# COMP5046
# Natural Language Processing

Lecture 3: Word Classification and Machine Learning
Semester 1, 2019
School of Computer Science
The University of Sydney, Australia

Caren Han
Caren.Han@sydney.edu.au

---

## 0 LECTURE PLAN

**Lecture 3: Word Classification and Machine Learning**

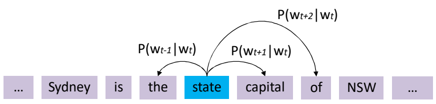
1. Word Embedding Review
2. Word Embedding Evaluation
3. Classification (Machine Learning)
4. Deep Learning for Natural Language Processing
   1. Perceptron and Neural Network (NN)
   2. Multilayer Perceptron
   3. Applications
5. Next Week Preview
   See how the Deep Learning can be used for NLP
   - Text Classification, etc.

---

## 1 Word2Vec

**Word2Vec**

**Sentence: "Sydney is the state capital of NSW"**




---

## 1 Word2Vec

**Word2Vec**

**Sentence: "Sydney is the state capital of NSW"**

*Using window sliding, develop the training data*

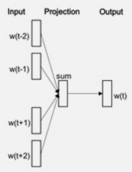
| Center word | Context ("outside") word | |
|---|---|---|
| [1,0,0,0,0,0,0] | [0,1,0,0,0,0,0], [0,0,1,0,0,0,0] | Sydney is the state capital of NSW |
| [0,1,0,0,0,0,0] | [1,0,0,0,0,0,0], [0,0,1,0,0,0,0], [0,0,0,1,0,0,0] | Sydney is the state capital of NSW |
| [0,0,1,0,0,0,0] | [1,0,0,0,0,0,0], [0,1,0,0,0,0,0] [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] | [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 |
| [0,0,0,0,1,0,0] | [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 |
| [0,0,0,0,0,1,0] | [1,0,0,0,0,0,0], [0,1,0,0,0,0,0] [0,0,0,1,0,0,0] | Sydney is the state capital of NSW |
| [0,0,0,0,0,0,1] | [1,0,0,0,0,0,0], [0,1,0,0,0,0,0] | Sydney is the state capital of NSW |

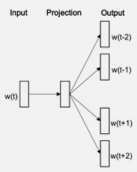
Center word
Context ("outside") word

## 1 Word2Vec

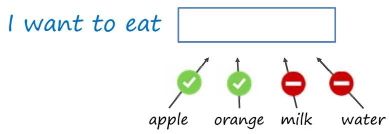
### Word2Vec Models

| CBOW | Skip-gram |
|---|---|
| *Predict center word from (bag of) context words* | *Predict context words given center word* |




## 1 Word2Vec: Demo




https://ronxin.github.io/wevi/

## 0 LECTURE PLAN

### Lecture 3: Word Classification and Machine Learning

1. Word Embedding Review
2. **Word Embedding Evaluation**
3. Classification (Machine Learning)
4. Deep Learning for Natural Language Processing
   1. Perceptron and Neural Network (NN)
   2. Multilayer Perceptron
   3. Applications
5. Next Week Preview

   See how the Deep Learning can be used for NLP
   - Text Classification, etc.

## 2 Word Embedding Evaluation

### How to evaluate word vectors?

| Type | How to work / Benefit |
|---|---|
| Intrinsic | Evaluation on a specific/intermediate subtask<br>• Fast to compute<br>• Helps to understand that system<br>• Not clear if really helpful unless correlation to real task is established |
| Extrinsic | Evaluation on a real task<br>• Can take a long time to compute accuracy<br>• Unclear if the subsystem is the problem or its interaction or other subsystems |

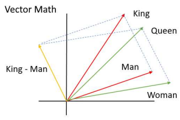
## 2 Word Embedding Evaluation

### Intrinsic word vector evaluation

**Word Vector Analogies**

*a*↔*b* :: *c*↔???
*man*↔*women* :: *king* ↔???

