COMP5046
Natural Language Processing

Lecture 3: Word Classification and Machine Learning

Semester 1, 2020

School of Computer Science

The University of Sydney, Australia

THE UNIVERSITY OF SYDNEY

Dr Caren Han

Caren.Han@sydney.edu.au

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

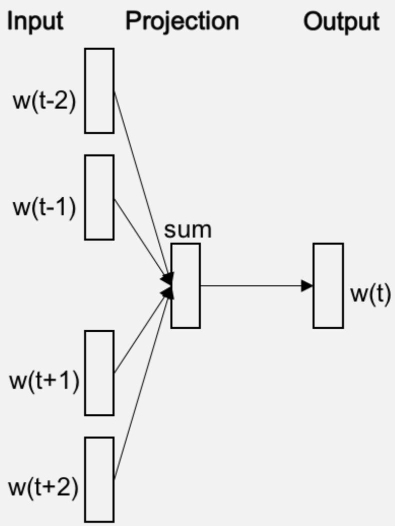
    See how the Deep Learning can be used for NLP

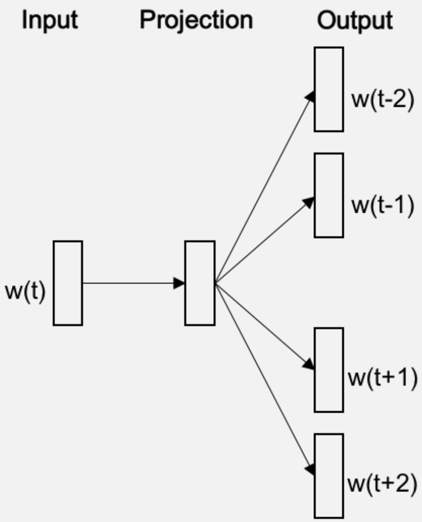
    - Text Classification, etc.

1 Previous Lecture Review

Word2Vec Models

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



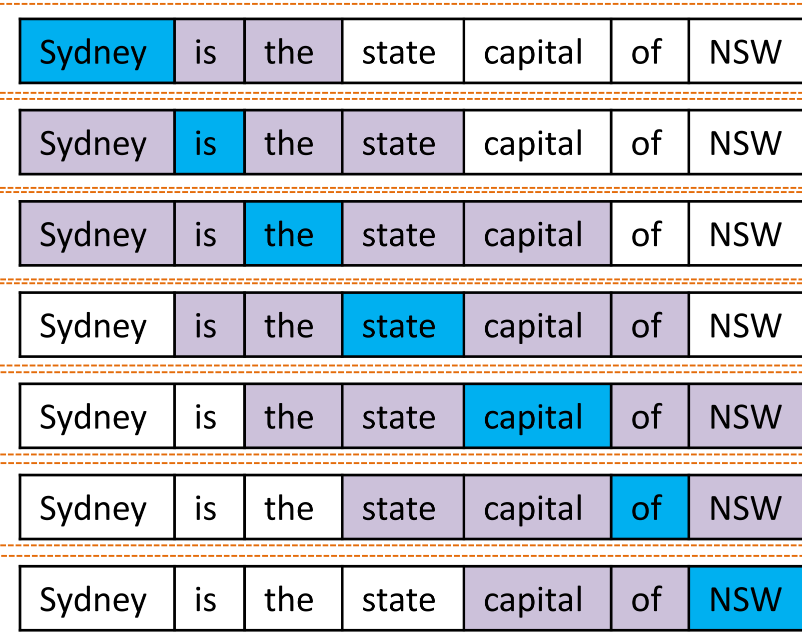


Word2Vec

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]
[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]
[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]
[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]
[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]
[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]
[0,0,0,0,0,0,1]	[0,0,0,0,1,0,0], [0,0,0,0,0,1,0]



