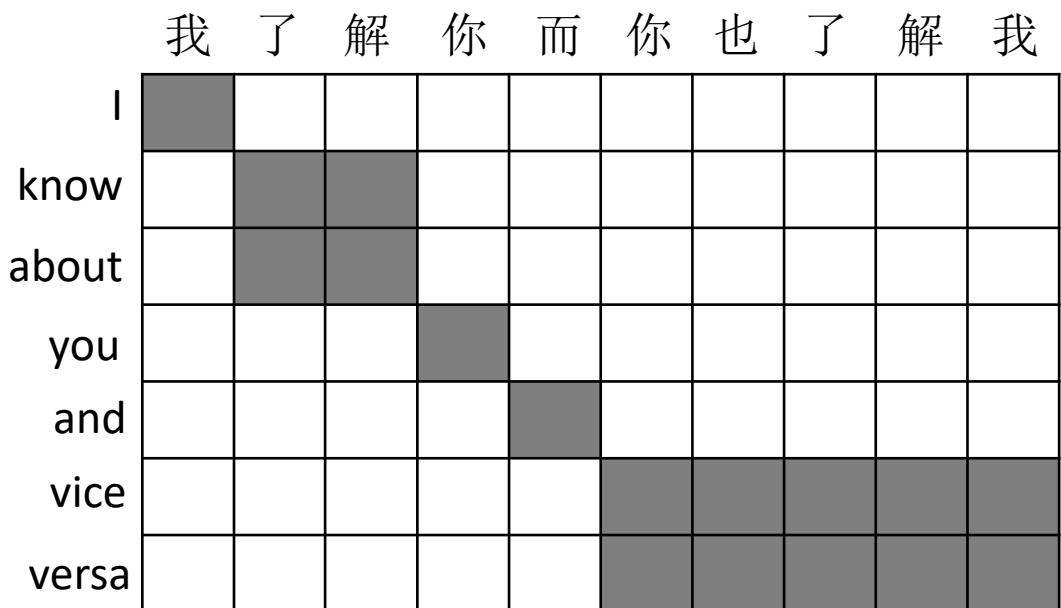
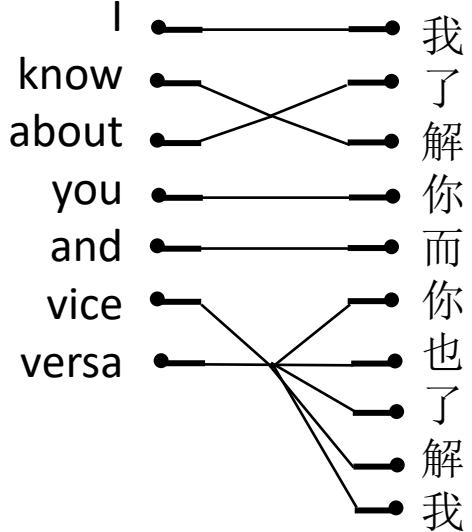
What is Alignment α ?

One-to-Many Alignment



What is Alignment α ?

Many-to-many Alignment



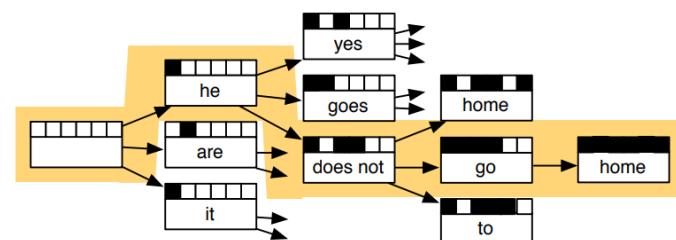
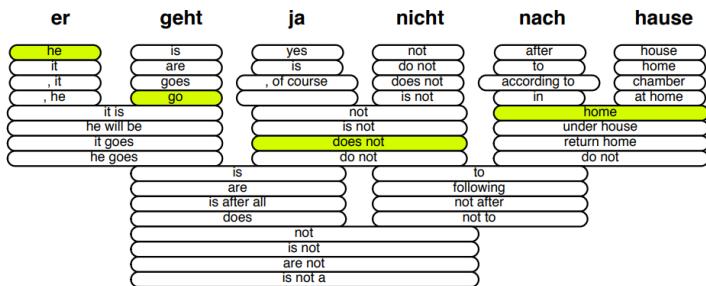
Decoding for SMT

$$\operatorname{argmax}_y \underline{P(x|y)} \underline{P(y)}$$

Translation Model (fidelity)
Learnt from **parallel data**

Language Model (fluency)
Learnt from **monolingual data**

- We could enumerate every possible y and calculate the probability?
Too expensive!
- Answer: Use a **heuristic search algorithm** to **search for the best** translation, discarding hypotheses that are too low-probability



backtrack from highest scoring complete hypothesis

Statistical Machine Translation

Statistical Machine Translation

The Best System

SMT was a huge research field and Extremely complex System

Hundreds of important details (haven't mentioned here)

- *Systems had many separately-designed subcomponents*
- *Lots of feature engineering*
 - *Need to design features to capture particular language phenomena*
- *Require compiling and maintaining extra resources*
 - *Like tables of equivalent phrases*
- *Lots of human effort to maintain*
 - *Repeated effort for each language pair!*

3

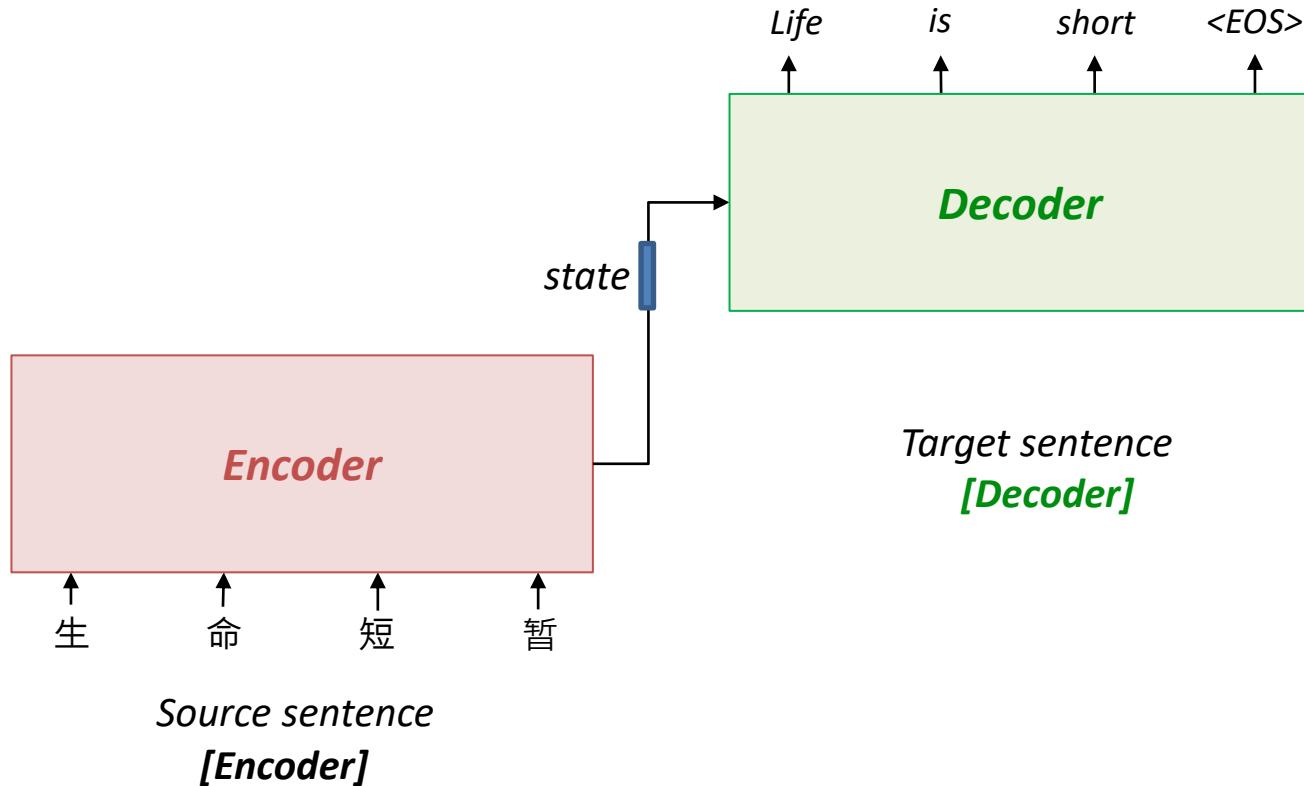
Neural Machine Translation



Neural Machine Translation with Seq2Seq

“a way to do Machine Translation with a single neural network (NN)”

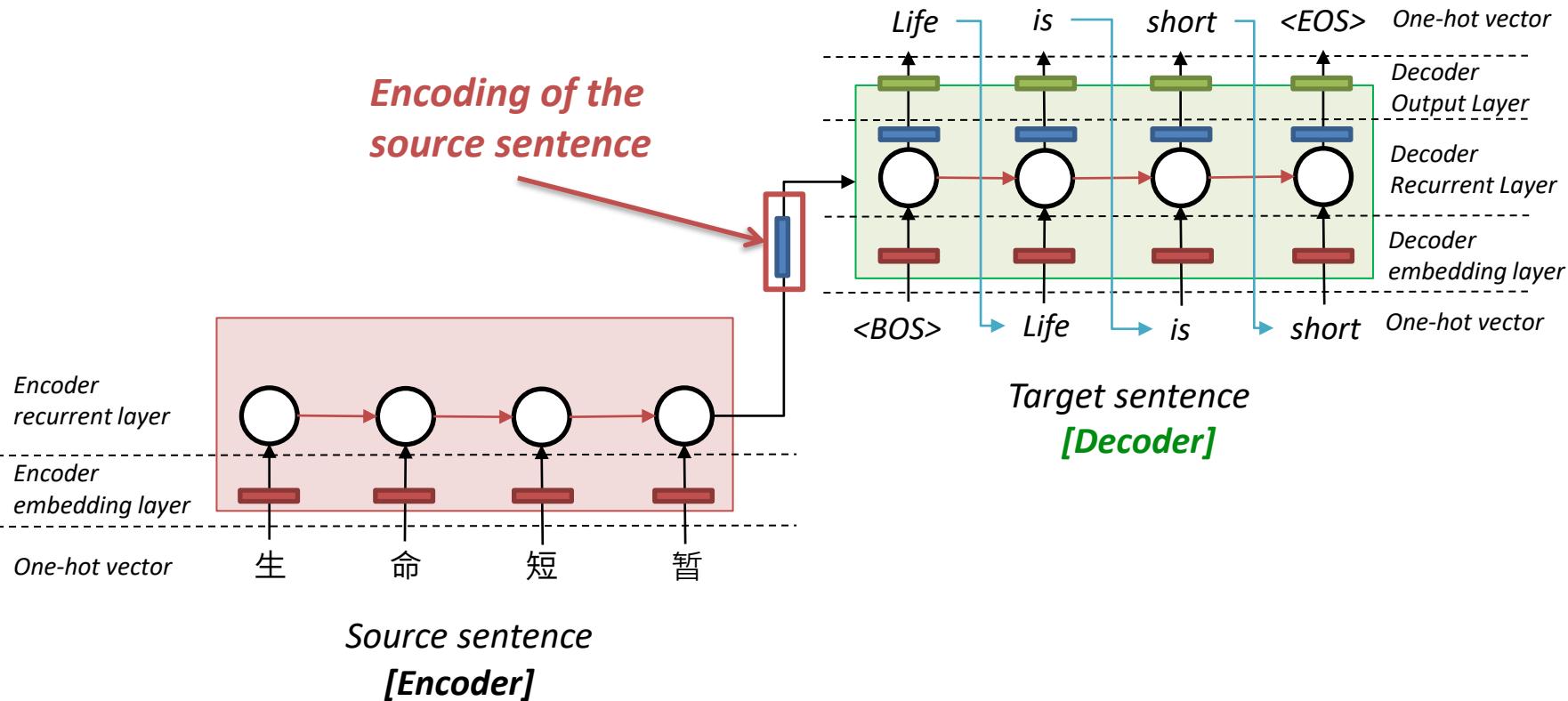
- *The NN architecture is called seq2seq and involves two RNNs.*



Neural Machine Translation with Seq2Seq

“a way to do Machine Translation with a single neural network (NN)”

- The NN architecture is called *seq2seq* and involves two RNNs.

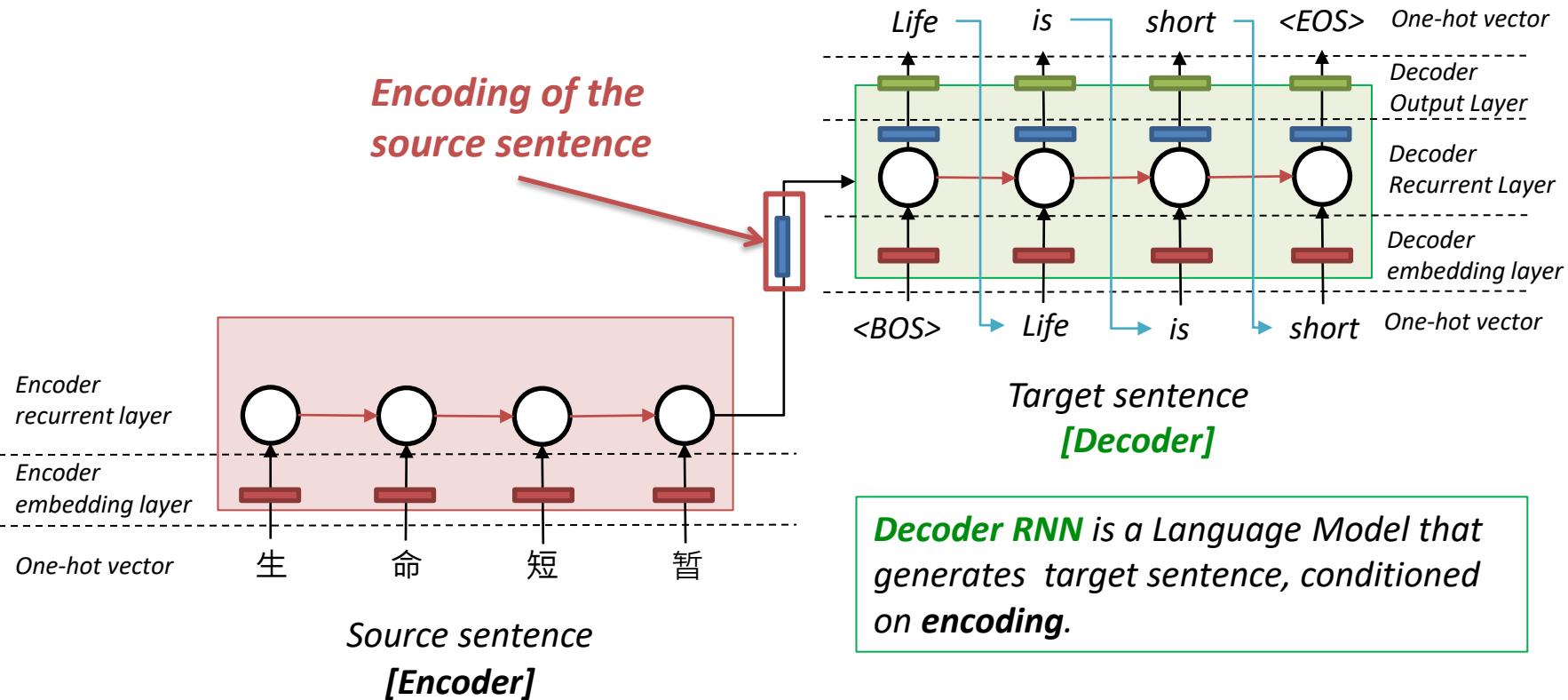


Neural Machine Translation

Neural Machine Translation with Seq2Seq

“a way to do Machine Translation with a single neural network (NN)”

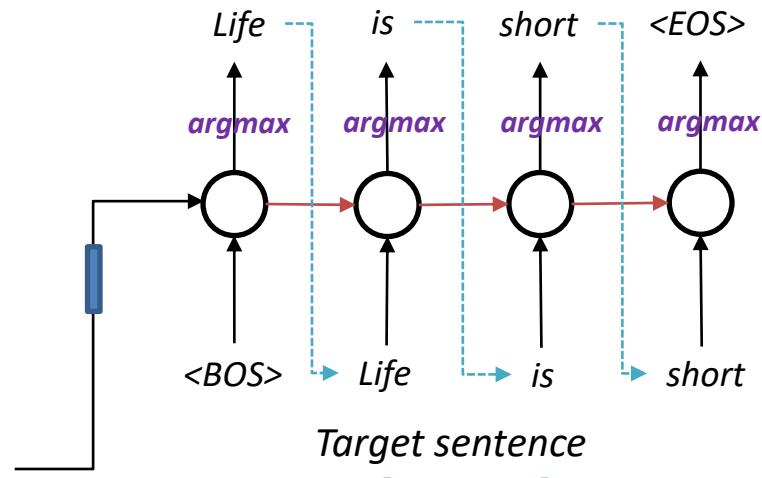
- The NN architecture is called *seq2seq* and involves two RNNs.