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




## 2 Word Embedding Evaluation

### Intrinsic word vector evaluation

**Word Vector Analogies**

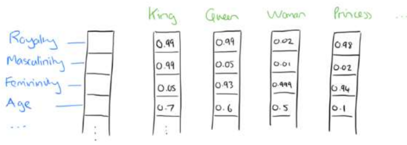
**King – Man + Woman = ?**

| No | 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

**How the word embedding can include meaning**




## 2 Word Embedding Evaluation

### Intrinsic word vector evaluation

**Evaluation Result Comparison**

The Semantic-Syntatic 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% syntatic accuracy.

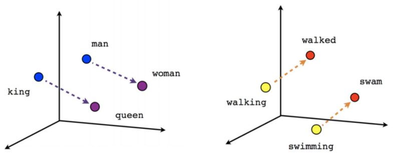
Table 3: *Comparison of architectures using models trained on the same data, with 640-dimensional word vectors. The accuracies are reported on our Semantic-Syntactic Word Relationship test set, and on the syntactic relationship test set of [20]*

| 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 |

## 2 Word Embedding Evaluation

**Intrinsic word vector evaluation**

**Word Vector Visualisation**




Male-Female

Verb tense

## 0 LECTURE PLAN

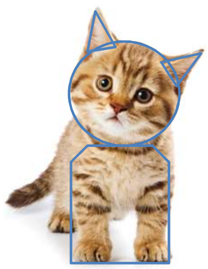
**Lecture 3: Word Classification and Machine Learning**

1. Word Embedding Review
2. Word Embedding Evaluation
3. **Classification (Machine Learning)**
4. Deep Learning for Natural Language Processing
   1. Perceptron and Neural Network (NN)
   2. Multilayer Perceptron
   3. Applications
5. Next Week Preview

   See how the Deep Learning can be used for NLP
   - Text Classification, etc.

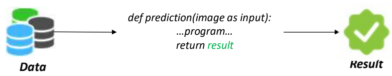
## 3 Classification (Machine Learning)

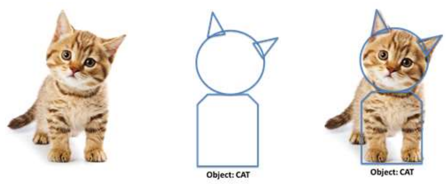
**How to classify this?**




## 3 Classification (Machine Learning)

**Computer System**



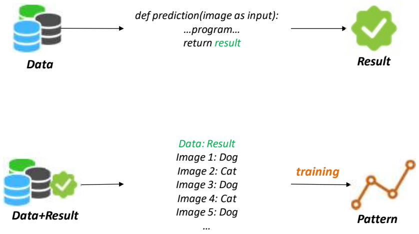
## 3 Classification (Machine Learning)

**Machine learning?**


*Object: ???* *Object: ???* *Object: ???*

## 3 Classification (Machine Learning)

**Computer System VS Machine Learning**




## 0 LECTURE PLAN

**Lecture 3: Word Embeddings and Representation**

1. Word Embedding Review
2. Word Embedding Evaluation
3. Classification (Machine Learning)
4. **Deep Learning for Natural Language Processing**
   1. Perceptron and Neural Network (NN)
   2. Multilayer Perceptron
   3. Applications
5. Next Week Preview

   See how the Deep Learning can be used for NLP
   - Text Classification, etc.

## 3 Idea of Classification

**Classification for Image Processing and NLP**


Data and Result for Natural Language Processing

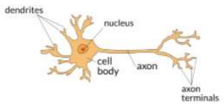
| | | |
|---|---|---|
| $x_i$ | Inputs | words (indices or vectors!), context windows, sentences, documents, etc. |
| $y_i$ | label | What we try to predict<br>• Class: word meaning, sentiment, name entity |