Center word

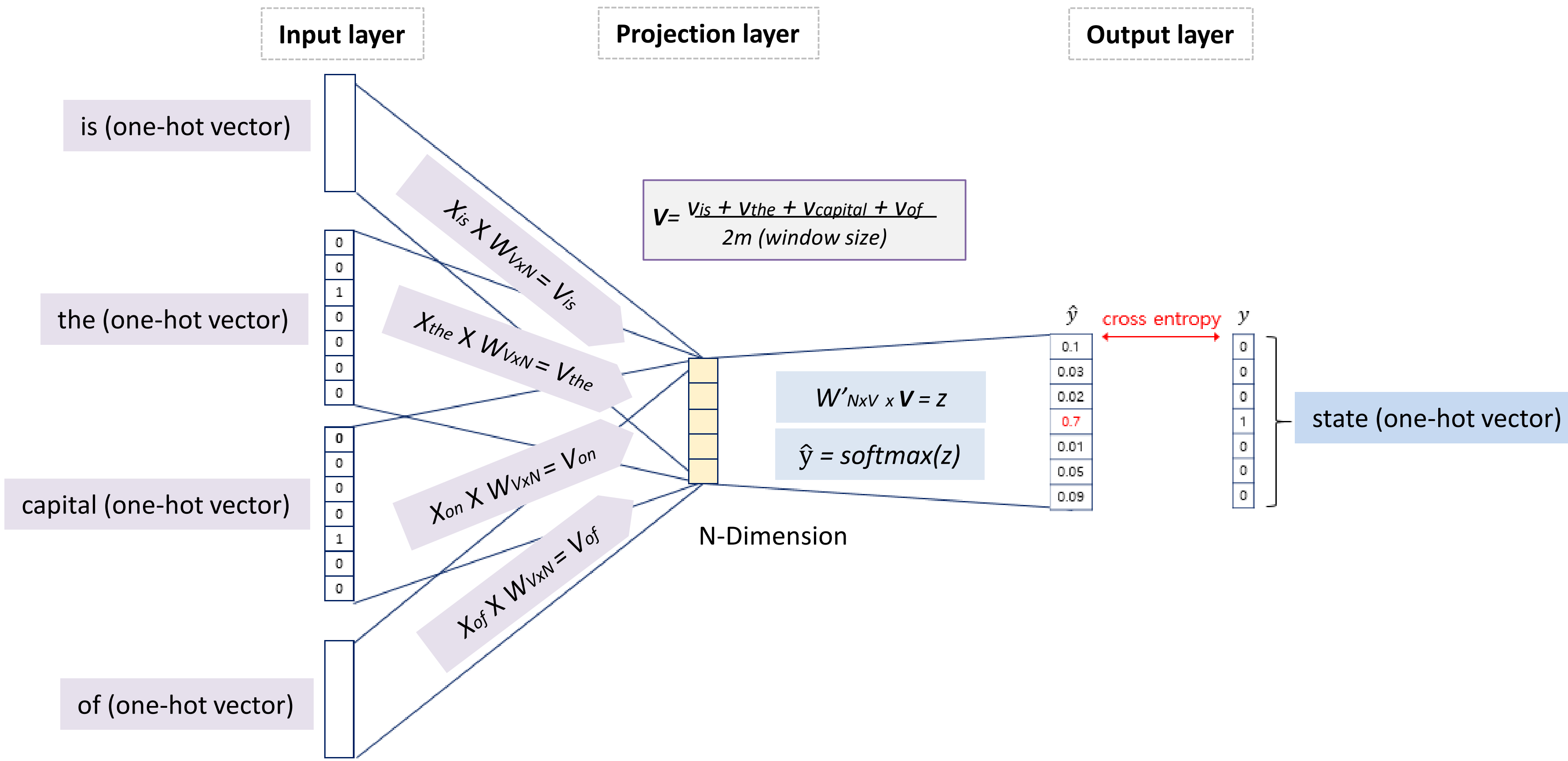
Context ("outside") word

Previous Lecture Review

CBOW Process

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.

2 Word Embedding Evaluation

How to evaluate word vectors?