Neural Machine Translation: Greedy Decoding [Recap]

Language Model Decoding: Recap

- Generate the sentence by taking *argmax* (the most probable word) on each step
- Use that as the next word, and feed it as input on the next step
- Keep going until you produce *<EOS>*



*Target sentence
[Decoder]*

***Greedy decoding has no way to undo decisions!!
(Ungrammatical, unnatural)***

Solution..? try computing all possible sequences

3

Neural Machine Translation

Neural Machine Translation: Beam Search Decoding [Recap]

Language Model Decoding: Recap

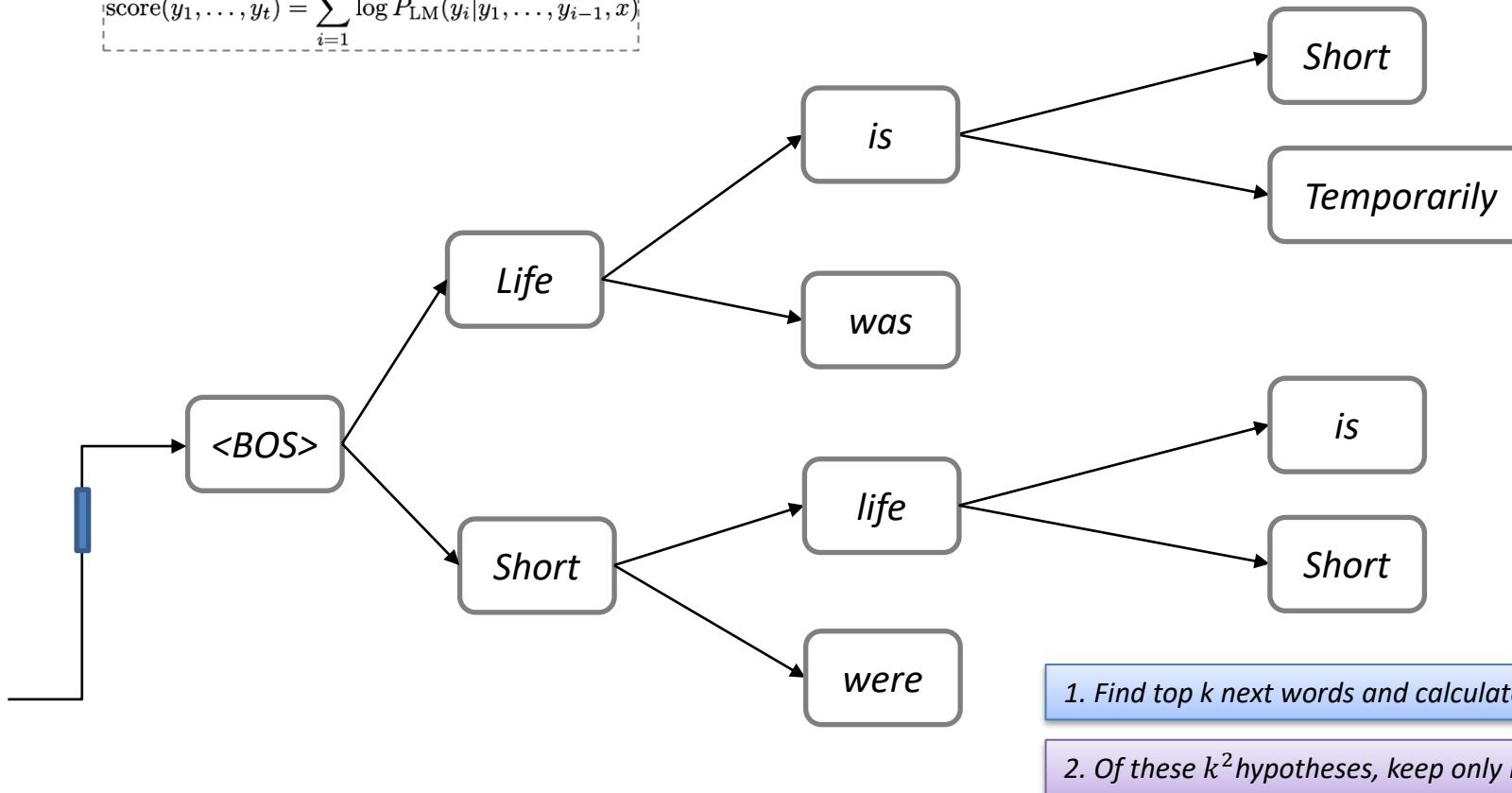
- A search algorithm which aims to find a high-probability sequence (not necessarily the optimal sequence, though) by tracking multiple possible sequences at once.
- On each step of decoder, keep track of the k most probable partial sequences (which we call hypotheses)
- K is the beam size (in practice around 5 to 10)
- After you reach some stopping criterion, choose the sequence with the highest probability (factoring in some adjustment for length)

Neural Machine Translation: Beam Search Decoding

Language Model Decoding: Recap

Assume that $k(\text{beam size})=2$

$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$



Neural Machine Translation

Evaluate Machine Translation

BLEU (*Bilingual Evaluation Understudy*)

*“Compares the **machine-written translation** to one or several **human-written translation(s)**, and computes a similarity score based on”*

- *n-gram precision (usually for 1 to 4-grams)*
- *Plus a penalty for too-short system translations*

BLEU is useful but imperfect

- *Many valid ways to translate a sentence*
- *So a good translation can get a poor BLEU score because it has low n-gram overlap with the human translation*

3

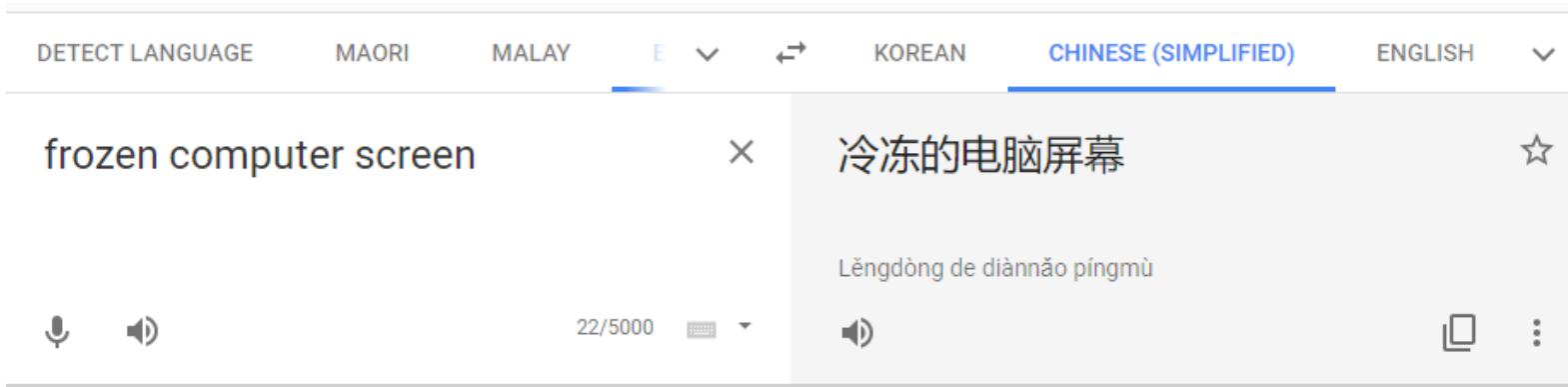
Neural Machine Translation

However, there are still several difficulties...

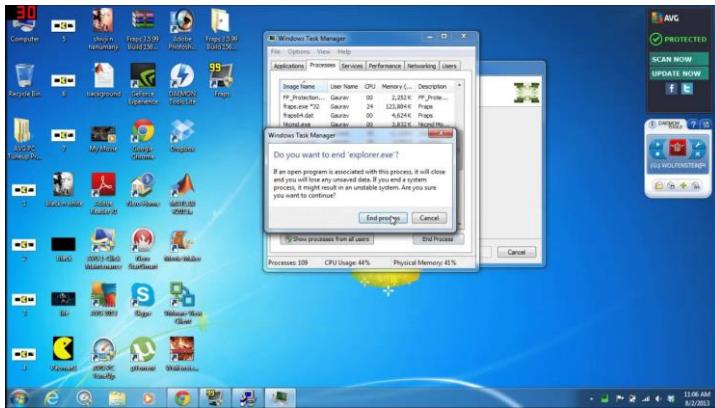
- *Out-of-vocabulary (OOV) words*
- *Domain mismatch between train and test data*
- *Maintaining context over longer text*
- *Low-resource language pairs*

Neural Machine Translation

Machine Translation is not **PERFECT...**



The screenshot shows a machine translation interface with tabs for DETECT LANGUAGE, MAORI, MALAY, KOREAN, CHINESE (SIMPLIFIED), and ENGLISH. The CHINESE (SIMPLIFIED) tab is selected. On the left, the English phrase "frozen computer screen" is entered. On the right, the Chinese translation "冷冻的电脑屏幕" is displayed with its pinyin equivalent "Lěngdòng de diànnǎo píngmù". Below the text are standard UI elements like a microphone icon, volume slider, and file operations.

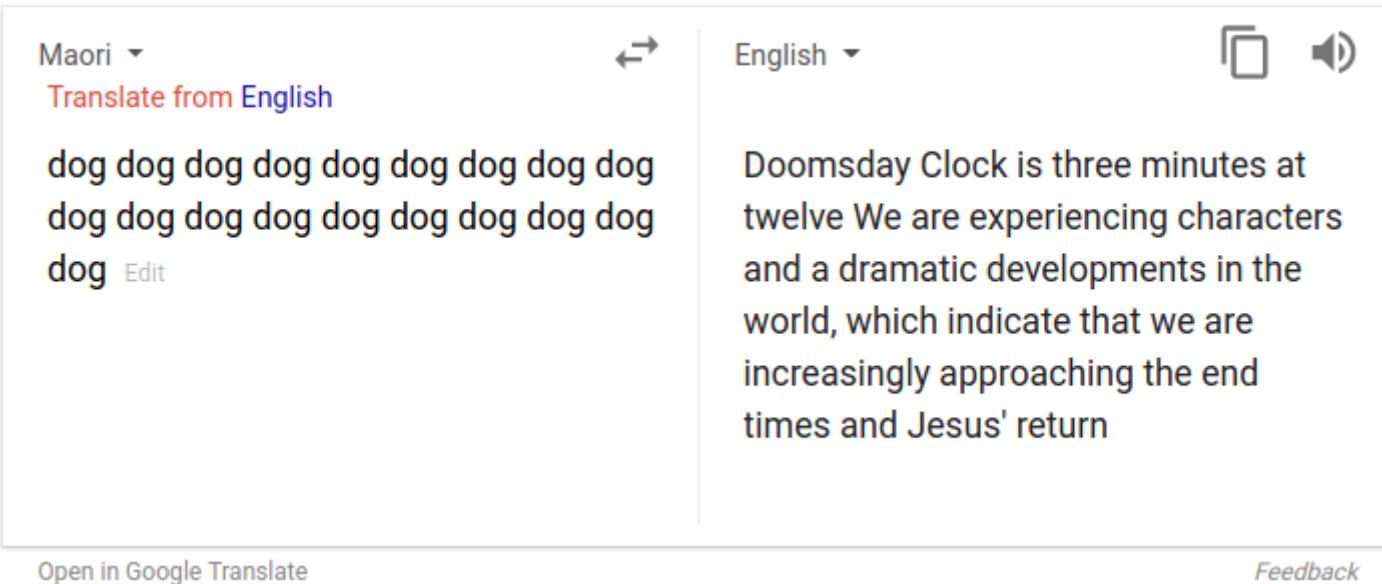


Using common sense is still hard and NMT picks up biases in training data

3

Neural Machine Translation

Machine Translation is not PERFECT...



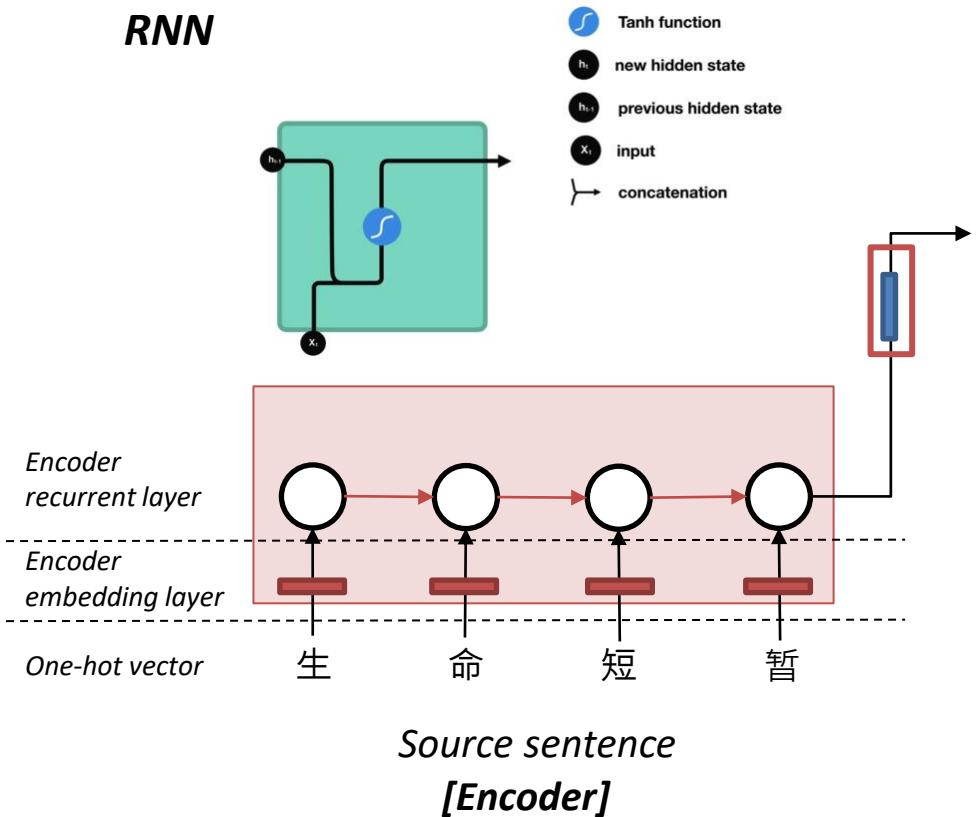
The screenshot shows a Google Translate interface. On the left, the source language is set to Maori, and the target language is English. The input text in Maori is "dog dog dog". Below the input is a red link "Translate from English". The output in English is: "Doomsday Clock is three minutes at twelve We are experiencing characters and a dramatic developments in the world, which indicate that we are increasingly approaching the end times and Jesus' return". At the bottom left is a link "Open in Google Translate", and at the bottom right is a link "Feedback".

Uninterpretable systems do strange things

Neural Machine Translation with Seq2Seq

RNN-based neural MT was sort of successful! But...