## 4 Deep Learning for NLP

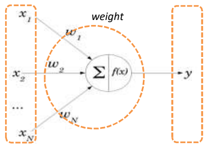
### Deep Learning with Neural Network

Neuron and Perceptron

*Neuron*




*Perceptron*



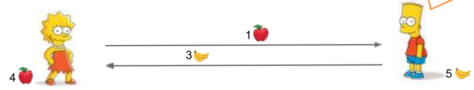

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: **Function**, Parameter, Cost, Optimiser, and Gradient

*Input: x=number of apple given by Lisa*
*Output: y=number of banana received by Lisa*

***y =3x***



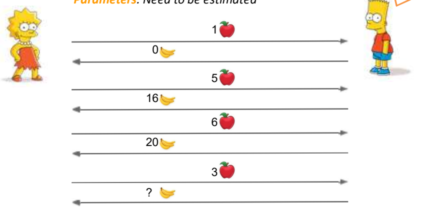

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: **Function**, Parameter, Cost, Optimiser, and Gradient

*Input: x=number of apple given by Lisa*
*Output: y=number of banana received by Lisa*
*Parameters: Need to be estimated*

***y =wx+b***



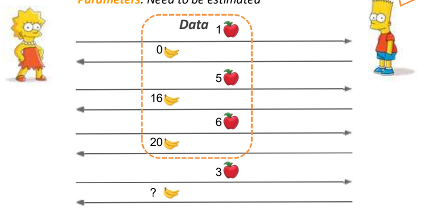

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: **Function**, Parameter, Cost, Optimiser, and Gradient

*Input: x=number of apple given by Lisa*
*Output: y=number of banana received by Lisa*
*Parameters: Need to be estimated*

*Model*
***y =wx+b***



4 **Deep Learning for NLP**

### Deep Learning with Neural Network

A Neuron: **Function**, Parameter, Cost, Optimiser, and Gradient

*Input: x=number of apple given by Lisa*
*Output: y=number of banana received by Lisa*
*Parameters: Need to be estimated*

*Model*

$y = wx + b$

*Data*

| x | ŷ |
|---|---|
| 1 | 0 |
| 5 | 16 |
| 6 | 20 |

3

?

---

4 **Deep Learning for NLP**

### Deep Learning with Neural Network

A Neuron: Function, **Parameter,** Cost, Optimiser, and Gradient

*Data*

| x | ŷ |
|---|---|
| 1 | 0 |
| 5 | 16 |
| 6 | 20 |

*Model*

$y = wx + b$

*How can we find the parameters, w and b?*

---

4 **Deep Learning for NLP**

### Deep Learning with Neural Network

A Neuron: Function, **Parameter,** Cost, Optimiser, and Gradient

*Data*

| x | ŷ |
|---|---|
| 1 | 0 |
| 5 | 16 |
| 6 | 20 |

*Model*

$y = wx + b$

*Which one is better?*

*Model Ex#1*

*w=1, b=0*

$y = 1x + 0$

| x | ŷ | y |
|---|---|---|
| 1 | 0 | 1 |
| 5 | 16 | 5 |
| 6 | 20 | 6 |

*Model Ex#2*

*w=2, b=2*

$y = 2x + 2$

| x | ŷ | y |
|---|---|---|
| 1 | 0 | 4 |
| 5 | 16 | 12 |
| 6 | 20 | 14 |

---

4 **Deep Learning for NLP**

### Deep Learning with Neural Network

A Neuron: Function, Parameter**, Cost**, Optimiser, and Gradient

*Data*

| n | x | y |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |

*Model*

$y_n = wx_n + b$

*Cost*

*Square Loss*

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

*Model Ex#1*

*w=1, b=0*

$y = 1x + 0$

| x | ŷ | y |
|---|---|---|
| 1 | 0 | 1 |
| 5 | 16 | 5 |
| 6 | 20 | 6 |

*Model Ex#2*

*w=2, b=2*

$y = 2x + 2$

| x | ŷ | y |
|---|---|---|
| 1 | 0 | 4 |
| 5 | 16 | 12 |
| 6 | 20 | 14 |

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, **Cost**, Optimiser, and Gradient