Type	How to work / Benefit
Intrinsic	Evaluation on a specific/intermediate subtask • 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 • Can take a long time to compute accuracy • Unclear if the subsystem is the problem or its interaction or other subsystems

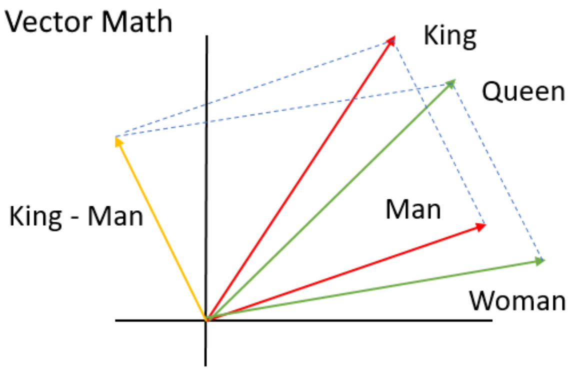
Word Embedding Evaluation

Intrinsic word vector evaluation

Word Vector Analogies

$$a \leftrightarrow b \quad :: \quad c \leftrightarrow ???$$
$$man \leftrightarrow women :: king \leftrightarrow ???$$

- 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

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]*

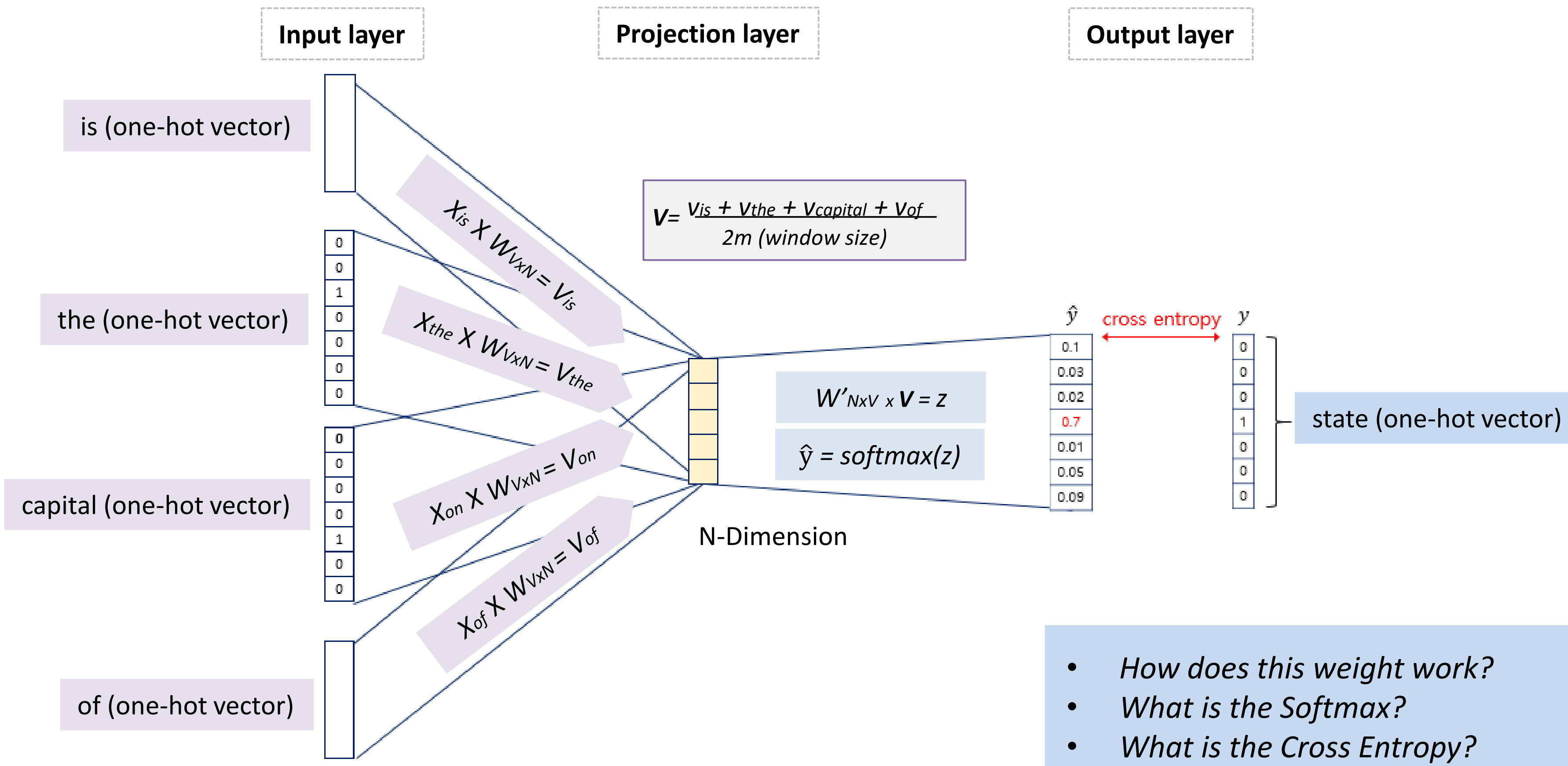
Model Architecture	Semantic-Syntactic Word Relationship test set		MSR Word Relatedness Test Set [20]
	Semantic Accuracy [%]	Syntactic Accuracy [%]	
RNNLM	9	36	35
NNLM	23	53	47
CBOW	24	64	61
Skip-gram	55	59	56