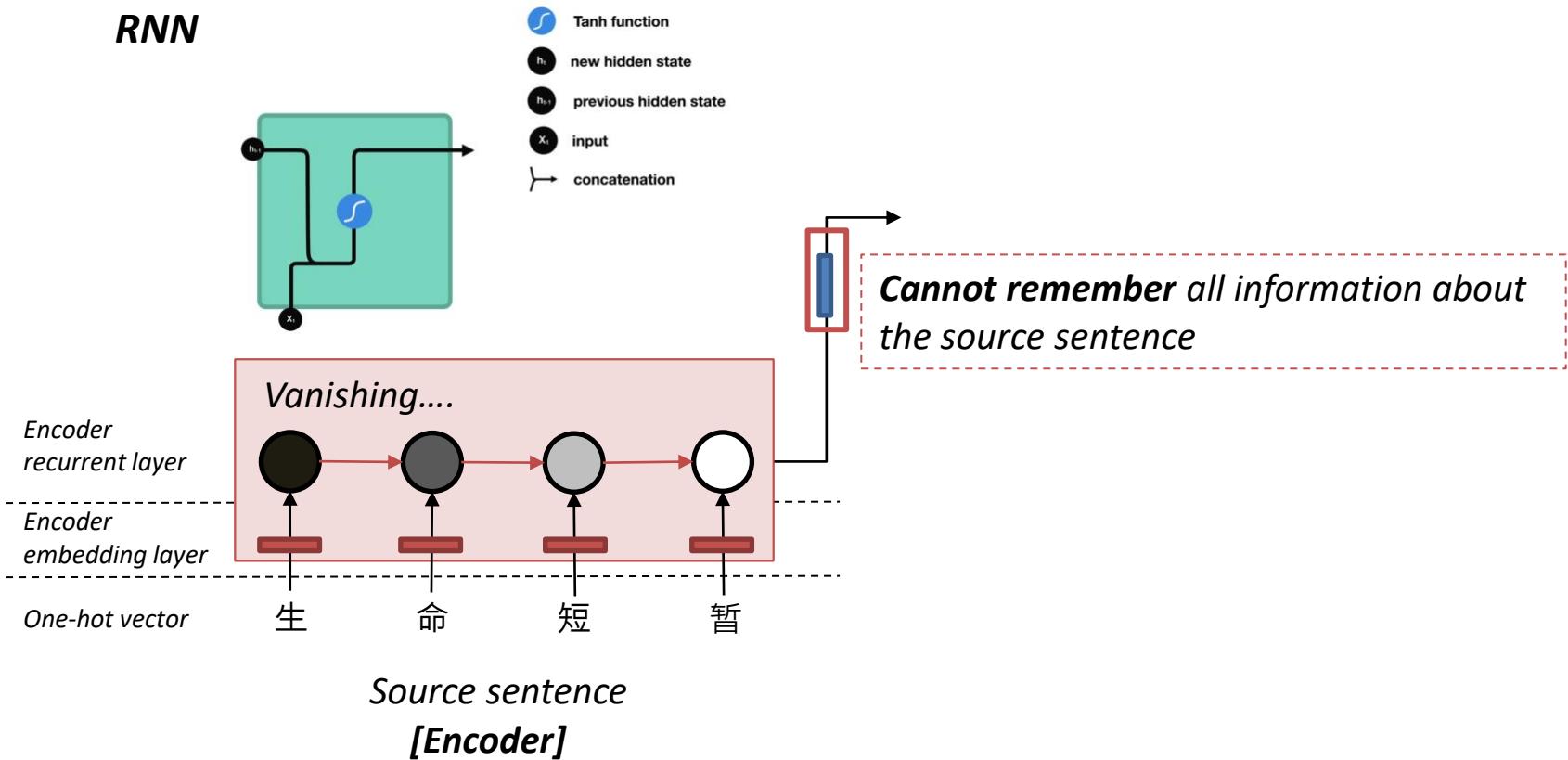
RNN



Neural Machine Translation with Seq2Seq

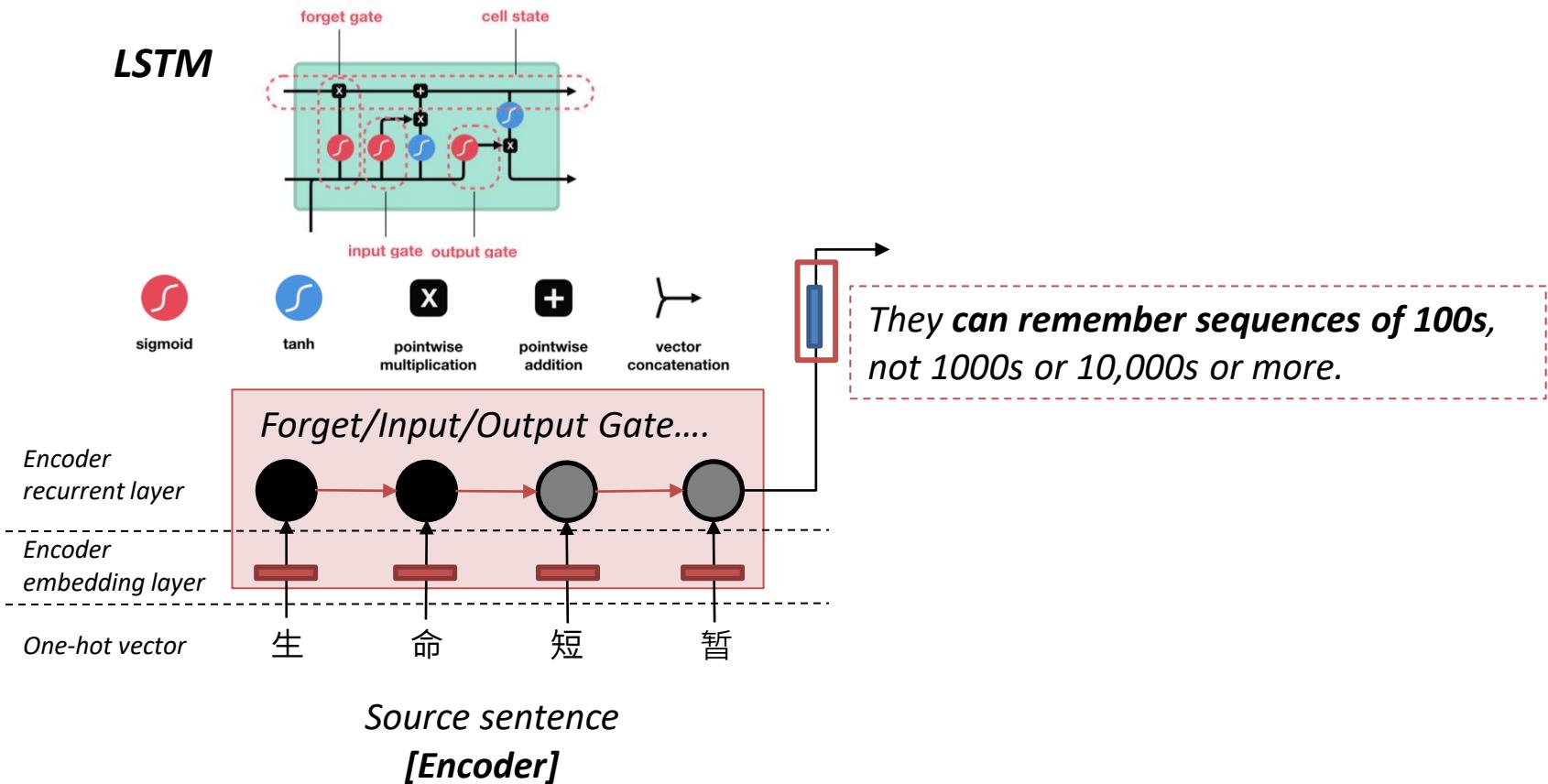
RNN-based neural MT was sort of successful! But...

RNN



Neural Machine Translation with Seq2Seq

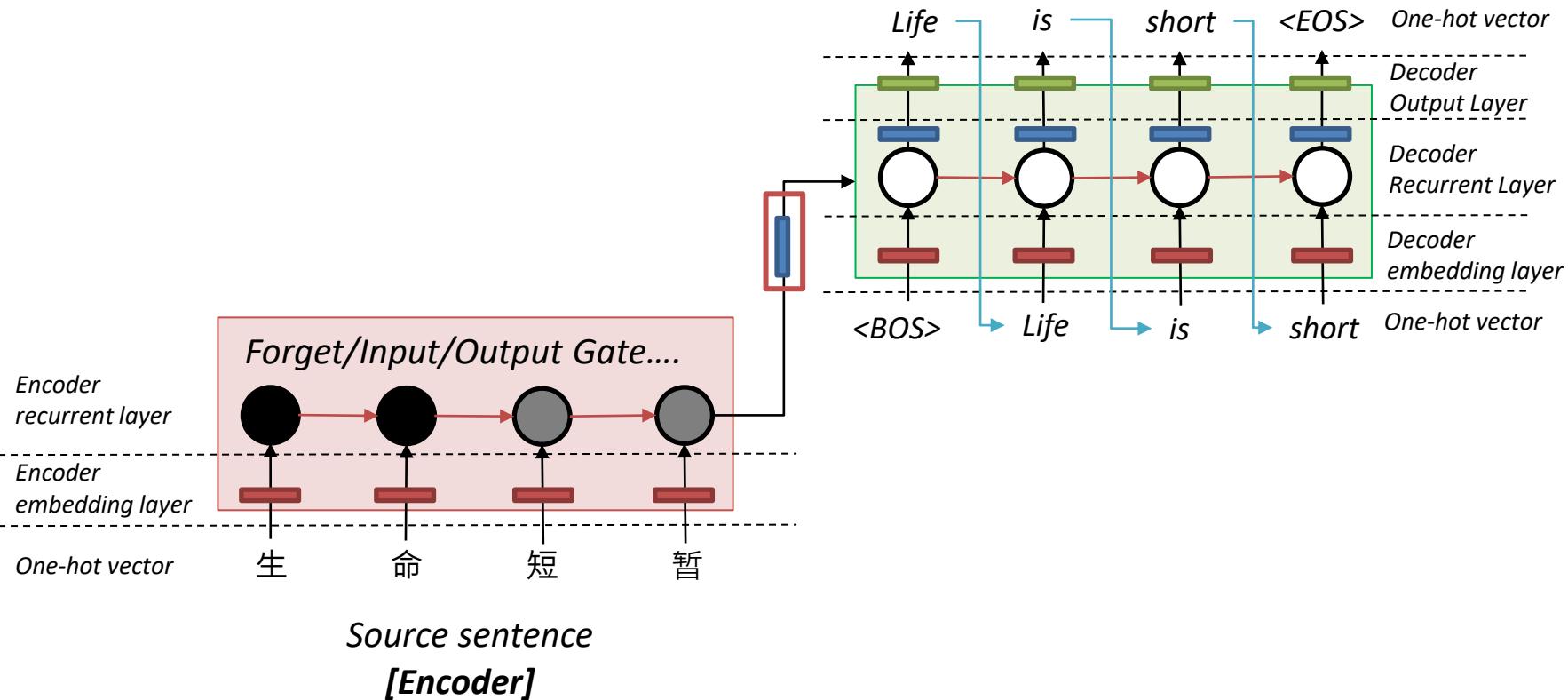
RNN-based neural MT was successful! But...



Neural Machine Translation with Seq2Seq

Then, how to solve the information bottleneck issue?

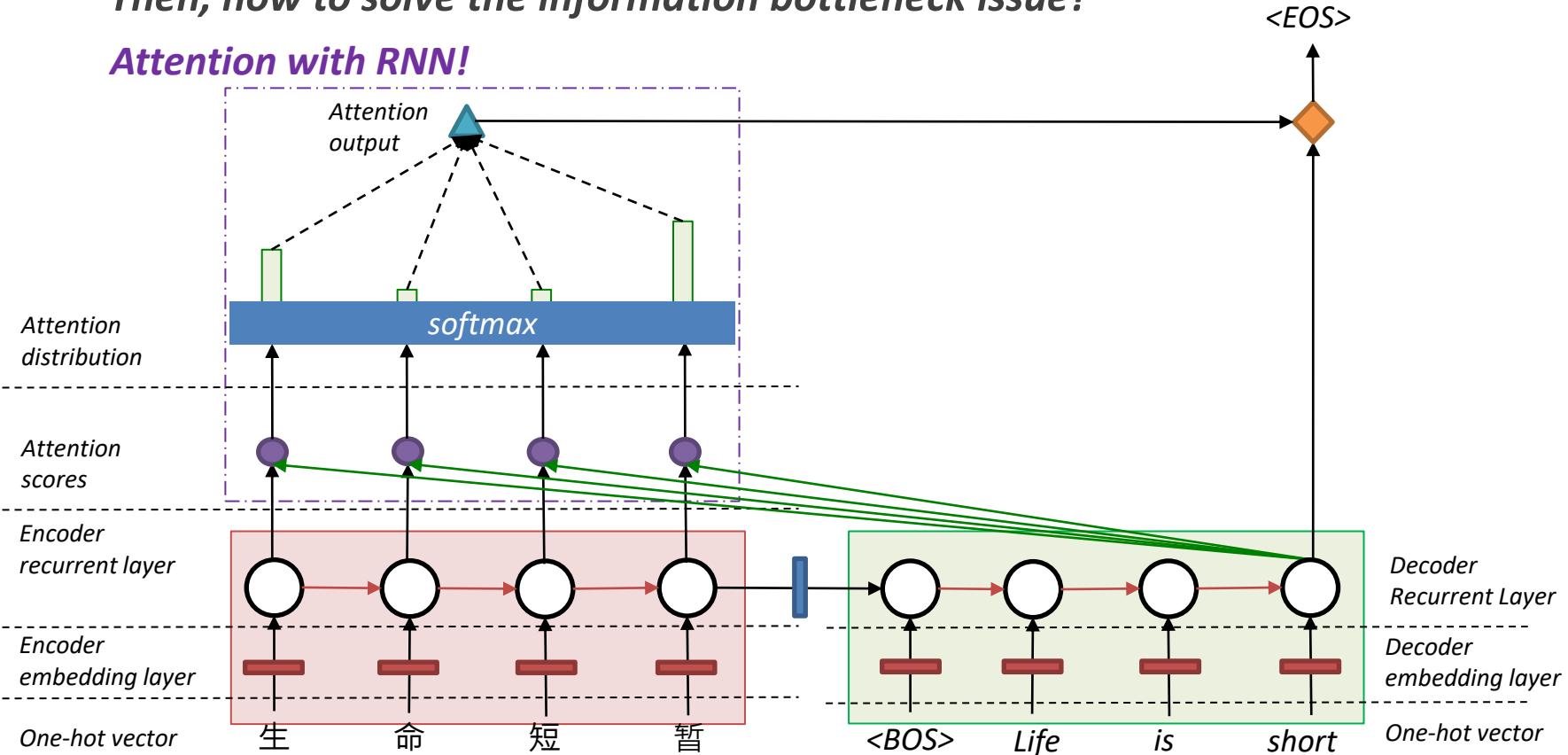
Attention!



Neural Machine Translation with RNN and Attention

Then, how to solve the information bottleneck issue?

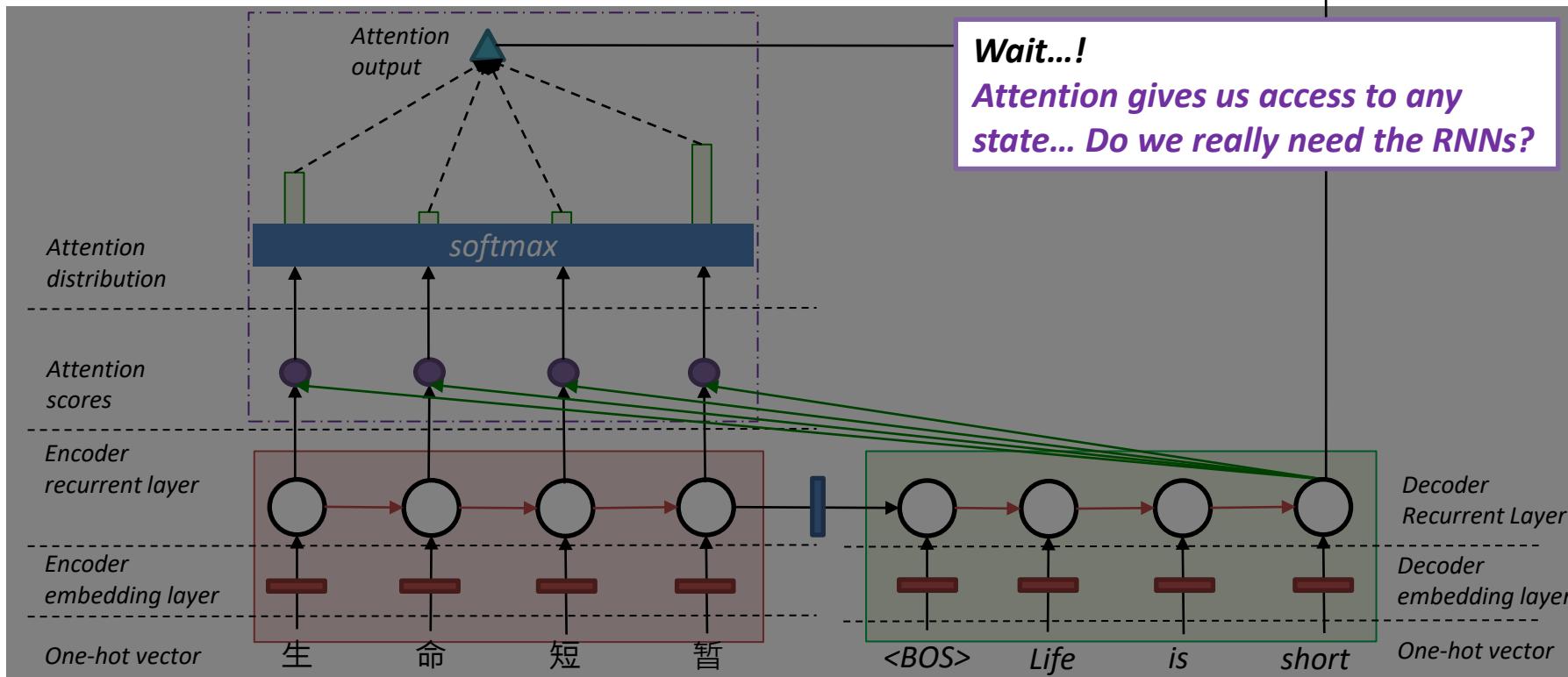
Attention with RNN!



Neural Machine Translation with RNN and Attention

Then, how to solve the information bottleneck issue?

Attention with RNN!