***Data***

| n | x | y |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |

***Model***

$y_n = wx_n + b$

***Cost***

***Square Loss***

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

*Model Ex#1*

*w=1, b=0*

$y = 1x + 0$

| n | x | ŷ | y | (y-ŷ)² |
|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 1 |
| 1 | 5 | 16 | 5 | 121 |
| 2 | 6 | 20 | 6 | 196 |
| | | | C(1,0) | 318 |

*Model Ex#2*

*w=2, b=2*

$y = 2x + 2$

| n | x | ŷ | y | (y-ŷ)² |
|---|---|---|---|---|
| 0 | 1 | 0 | 4 | 16 |
| 1 | 5 | 16 | 12 | 16 |
| 2 | 6 | 20 | 14 | 36 |
| | | | C(2,2) | 68 |

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, **Cost**, Optimiser, and Gradient

***Data***

| n | x | y |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |

***Model***

$y_n = wx_n + b$

***Cost***

***Square Loss***

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

*Model Ex#1*

*w=1, b=0*

$y = 1x + 0$

| n | x | ŷ | y | (y-ŷ)² |
|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 1 |
| 1 | 5 | 16 | 5 | 121 |
| 2 | 6 | 20 | 6 | 196 |
| | | | C(1,0) | 318 |

*Model Ex#2*

*w=2, b=2*

$y = 2x + 2$

| n | x | ŷ | y | (y-ŷ)² |
|---|---|---|---|---|
| 0 | 1 | 0 | 4 | 16 |
| 1 | 5 | 16 | 12 | 16 |
| 2 | 6 | 20 | 14 | 36 |
| | | | C(2,2) | 68 |

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, **Optimiser**, and Gradient

***Data***

| n | x | y |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |

***Model***

$y_n = wx_n + b$

***Cost***

***Square Loss***

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

***Optimiser***

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

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, **Optimiser**, and Gradient

***Data***

| n | x | y |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |

***Model***

$y_n = wx_n + b$

***Cost***

***Square Loss***

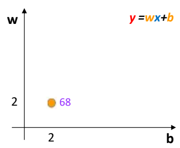
$C(w,b) = \sum_{n\in\{0,1,2\}} (y_n - \hat{y}_n)$

***Optimiser***

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

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

$w_0,b_0 = 2,2 : C(w_0,b_0) = 68$



## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, **Optimiser**, and Gradient

***Data***

| n | x | y |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |

***Model***

$y_n = wx_n + b$

***Cost*** (Square Loss)

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

***Optimiser***

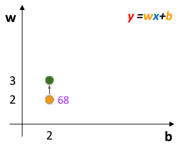
$\arg\min_{w,b\in[-\infty,\infty]} C(w,b)$

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

$w_0,b_0 = 2,2 : C(w_0,b_0) = 68$

$w_1,b_1 = 3,2 : C(w_1,b_1) = ??$

| n | x | ŷ | y | $(y-\hat{y})^2$ |
|---|---|---|---|---|
| 0 | 1 | 0 | 5 | 25 |
| 1 | 5 | 16 | 17 | 1 |
| 2 | 6 | 20 | 20 | 0 |
| | | | C(3,2) | 26 |




## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, **Optimiser**, and Gradient

***Data***

| n | x | y |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |

***Model***

$y_n = wx_n + b$

***Cost*** (Square Loss)

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

***Optimiser***

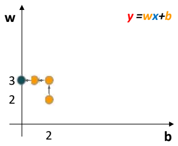
$\arg\min_{w,b\in[-\infty,\infty]} C(w,b)$

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

$w_1,b_1 = 3,2 : C(w_1,b_1) = 26$

$w_3,b_3 = 3,0 : C(w_3,b_3) = 13$

| n | x | ŷ | y | $(y-\hat{y})^2$ |
|---|---|---|---|---|
| 0 | 1 | 0 | 3 | 9 |
| 1 | 5 | 16 | 15 | 1 |
| 2 | 6 | 20 | 18 | 4 |
| | | | C(3,0) | 13 |




## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, **Optimiser**, and Gradient

***Data***

| n | x | y |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |

***Model***

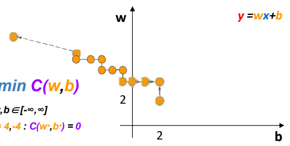
$y_n = wx_n + b$

***Cost*** (Square Loss)

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

***Optimiser***

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




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

$w^*,b^* = 4,-4 : C(w^*,b^*) = 0$

$y = 4x - 4$

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, **Optimiser**, and Gradient

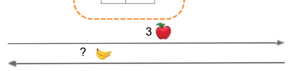
*Input*: x=number of apple given by Lisa
*Output*: y=number of banana received by Lisa
*Parameters*: Need to be estimated

***Model***

$y = 4x - 4$

***Data***

| x | ŷ |
|---|---|
| 1 | 0 |
| 5 | 16 |
| 6 | 20 |



## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, **Optimiser**, and Gradient

$$y = \underset{pixel(1,1)}{w_1 x_1} + \underset{pixel(1,2)}{w_2 x_2} + w_3 x_3 + w_4 x_4 + \ldots + w_n x_n + b$$

*Data*

*Millions of **Parameters***
*Millions of **Samples***

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, **Optimiser**, and Gradient

$$y = \underset{vector1}{w_1 x_1} + \underset{vector2}{w_2 x_2} + w_3 x_3 + w_4 x_4 + \ldots + w_n x_n + b$$

*Data*

*Millions of **Parameters***
*Millions of **Samples***

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, Optimiser, and **Gradient**

***Data***

| n | x | y |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |

***Model***

$$y_n = wx_n + b$$

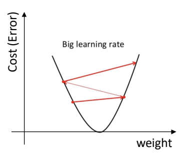
***Cost***

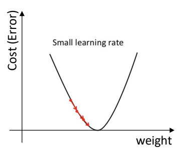
Square Loss

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

***Optimiser***

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




*new_weight = existing_weight — learning_rate * gradient*

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

Parameters vs Hyper-Parameters

1. Parameters:
   - Parameters are the tunable components of the model.
   - Learnt from training data.
   - Typical examples: probabilities, feature weights
2. Hyper-parameters:
   - Variables that control how parameters are learnt.
   - Chosen a priori (or tuned using held-out data).
   - Typical examples: model size (depth, complexity), learning rate

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, Optimiser, and **Gradient**

***Data***

| n | x | y |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |

***Model***

$y_n = wx_n + b$

***Cost***

Square Loss

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

***Optimiser***

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

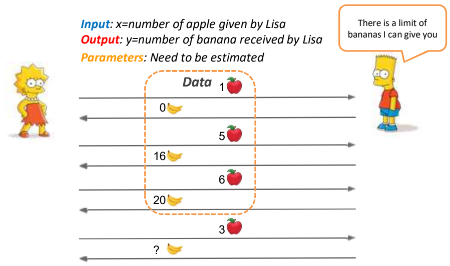
*System*

$y = 4x - 4$

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

A Neuron: Function, Parameter, Cost, Optimiser, and Gradient



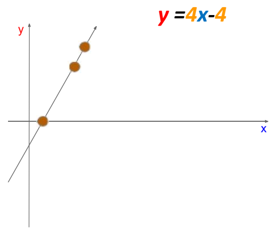

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

Nonlinear Neural Network

***Data***

| x | ŷ |
|---|---|
| 1 | 0 |
| 5 | 16 |
| 6 | 20 |



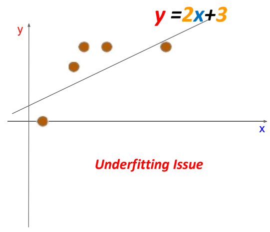

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

Nonlinear Neural Network

***Data***

| n | x | ŷ |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |
| 3 | 9 | 20 |
| 4 | 11 | 20 |