Previous Lecture Review

CBOW Process

Predict center word from (bag of) context words

Sentence: "Sydney is the state capital of NSW"



- *How does this weight work?*
- *What is the Softmax?*
- *What is the Cross Entropy?*

0 LECTURE PLAN

Lecture 3: Word Classification and Machine Learning

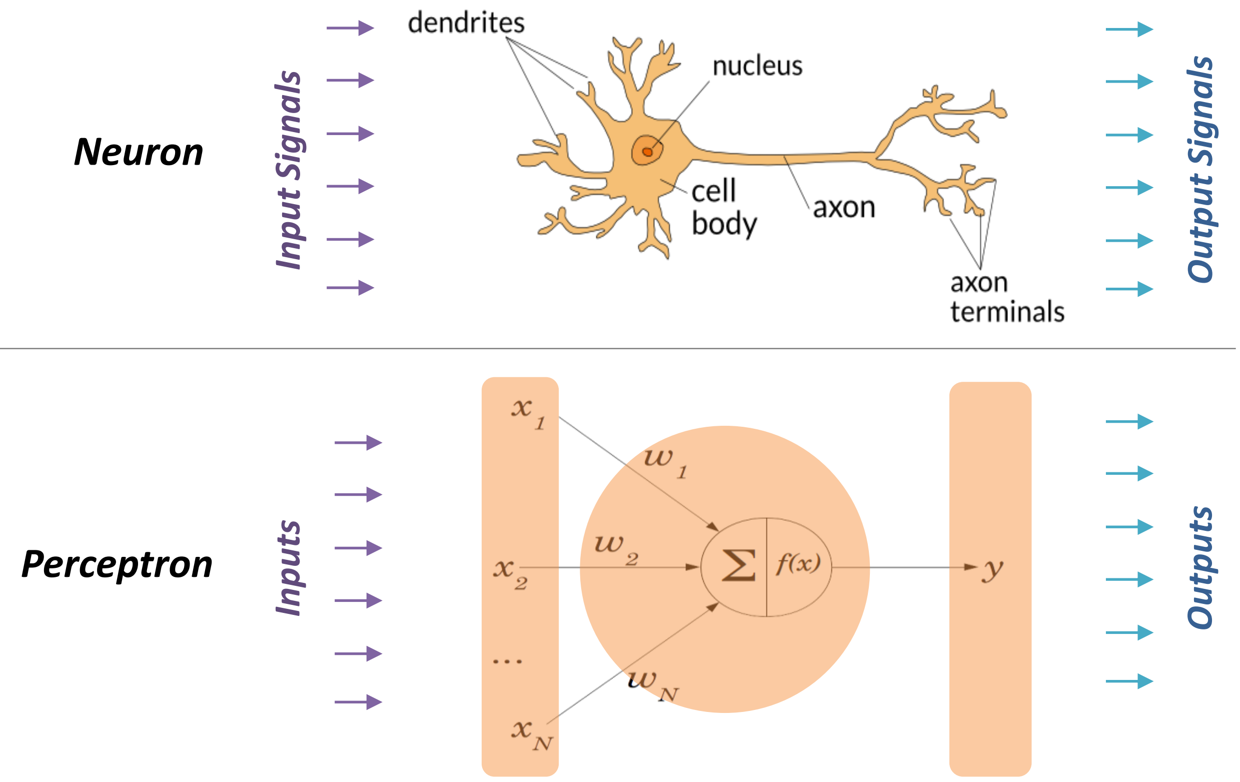
1. Previous Lecture: Word Embedding Review
2. Word Embedding Evaluation
3. **Deep Neural Network for Natural Language Processing**
    1. Perceptron and Neural Network (NN)
    2. Multilayer Perceptron
    3. Applications
4. Next Week Preview

    See how the Deep Learning can be used for NLP

    - Text Classification, etc.