4

Attention and Transformer for MT

Early 2018 ~

Attention is All You Need (Vaswani et al., 2017)

Encoder-Decoder with only Attention

Core Task: Machine Translation with Parallel Corpus

- Use self-attention in the encoder, instead of RNN or CNNs
- Predict each translated word
- Final cost/error function
- standard cross-entropy error on top of a softmax classifier

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Lukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Ilia Polosukhin* ‡
illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms to translate from one language to another entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Attention is All You Need!



'The Transformer'!!

The Transformer!

こんにちは世界

Input (Source Language)

Output (Target Language)

Hello World



Attention is All You Need (Vaswani et al., 2017)

Encoder-Decoder with only Attention

Core Task: Machine Translation with Parallel Corpus

- Use self-attention in the encoder, instead of RNN or CNNs
- Predict each translated word
- Final cost/error function
- standard cross-entropy error on top of a softmax classifier

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser*
Google Brain
lukasz.kaiser@google.com

Ilia Polosukhin* ‡
illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, without recurrent hidden states or convolutional layers. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Attention is All You Need?



'The Transformer'!!

Encoder

こんにちは世界

Input (Source Language)

Output (Target Language)

Hello World

Decoder

Attention and Transformer for MT

The Transformer



Encoder – Decoder Architecture

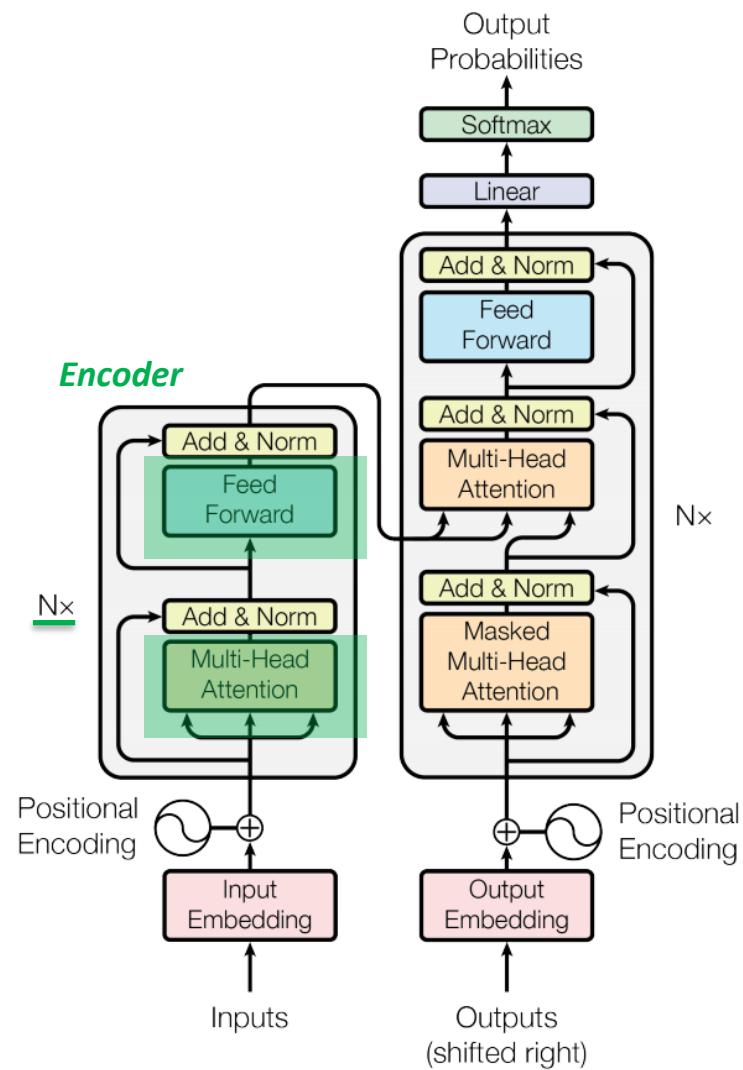
1. Encoder

A stack of **N=6 identical layers**.

Each layer with two sub-layers:

1. Multi-head self-attention mechanism
2. Position-wise fully connected feed-forward network

* Residual connection around each of the two sub-layers, followed by layer normalisation



The transformer – model architecture

The Transformer



Encoder – Decoder Architecture

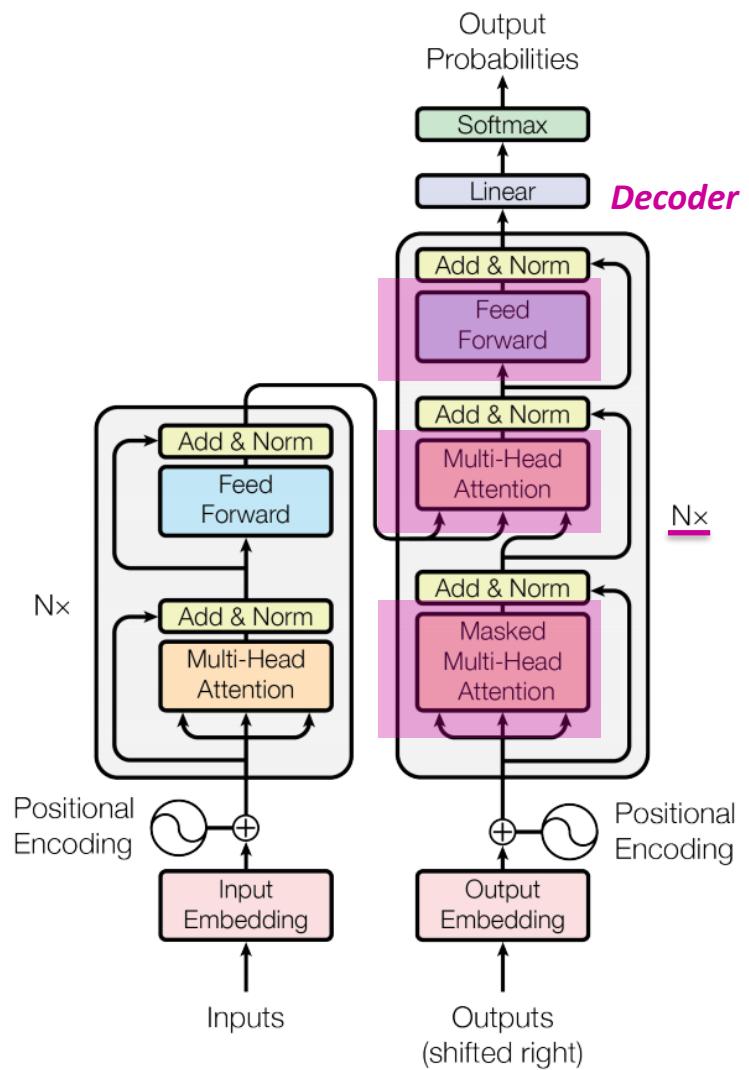
2. Decoder

A stack of $N=6$ identical layers.

Each layer with three sub-layers:

1. Multi-head self-attention mechanism
2. Position-wise fully connected feed-forward network
3. Masked Multi-head self-attention

* Residual connection around each of the two sub-layers, followed by layer normalisation



The transformer – model architecture

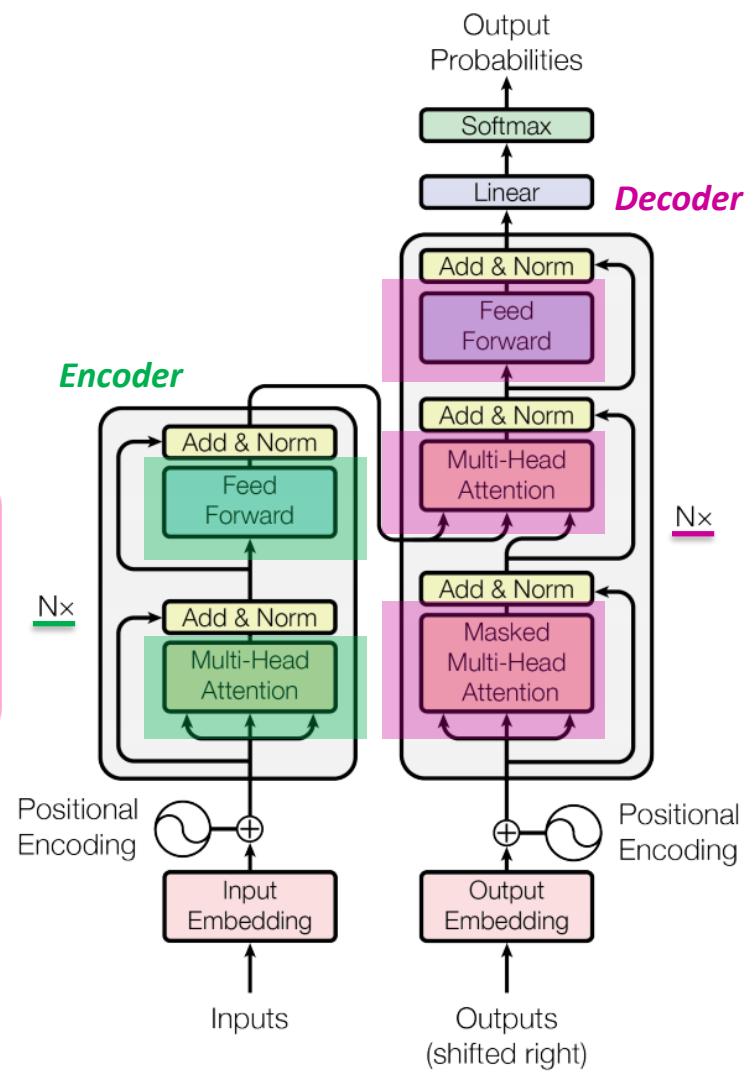
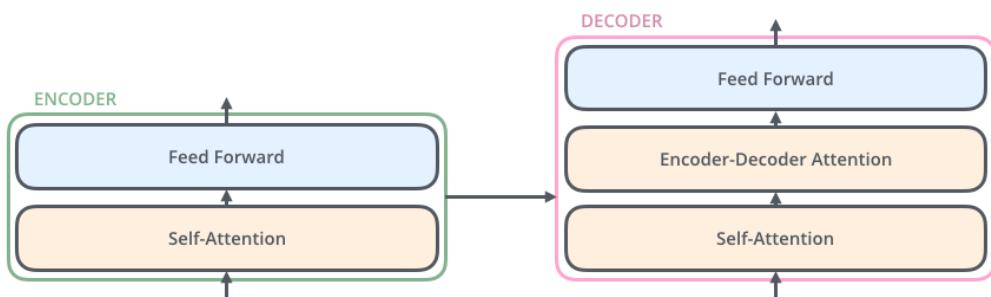
Attention and Transformer for MT

The Transformer



Encoder – Decoder Architecture

Brief Summary



The transformer – model architecture

Attention and Transformer for MT

The Transformer – Encoder (Stage1)

We are not using RNN anymore... No time step concept!

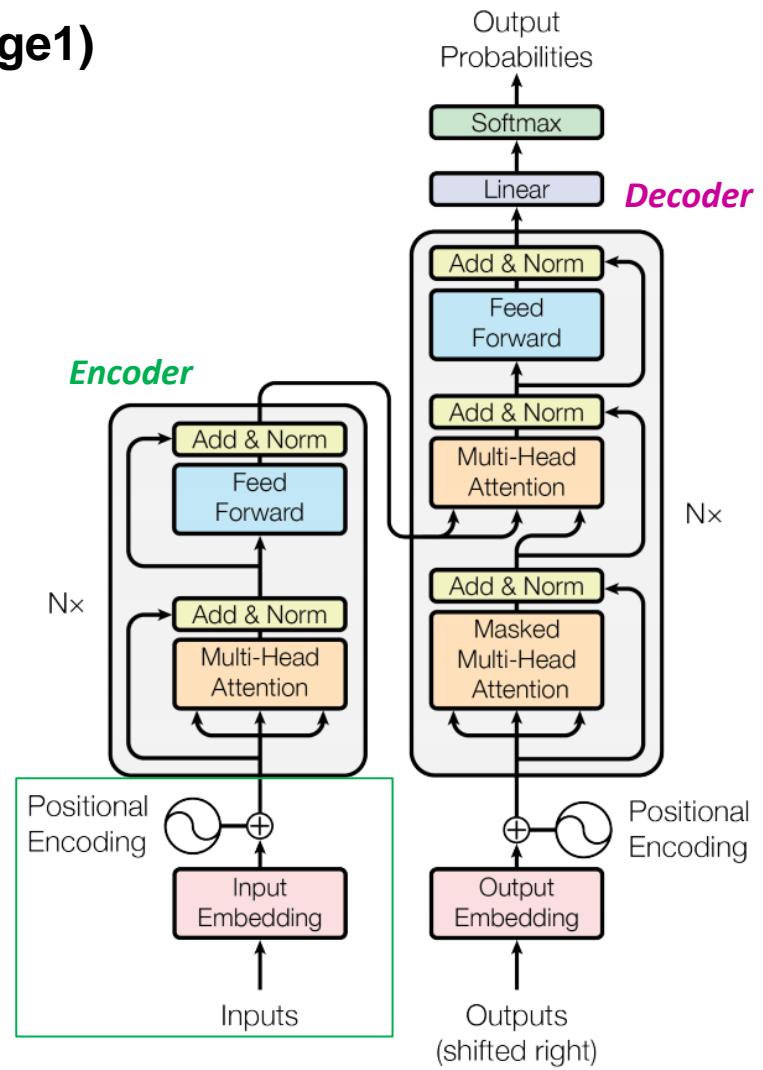
To make use of the order of the sequence,
inject information about the position of
the tokens in the sequence.

Positional Encoding
(use sin and cos for
position/dimension)

Input embedding
(a vector of size 512)

$$\begin{array}{c} \begin{matrix} 0 & 0 & 1 & 1 \end{matrix} \\ + \\ \begin{matrix} x_1 & \text{[green]} & \text{[green]} & \text{[green]} \end{matrix} \end{array} \quad \begin{array}{c} \begin{matrix} 0.91 & 0.0002 & -0.42 & 1 \end{matrix} \\ + \\ \begin{matrix} x_2 & \text{[green]} & \text{[green]} & \text{[green]} \end{matrix} \end{array}$$

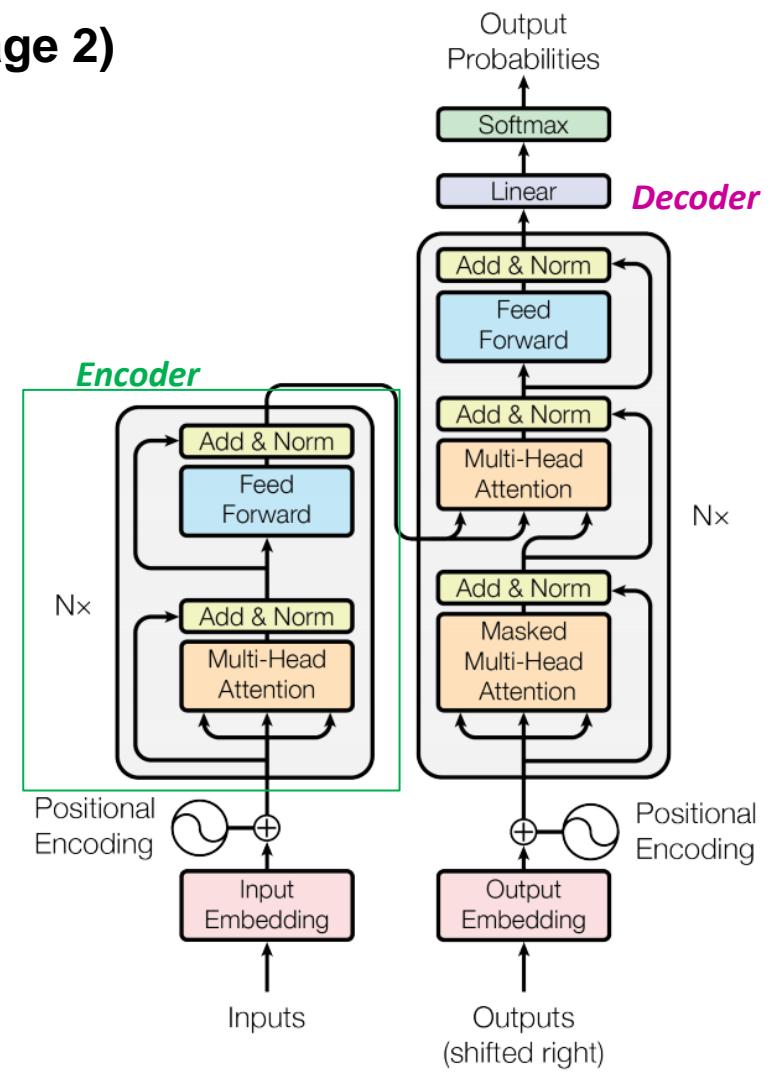
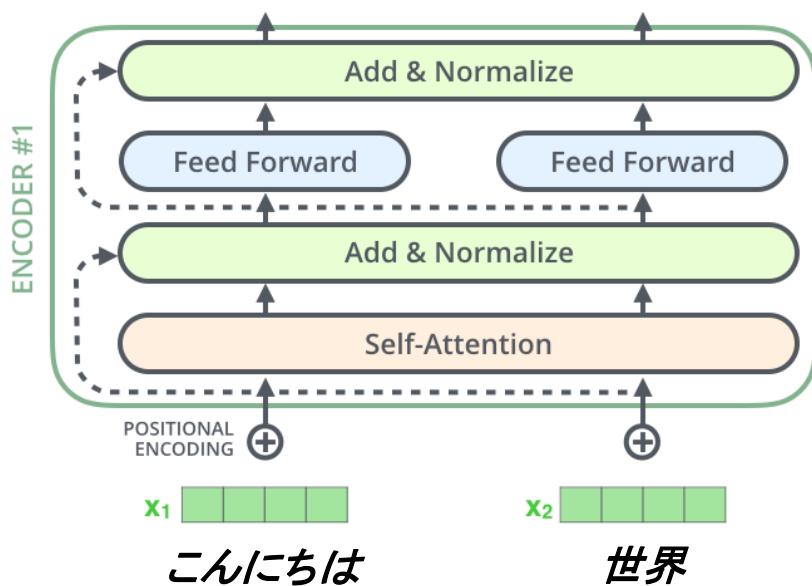
こんにちは 世界