## 4 Deep Learning for NLP

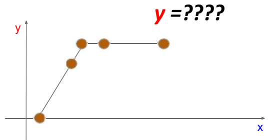
### Deep Learning with Neural Network

Nonlinear Neural Network

*Data*

| n | x | ŷ |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |
| 3 | 9 | 20 |
| 4 | 11 | 20 |

***y =????***




***How to make this possible?***

***A: Use different linear functions depending on the value of x?***

$y = (w_1x + b_1)s_1 + (w_2x+b_2)s_2$

## 4 Deep Learning for NLP

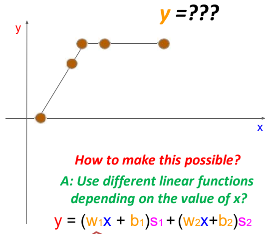
### Deep Learning with Neural Network

Nonlinear Neural Network

*Data*

| n | x | ŷ |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |
| 3 | 9 | 20 |
| 4 | 11 | 20 |

***y =???***




***How to make this possible?***

***A: Use different linear functions depending on the value of x?***

$y = (w_1x + b_1)s_1 + (w_2x+b_2)s_2$

**S1** - 1 if x < 6 and 0 otherwise
**S2** - 1 if x >= 6 and 0 otherwise

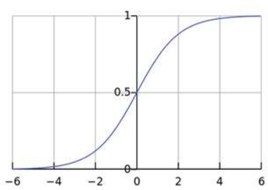
## 4 Deep Learning for NLP

### Deep Learning with Neural Network

Nonlinear Neural Network

$$s = o(wx + b)$$

$$o(t) = \frac{1}{1 + e^{-t}}$$




## 4 Deep Learning for NLP

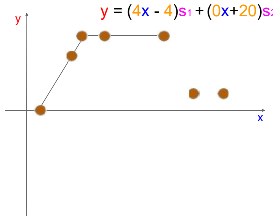
### Deep Learning with Neural Network

Nonlinear Neural Network

*Data*

| n | x | ŷ |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |
| 3 | 9 | 20 |
| 4 | 11 | 20 |

$y = (4x - 4)s_1 + (0x+20)s_2$



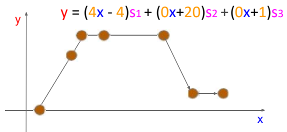
## 4 Deep Learning for NLP

### Deep Learning with Neural Network

Multilayer Perceptron

*Data*

| n | x | ŷ |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |
| 3 | 9 | 20 |
| 4 | 11 | 20 |
| 5 | 15 | 1 |
| 6 | 19 | 1 |




$y = (w_1x + b_1)s_1 + (w_2x+b_2)s_2 + (w_3x+b_3)s_3$

$s_1 = \sigma(w_4x + b_4)$ *Layer 1*

$s_2 = \sigma(w_5s_1 + w_6s_3 + b_5)$ *Layer 2*

$s_3 = \sigma(w_7x + b_6)$ *Layer 1*

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

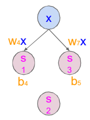
Multilayer Perceptron

$y = (w_1x + b_1)s_1 + (w_2x+b_2)s_2 + (w_3x+b_3)s_3$

$s_1 = \sigma(w_4x + b_4)$

$s_2 = \sigma(w_5s_1 + w_6s_3 + b_5)$

$s_3 = \sigma(w_7x + b_6)$




*Layer 1*

*Layer 2*

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

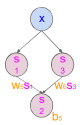
Multilayer Perceptron

$y = (w_1x + b_1)s_1 + (w_2x+b_2)s_2 + (w_3x+b_3)s_3$

$s_1 = \sigma(w_4x + b_4)$

$s_2 = \sigma(w_5s_1 + w_6s_3 + b_5)$

$s_3 = \sigma(w_7x + b_6)$




*Layer 1*

*Layer 2*

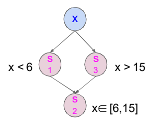
## 4 Deep Learning for NLP

### Deep Learning with Neural Network

Multilayer Perceptron

$y = (w_1x + b_1)s_1 + (w_2x+b_2)s_2 + (w_3x+b_3)s_3$

| n | x | ŷ |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |
| 3 | 9 | 20 |
| 4 | 11 | 20 |
| 5 | 15 | 1 |
| 6 | 19 | 1 |