The transformer – model architecture

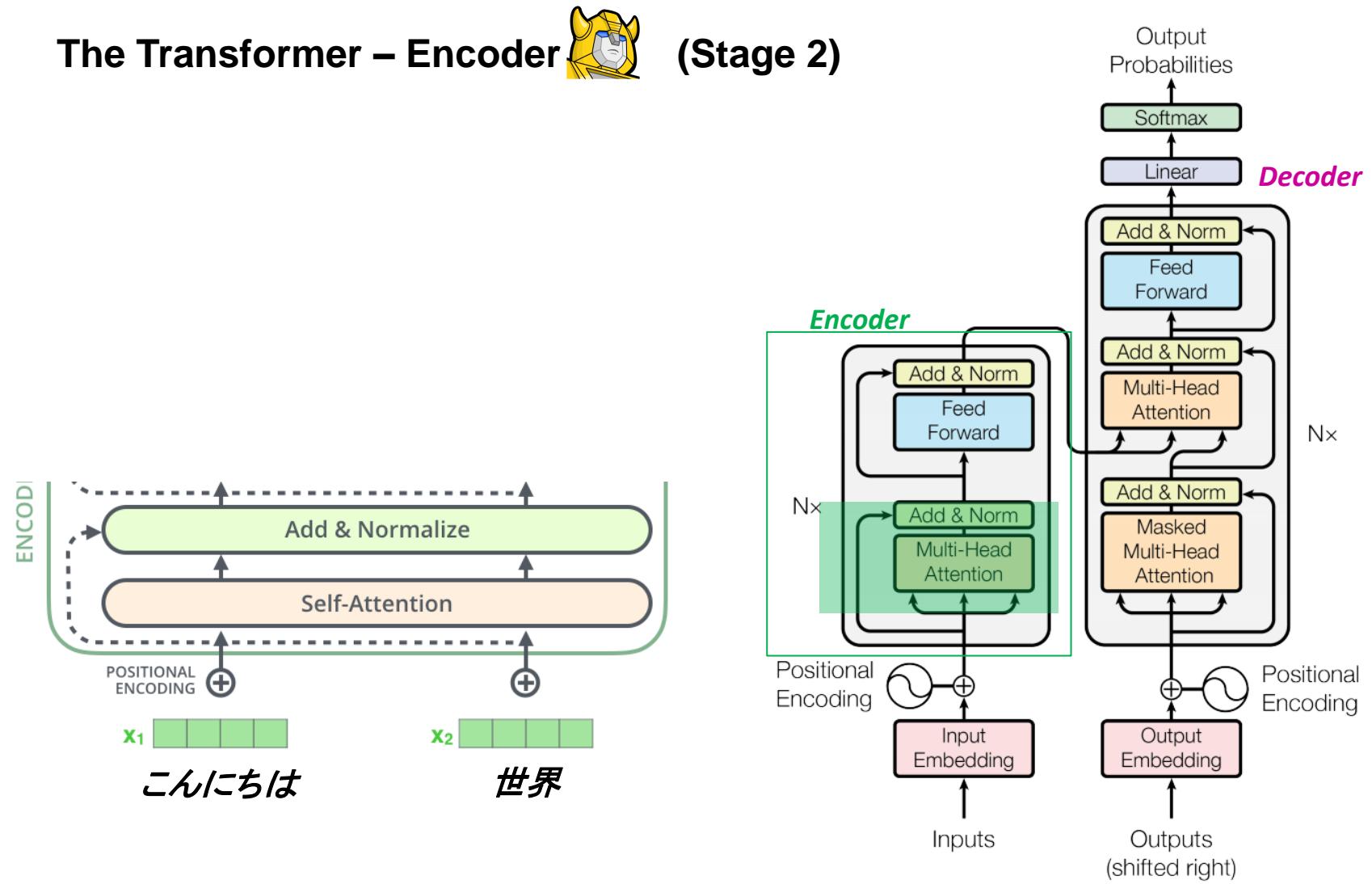
Attention and Transformer for MT

The Transformer – Encoder (Stage 2)



The transformer – model architecture

Attention and Transformer for MT

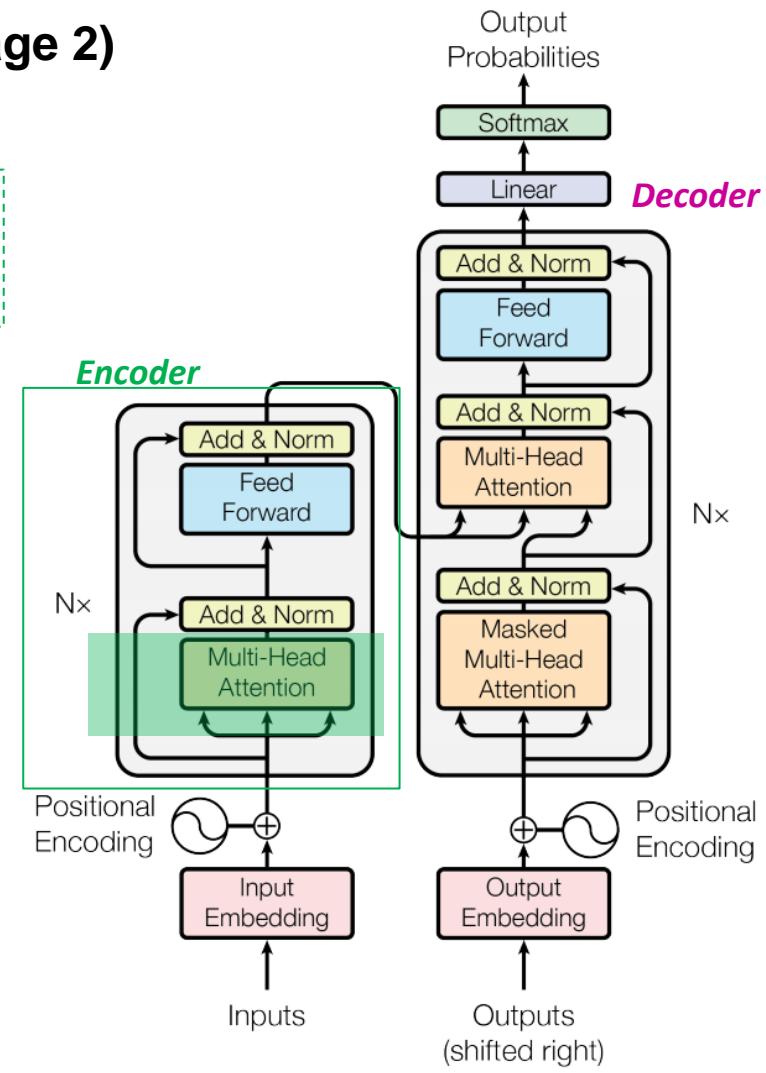
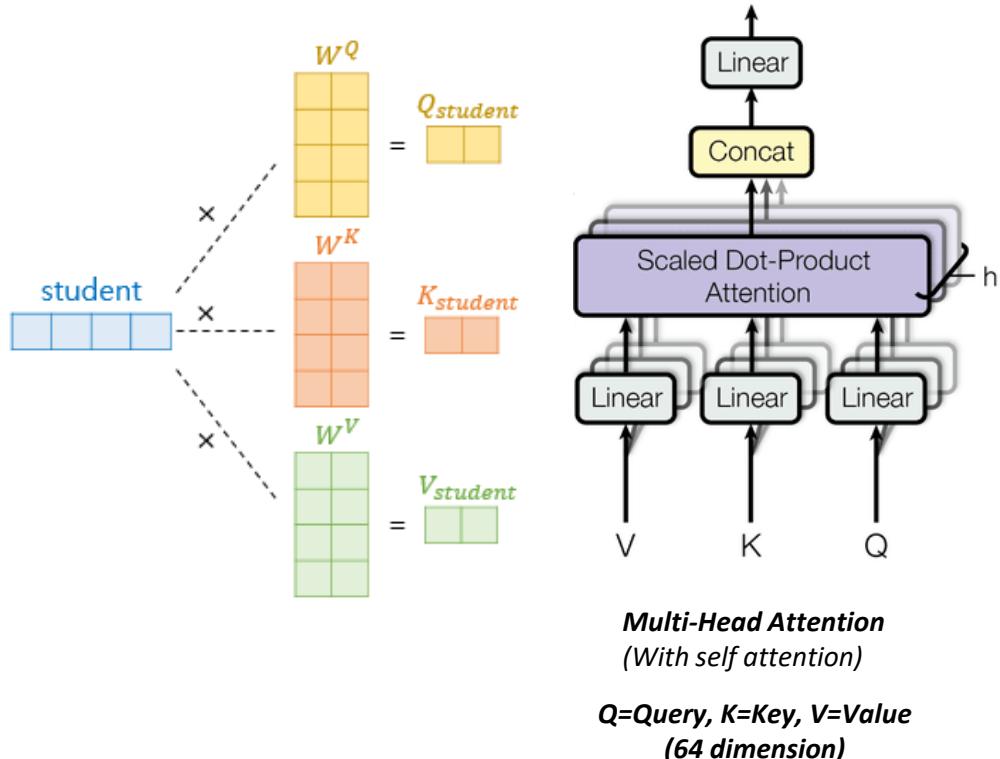


The transformer – model architecture

Attention and Transformer for MT

The Transformer – Encoder (Stage 2)

Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions

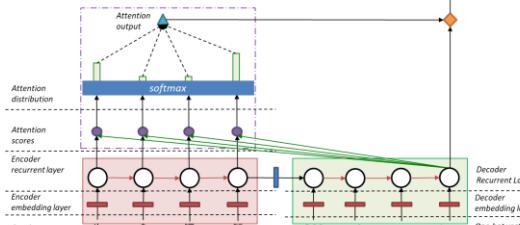


The transformer – model architecture

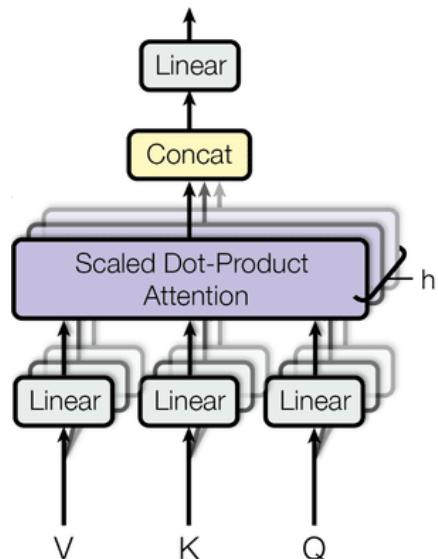
Attention and Transformer for MT

The Transformer – Encoder (Stage 2)

Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions

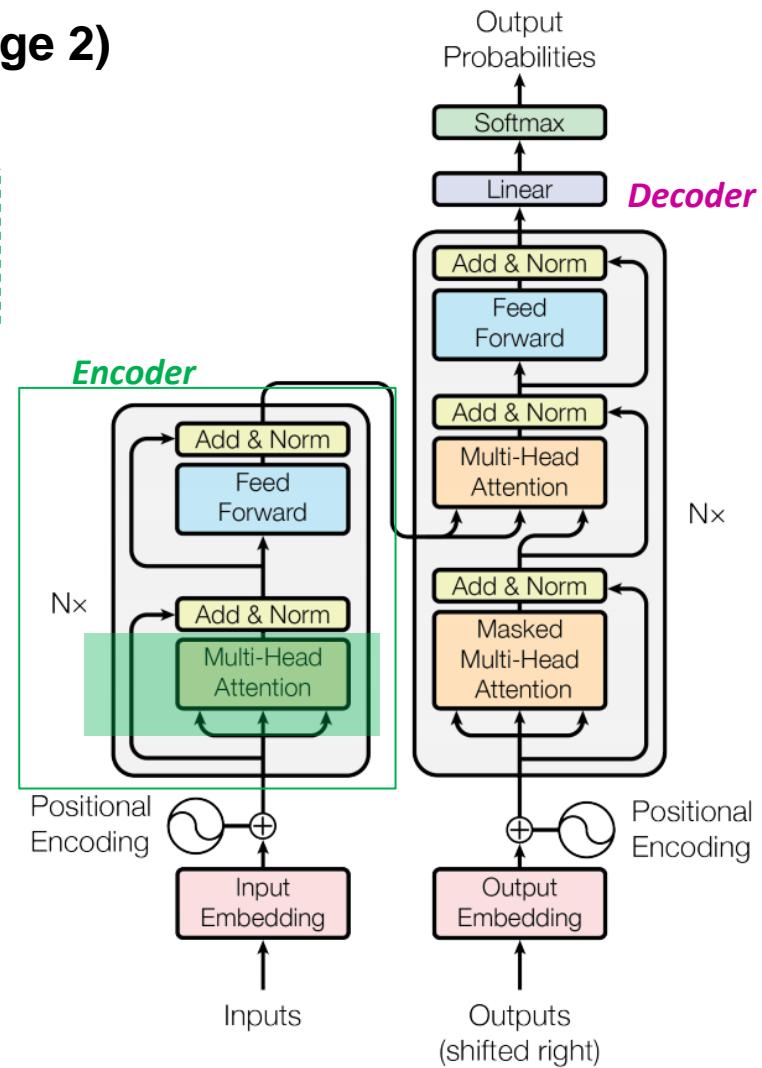


Source and Target attention



Multi-Head Attention
(With self attention)

$Q=Query$, $K=Key$, $V=Value$
(64 dimension)

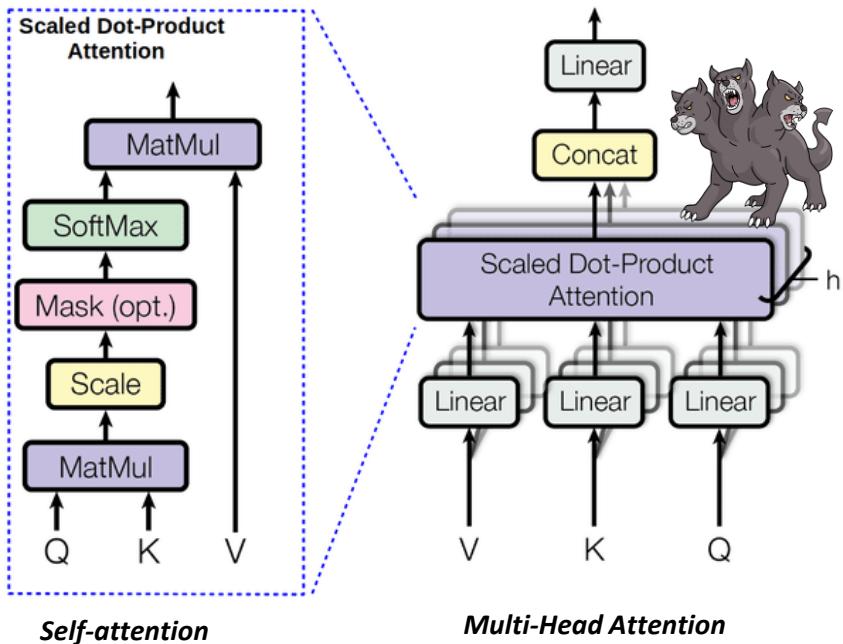


The transformer – model architecture

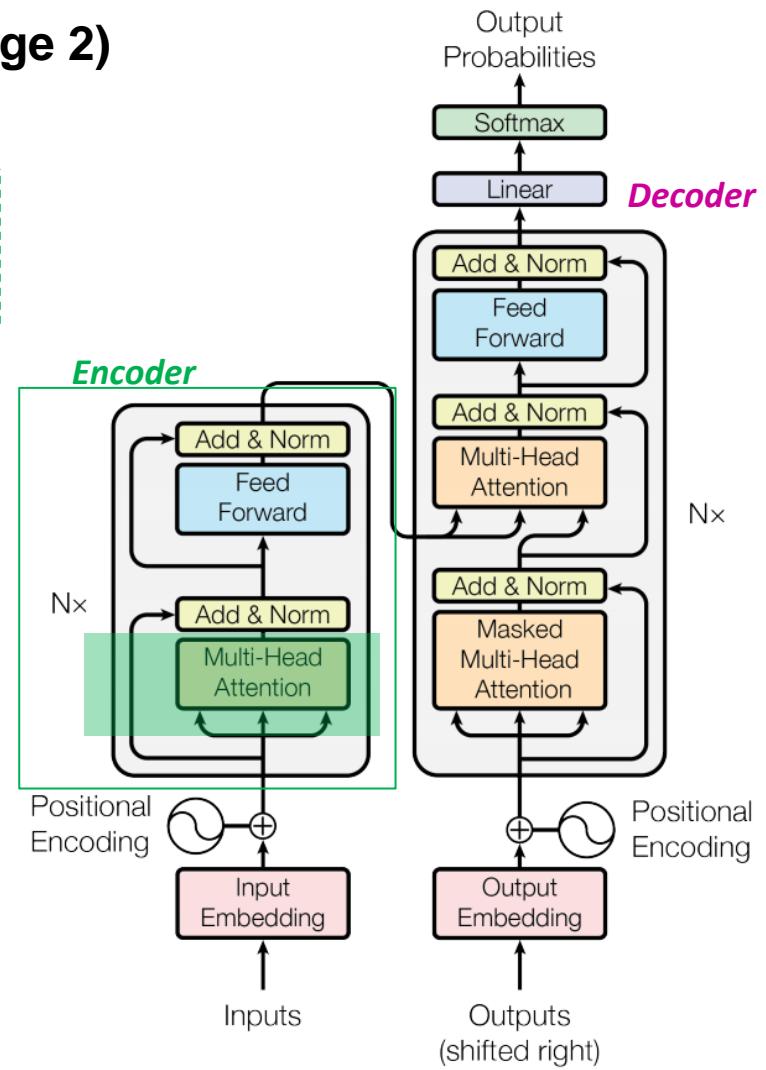
Attention and Transformer for MT

The Transformer – Encoder (Stage 2)

Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions

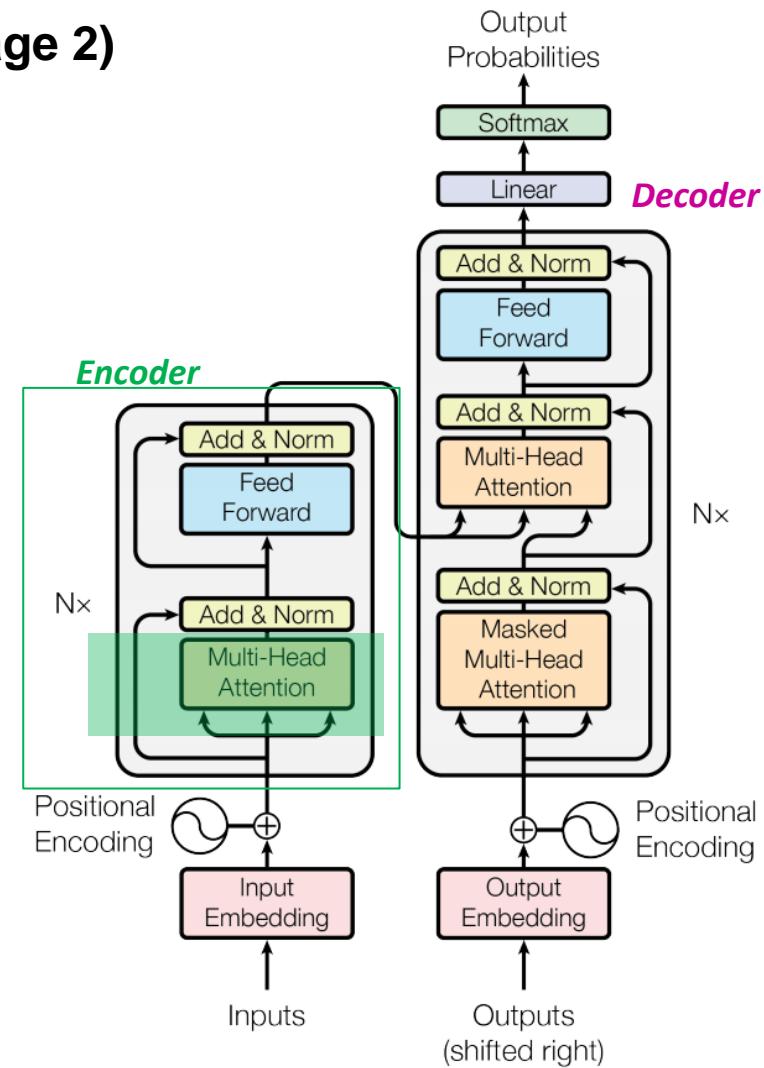
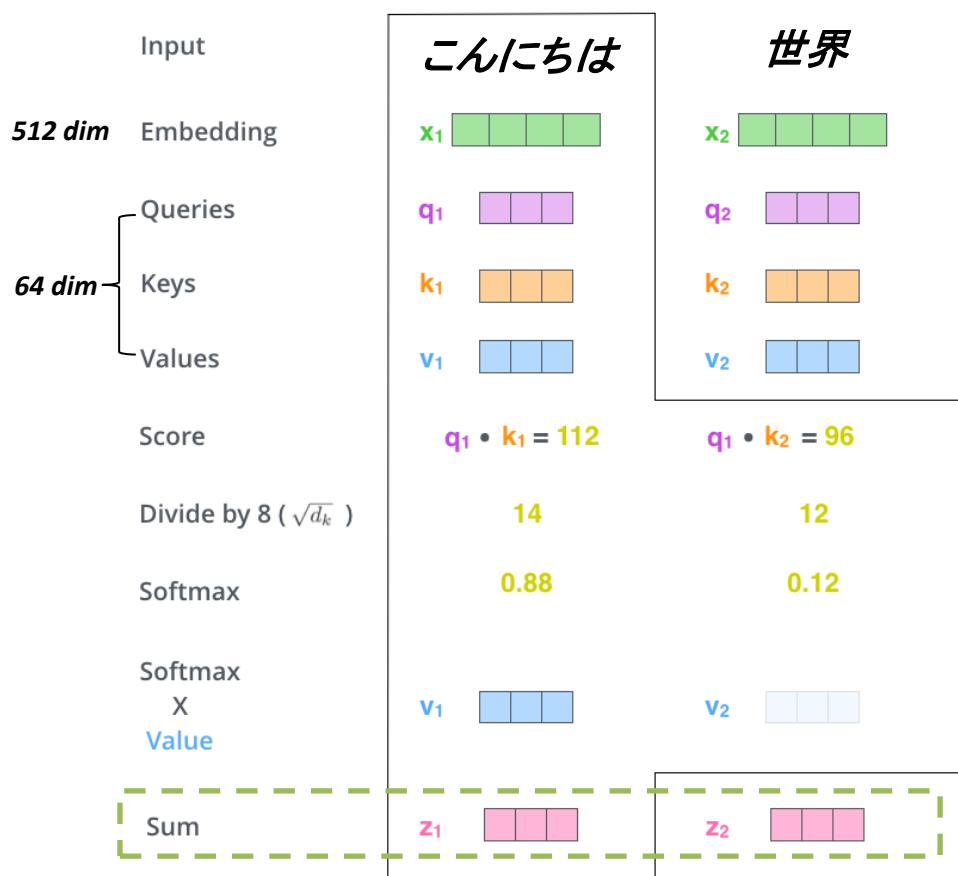


$Q=Query$, $K=Key$, $V=Value$
(64 dimension)



The transformer – model architecture

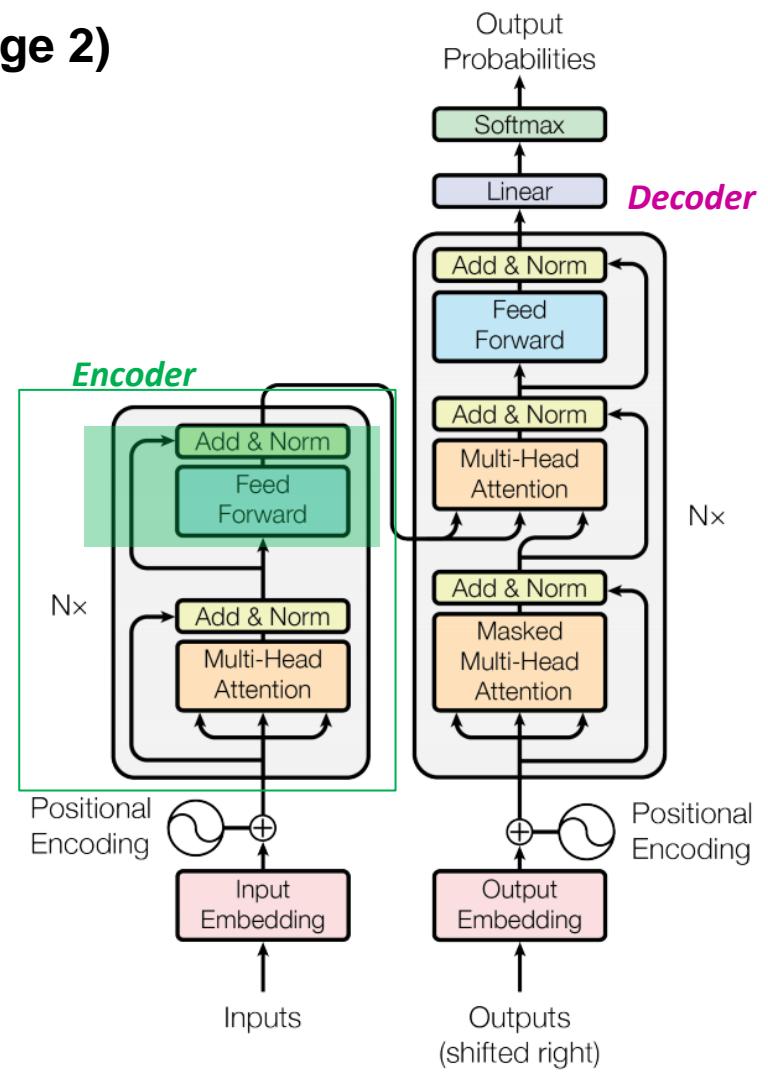
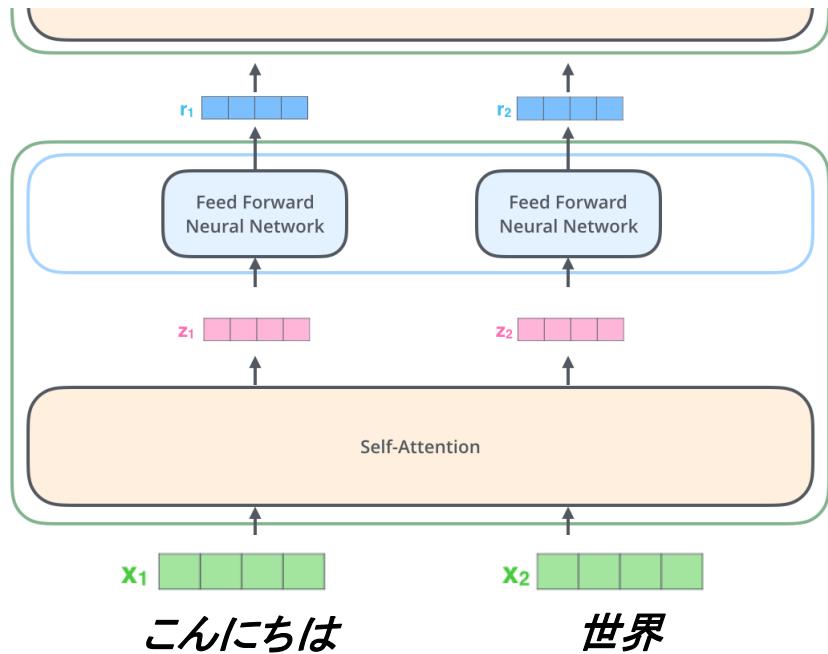
The Transformer – Encoder (Stage 2)



The transformer – model architecture

Attention and Transformer for MT

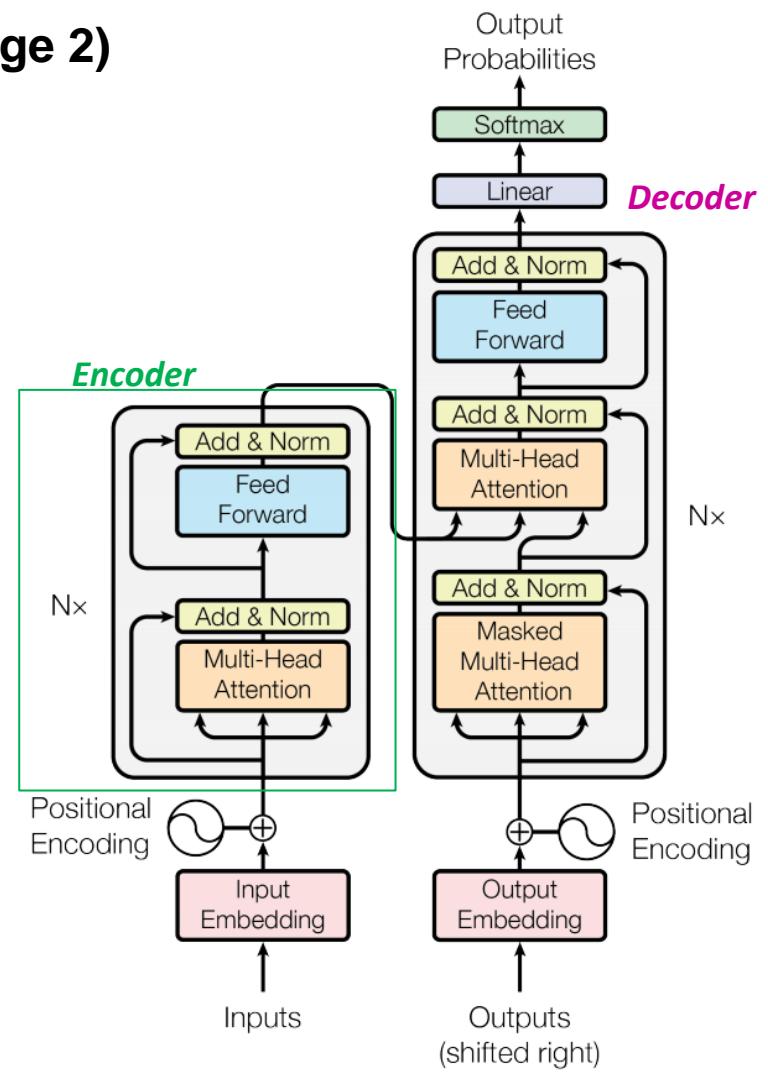
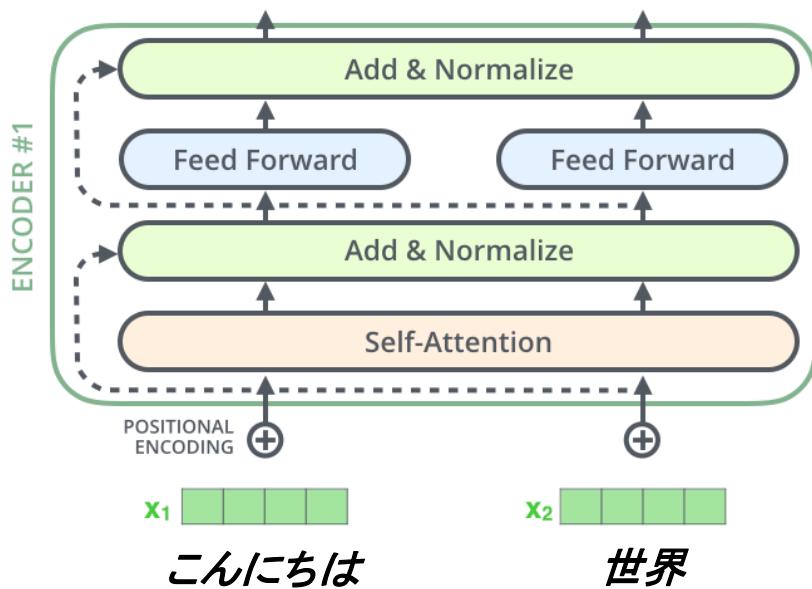
The Transformer – Encoder (Stage 2)



The transformer – model architecture

Attention and Transformer for MT

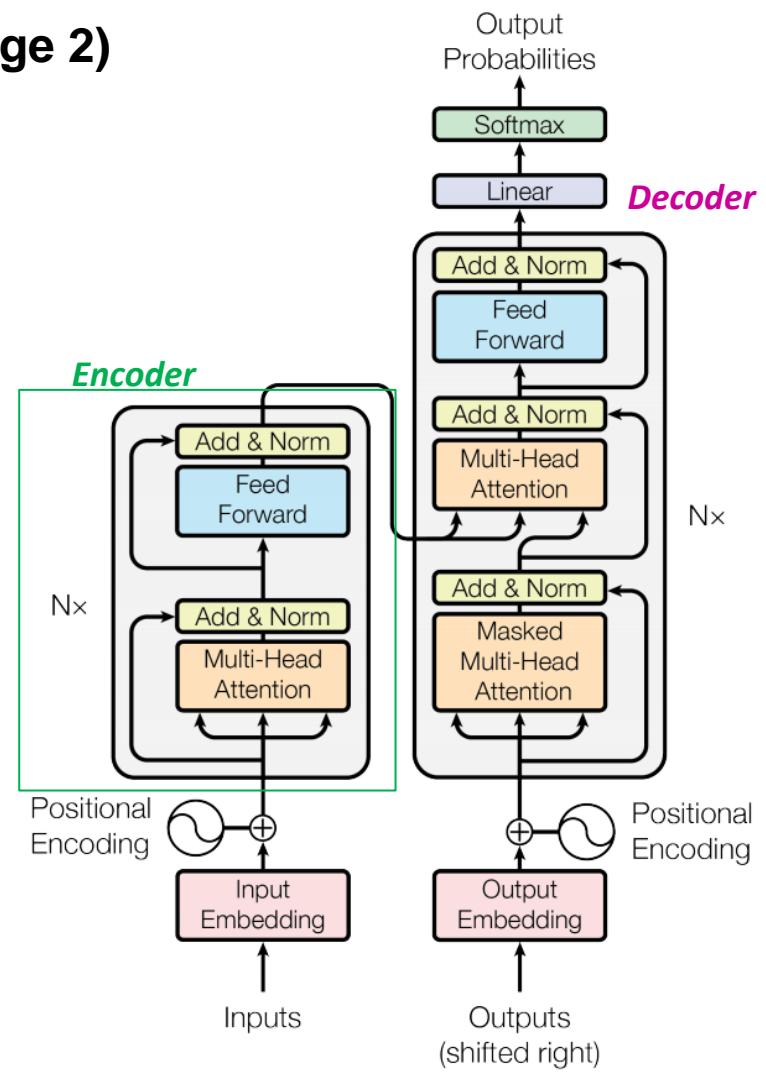
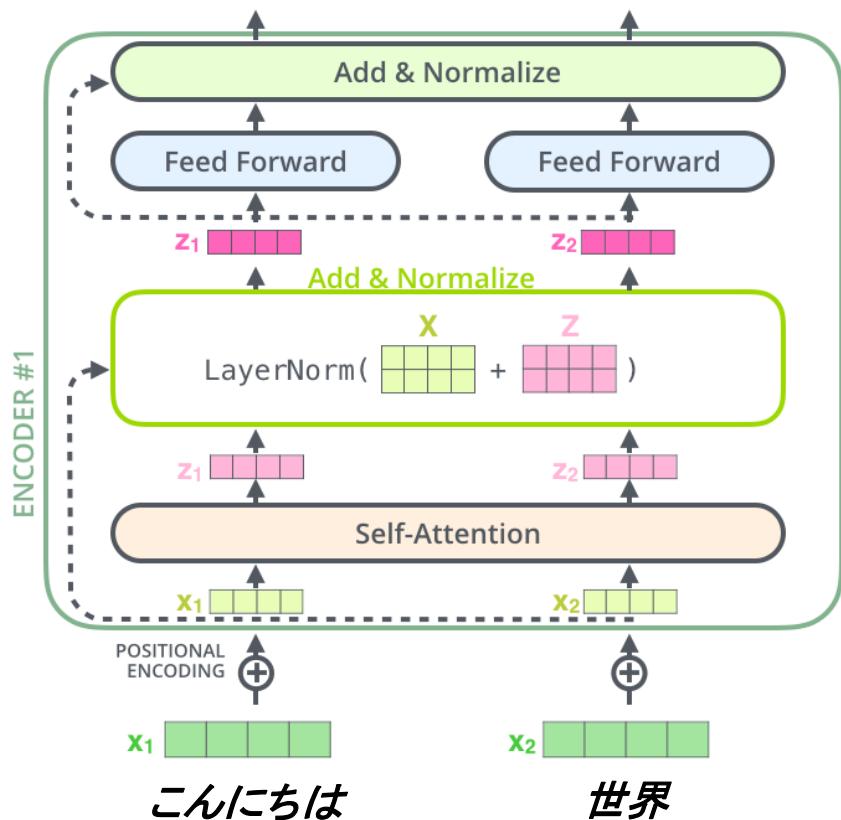
The Transformer – Encoder (Stage 2)



The transformer – model architecture

Attention and Transformer for MT

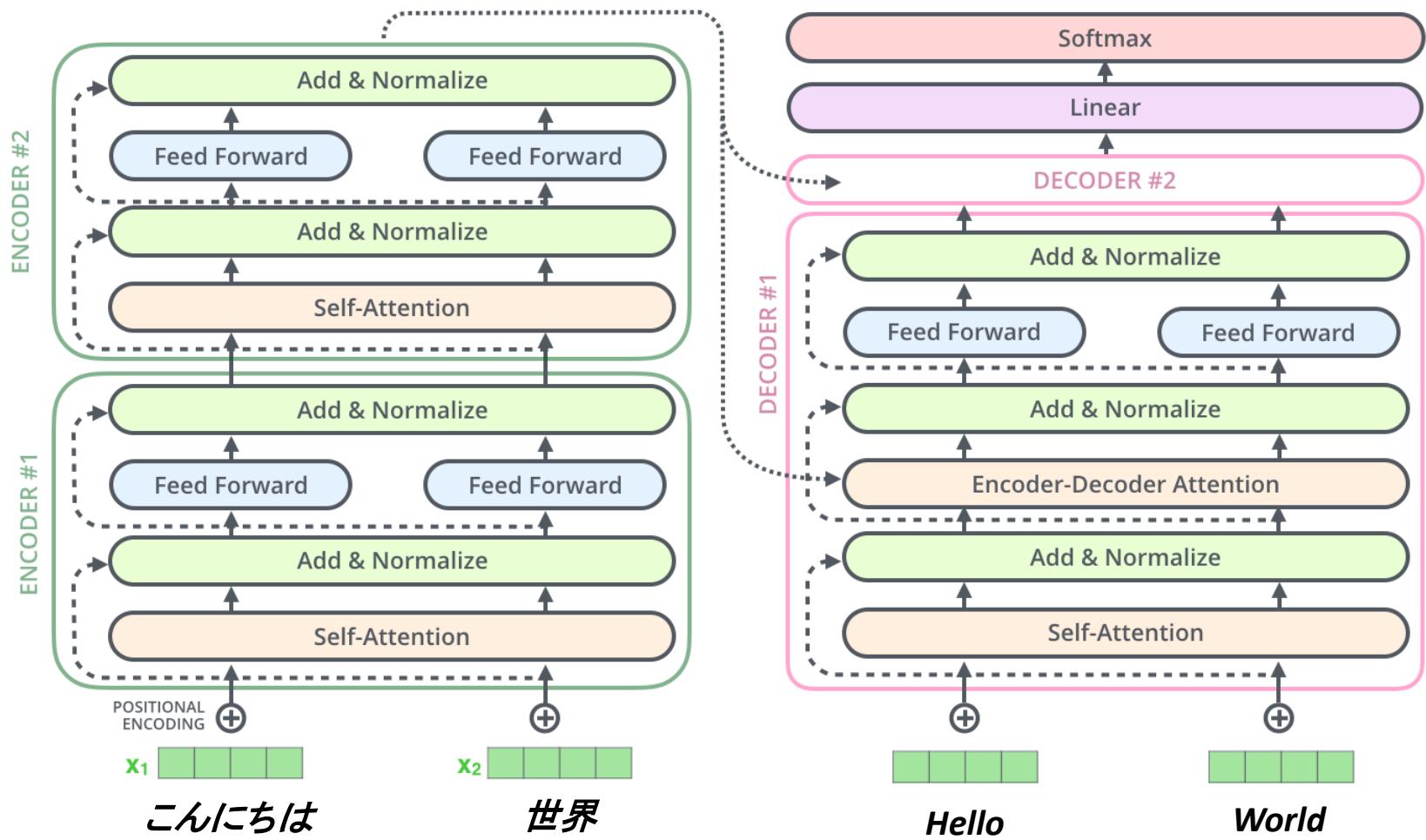
The Transformer – Encoder (Stage 2)



The transformer – model architecture

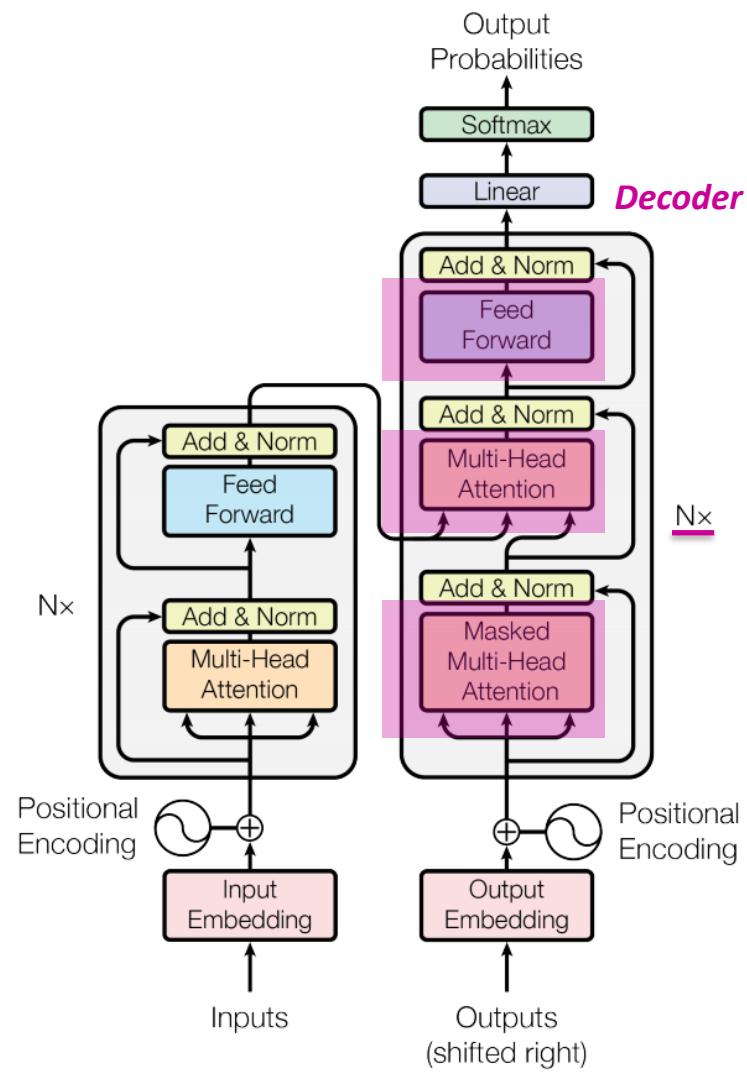
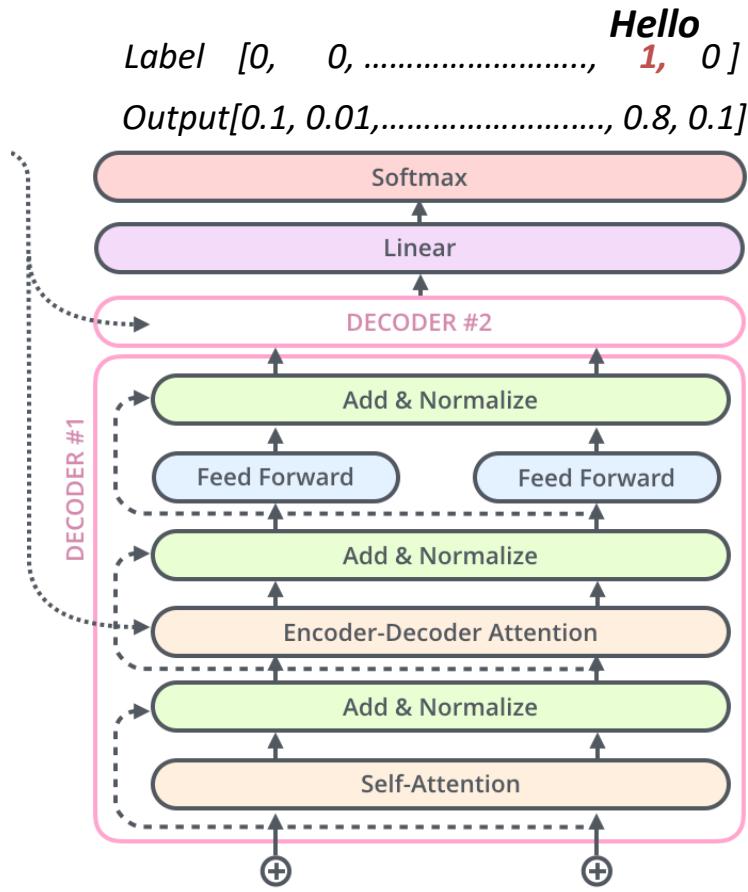
Attention and Transformer for MT