*Layer 1*

*Layer 2*

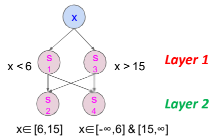
## 4 Deep Learning for NLP

### Deep Learning with Neural Network

Multilayer Perceptron

$y = (w_1x + b_1)s_1 + (w_2x+b_2)s_2 + (w_3x+b_3)s_3$

| n | x | ŷ |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |
| 3 | 9 | 20 |
| 4 | 11 | 20 |
| 5 | 15 | 1 |
| 6 | 19 | 1 |

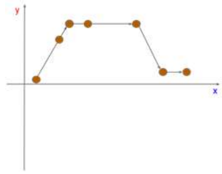


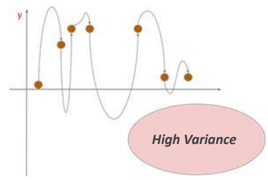

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

Multilayer Perceptron

$y = (w_1x + b_1)s_1 + (w_2x+b_2)s_2 + (w_3x+b_3)s_3$



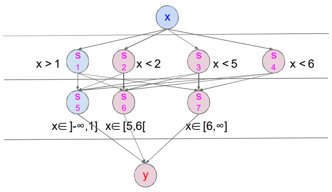

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

Multilayer Perceptron

*Data*

| n | x | ŷ |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |



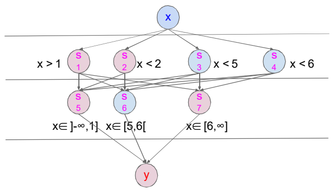

## 4 Deep Learning for NLP

### Deep Learning with Neural Network

Multilayer Perceptron

*Data*

| n | x | ŷ |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |



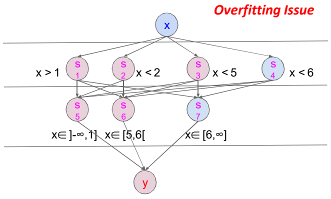
## 4 Deep Learning for NLP

### Deep Learning with Neural Network

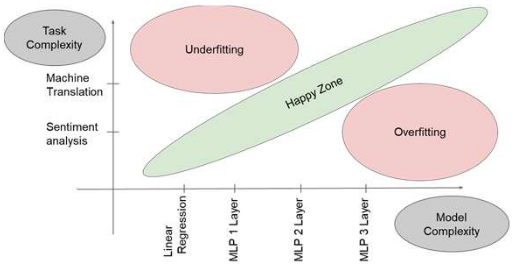
Multilayer Perceptron

*Data*

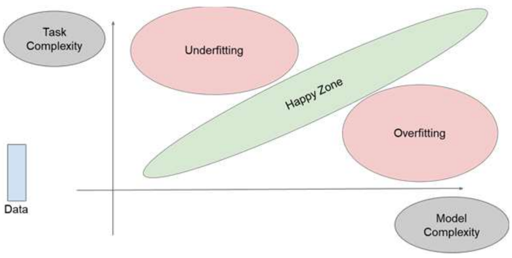
| n | x | ŷ |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 5 | 16 |
| 2 | 6 | 20 |