The Transformer – Encoder to Decoder



Attention and Transformer for MT

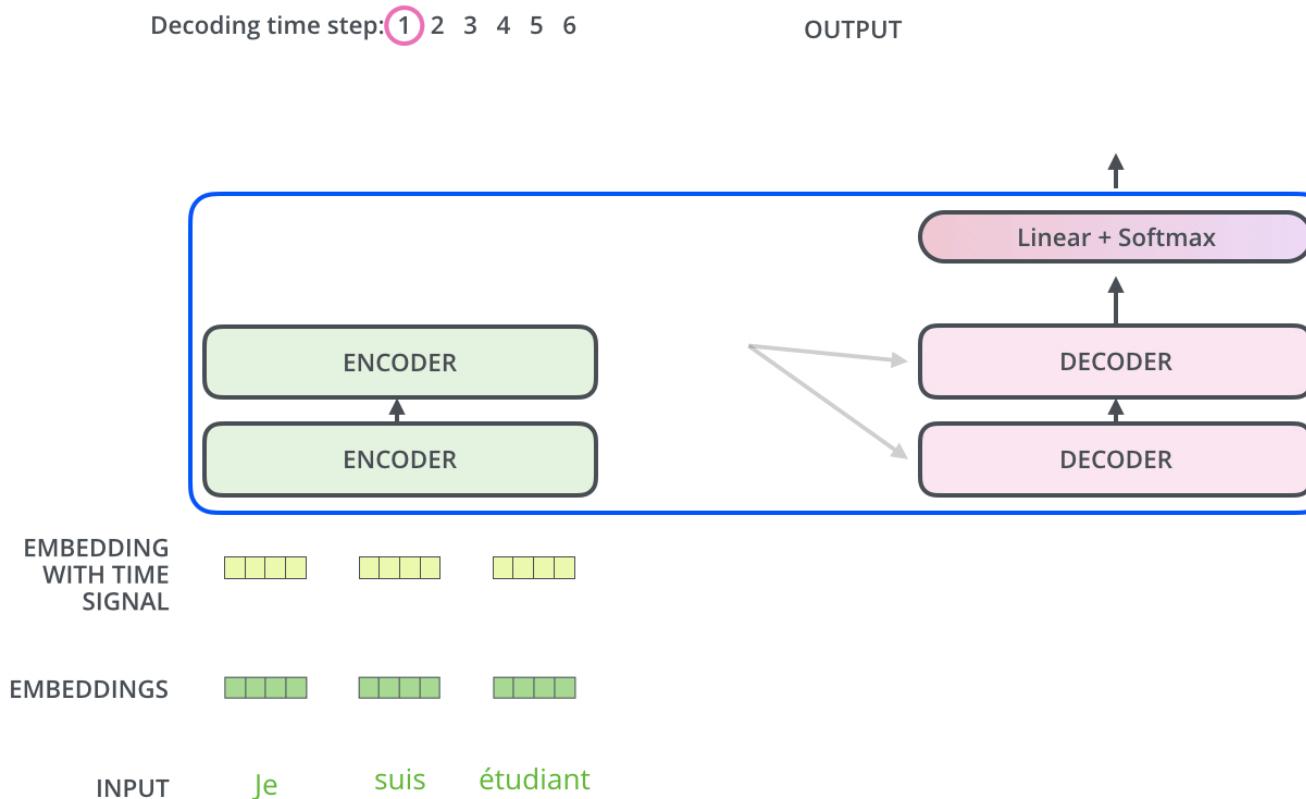
The Transformer - Decoder



The transformer – model architecture

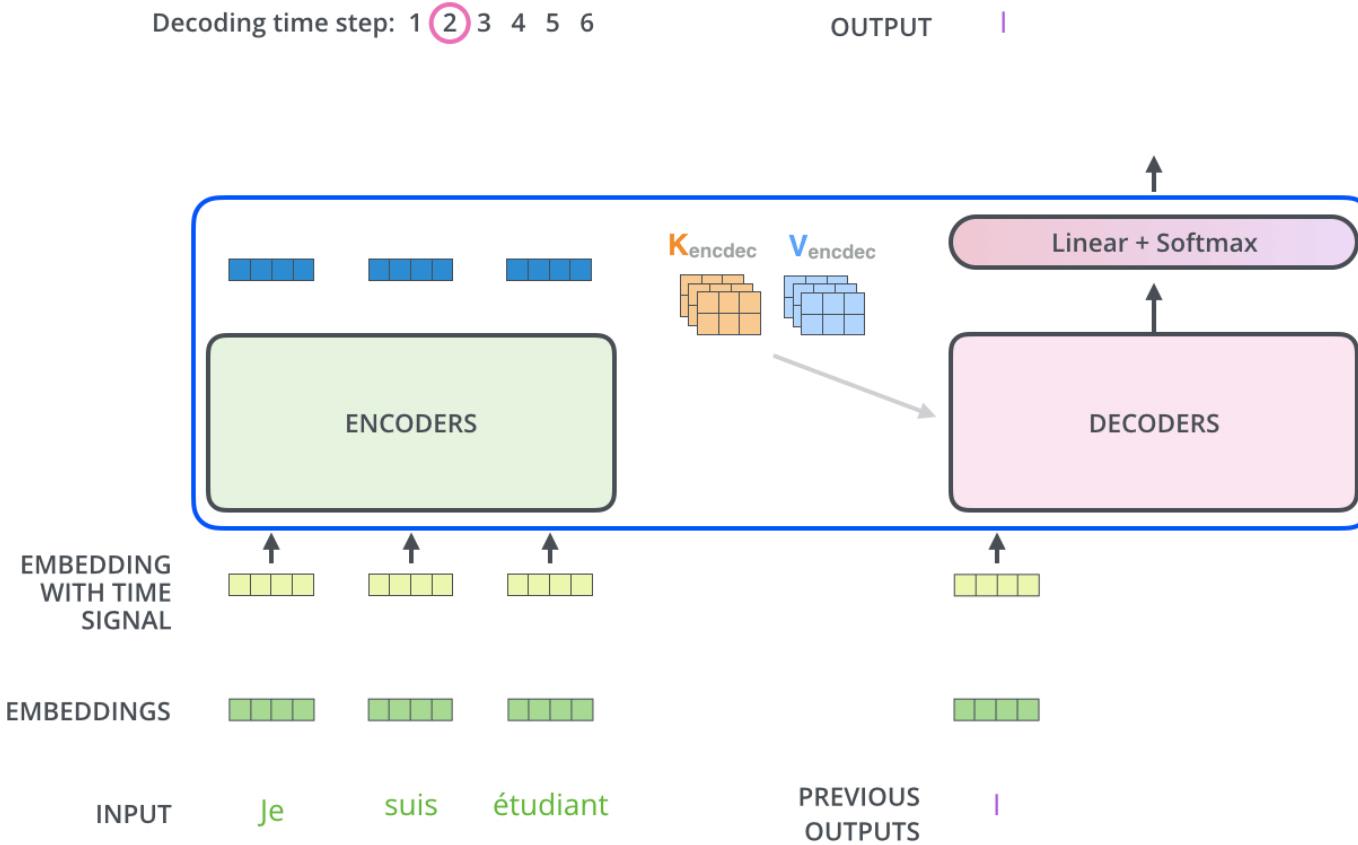
Attention and Transformer for MT

The Transformer with example – Encoder to Decoder



Attention and Transformer for MT

The Transformer with example – Decoding Phrases





5

The Rise of the Pre-trained Model

Early 2019 ~

5

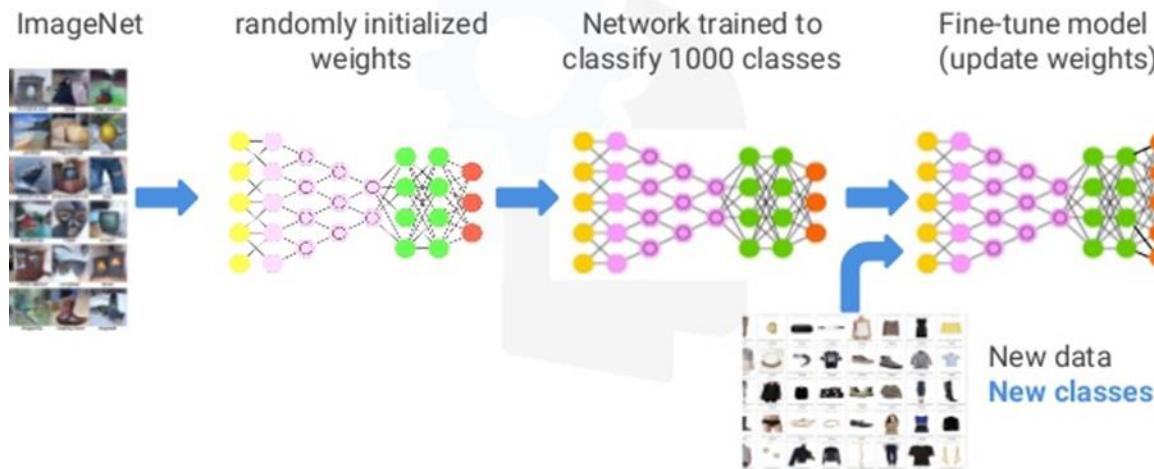
The Rise of the Pre-trained Model

Pre-training and Transfer Learning

In computer vision, prove the value of transfer learning

- pre-training a neural network on a known task (i.e. ImageNet)
- performing fine-tuning
- using the trained neural network as the basis of a new purpose-specific model.

Transfer Learning



5 The Rise of the Pre-trained Model

Pre-training and Transfer Learning in NLP

Popular Pre-trained Model in NLP

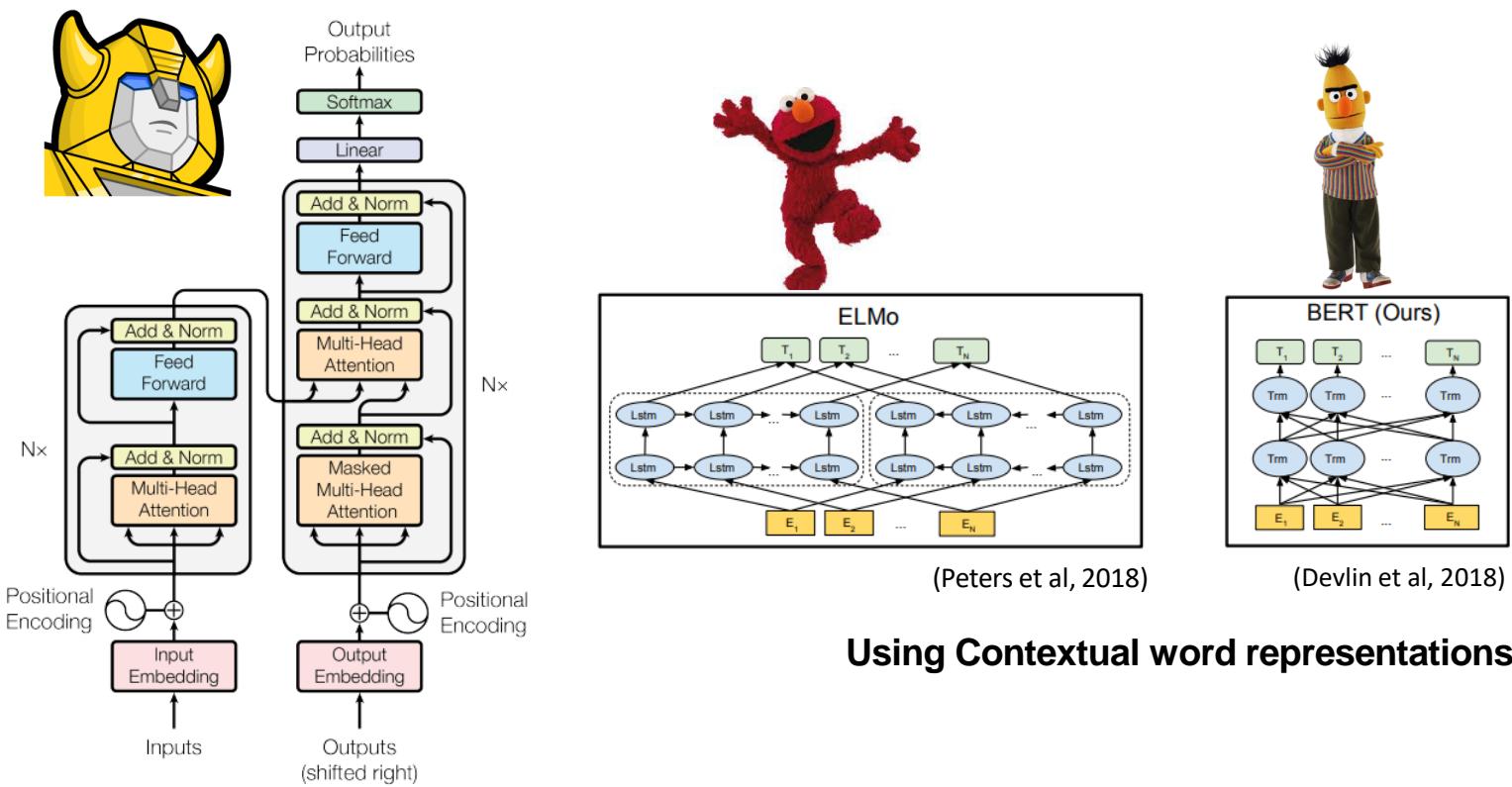


Figure 1: The Transformer - model architecture.

The Rise of the Pre-trained Model

Pre-training and Transfer Learning in NLP

Popular Pre-trained Model: Contextual Representations

Word embeddings (i.e. word2vec, fastText, GloVe) are applied in a context free manner

*Step up to the **bat** — **bat** [0.7, 0.2, -0.5, 1.1, ...]*

*A vampire **bat** — **bat** [0.7, 0.2, -0.5, 1.1, ...]*

*Need to train **contextual representation** on text corpus*

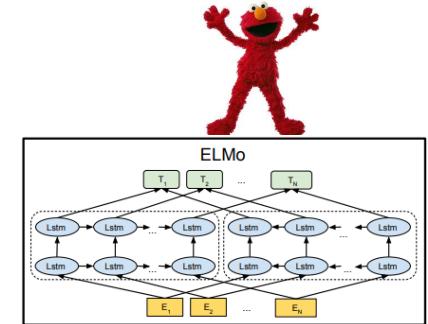
*Step up to the **bat** — **bat** [1.1, -0.7, 0.8, 2.1, ...]*

*A vampire **bat** — **bat** [0.3, 0.5, -0.9, 1.3, ...]*

The Rise of the Pre-trained Model

Pre-training and Transfer Learning in NLP

ELMo: Deep Contextual Word Embeddings (2017)



ELMo provided a significant step towards pre-training in the context of NLP. Let's dig in what the ELMo's big secret is!

5

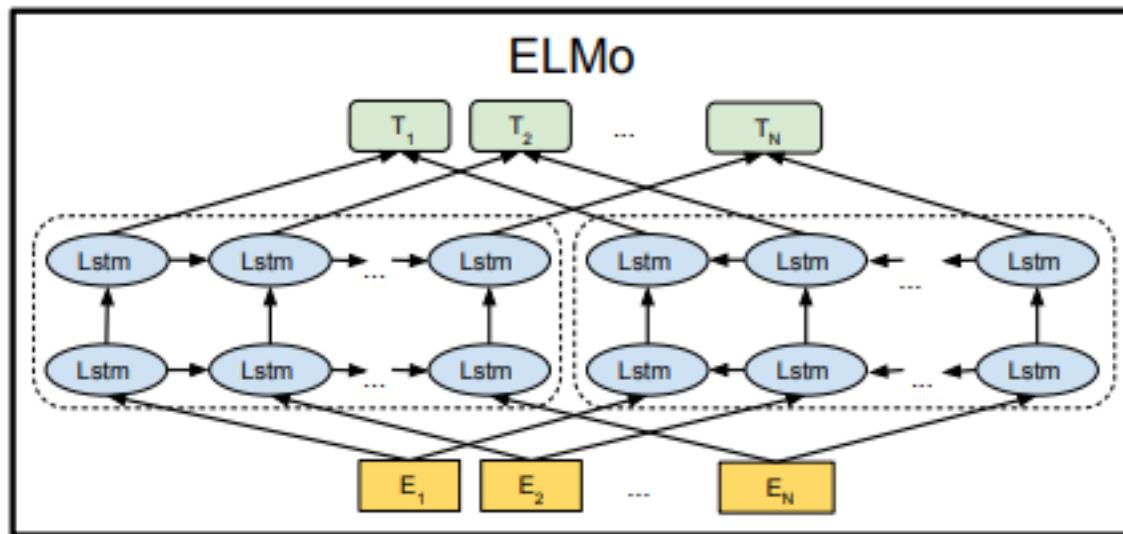
The Rise of the Pre-trained Model



Pre-training and Transfer Learning in NLP

ELMo: Deep Contextual Word Embeddings (2017)

ELMo gained its language understanding from being trained to predict the next word in a sequence of words, Language Modeling Tasks. This is convenient because we have vast amounts of text data that such a model can learn from without needing labels.

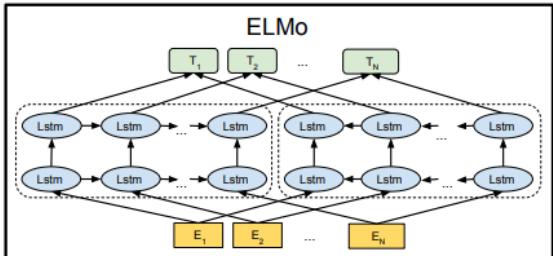


5

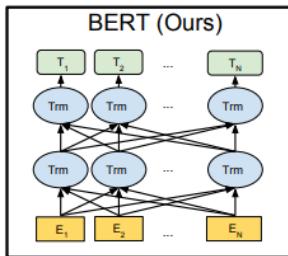
The Rise of the Pre-trained Model

Pre-training and Transfer Learning in NLP

ELMo and BERT



(Peters et al, 2018)



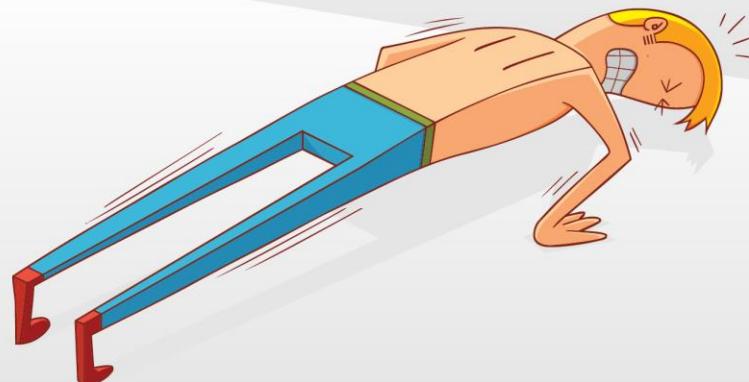
(Devlin et al, 2018)

5

The Rise of the Pre-trained Model

The future of NLP...

THE POWER OF THE PRE-TRAINING PRINCIPLE



/

COMP5046 Natural Language Processing

What we learned in this course!

Week 1: Introduction to Natural Language Processing (NLP)

Week 2: Word Embeddings (Word Vector for Meaning)

Week 3: Word Classification with Machine Learning I

Week 4: Word Classification with Machine Learning II

NLP and
Machine
Learning

Week 5: Language Fundamental

Week 6: Part of Speech Tagging

Week 7: Dependency Parsing

Week 8: Language Model

NLP
Techniques

Week 9: Information Extraction: Named Entity Recognition

Week 10: Advanced NLP: Attention and Reading Comprehension

Week 11: Advanced NLP: Transformer and Machine Translation

Week 12: Advanced NLP: Pretrained Model

Advanced
Topic

Week 13: Future of NLP and Exam Review

/ Reference

Reference for this lecture

- Deng, L., & Liu, Y. (Eds.). (2018). Deep Learning in Natural Language Processing. Springer.
- Rao, D., & McMahan, B. (2019). Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. " O'Reilly Media, Inc.".
- Manning, C. D., Manning, C. D., & Schütze, H. (1999). Foundations of statistical natural language processing. MIT press.
- Manning, C 2018, Natural Language Processing with Deep Learning, lecture notes, Stanford University
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).
- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
- Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. arXiv preprint arXiv:1802.05365.
- Miller, A., Fisch, A., Dodge, J., Karimi, A. H., Bordes, A., & Weston, J. (2016). Key-value memory networks for directly reading documents. arXiv preprint arXiv:1606.03126.
- Drawings
- <http://jalammar.github.io/illustrated-bert/>
- <http://jalammar.github.io/illustrated-transformer/>