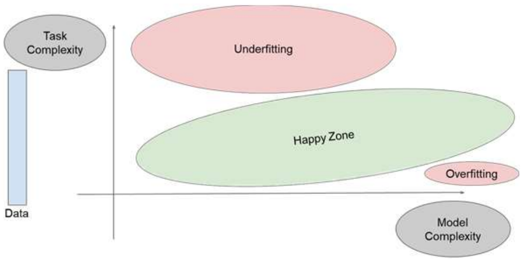

## 4 Deep Learning for NLP



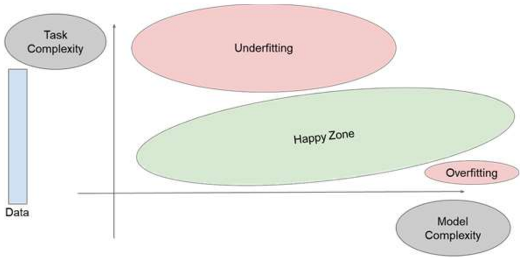

## 4 Deep Learning for NLP




## 4 Deep Learning for NLP



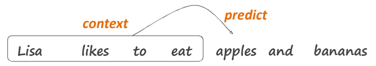
## 4 Deep Learning for NLP




## 4 Deep Learning for NLP

**Deep Learning with NLP**

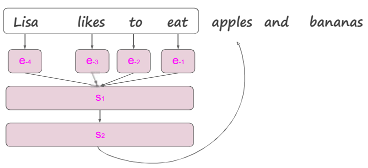
Application #1: Embedding Pretraining




## 4 Deep Learning for NLP

**Deep Learning with NLP**

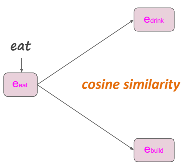
Application #1: Embedding Pretraining




## 4 Deep Learning for NLP

**Deep Learning with NLP**

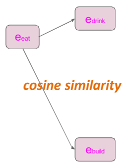
Application #1: Embedding Pretraining



## 4 Deep Learning for NLP

### Deep Learning with NLP

Application #1: Embedding Pretraining




http://bionlp-www.utu.fi/wv_demo/

## 4 Deep Learning for NLP

### Deep Learning with NLP

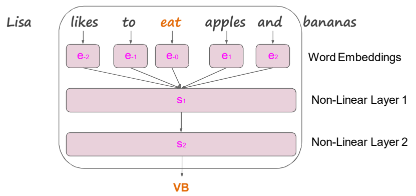
Application #2: Window-based Tagging

| Lisa | likes | to | eat | apples | and | bananas |
|---|---|---|---|---|---|---|
| NNP | VBZ | TO | VB | NNS | CC | NNS |

## 4 Deep Learning for NLP

### Deep Learning with NLP

Application #2: Window-based Tagging




## 4 Deep Learning for NLP

### Deep Learning with NLP

Application #3: Translation Rescoring

*Lisa gosta de comer Macas e bananas* source

*Bart does to eat coconuts and bananas* *translation#1*

*Lisa likes to eat apples and bananas* *translation#2*

*Lisa dislikes to drink apples and bananas* *translation#3*

## 4 Deep Learning for NLP

### Deep Learning with NLP

Application #3: Translation Rescoring

*Lisa gosta de comer Macas e bananas* source

*Lisa likes to eat apples and bananas* translation#2

0.2 0.1

## 4 Deep Learning for NLP

### Deep Learning with NLP

Application #3: Translation Rescoring

*Lisa gosta de comer Macas e bananas* source

*Lisa likes to eat apples and bananas* translation#2

0.2 0.1 0.3

## 4 Deep Learning for NLP

### Deep Learning with NLP

Application #3: Translation Rescoring

*Lisa gosta de comer Macas e bananas* source

*Lisa likes to eat apples and bananas* translation#2

0.2 0.1 0.3

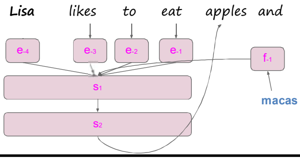
## 4 Deep Learning for NLP

### Deep Learning with NLP

Application #3: Translation Rescoring

*Lisa gosta de comer macas e bananas* source

*Lisa likes to eat apples and bananas* translation#2



## 4 Deep Learning for NLP

### Deep Learning with NLP

Application #3: Translation Rescoring

| | |
|---|---|
| Lisa gosta de corner Macas e bananas | source |
| Bart does to eat coconuts and bananas | 0.00003 |
| Lisa likes to eat apples and bananas | 0.000378 |
| Lisa dislikes to drink apples and bananas | 0.00012 |

## / 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