Deep Learning with Neural Network

Neuron and Perceptron

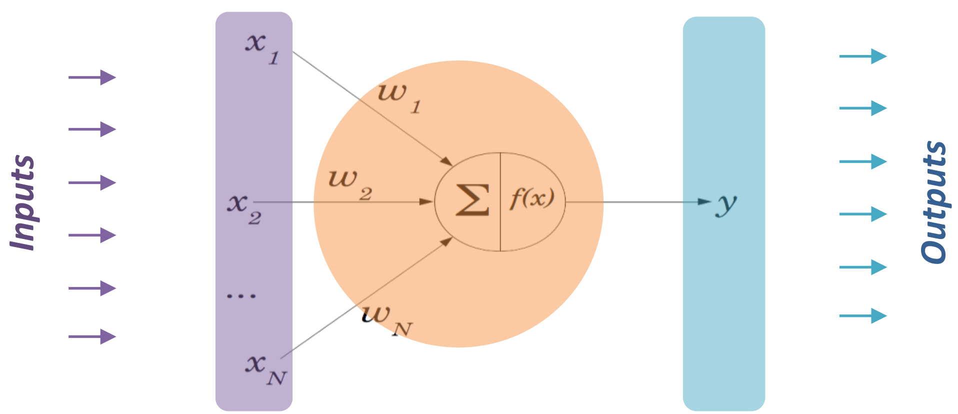


Deep Learning with Neural Network

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** • E.g. word meaning, sentiment, name entity

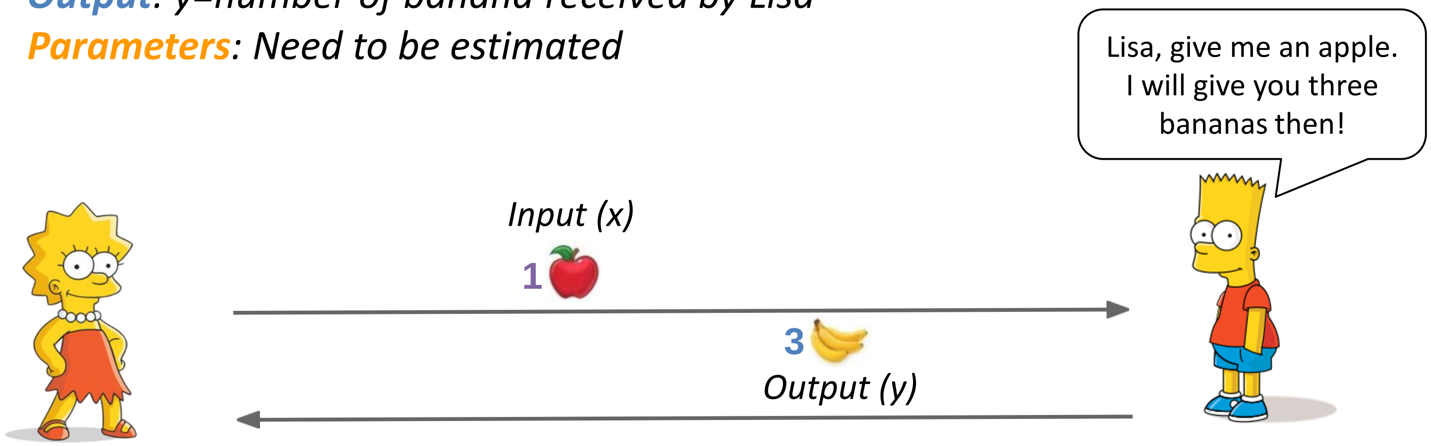
Perceptron



3 Deep Learning for NLP

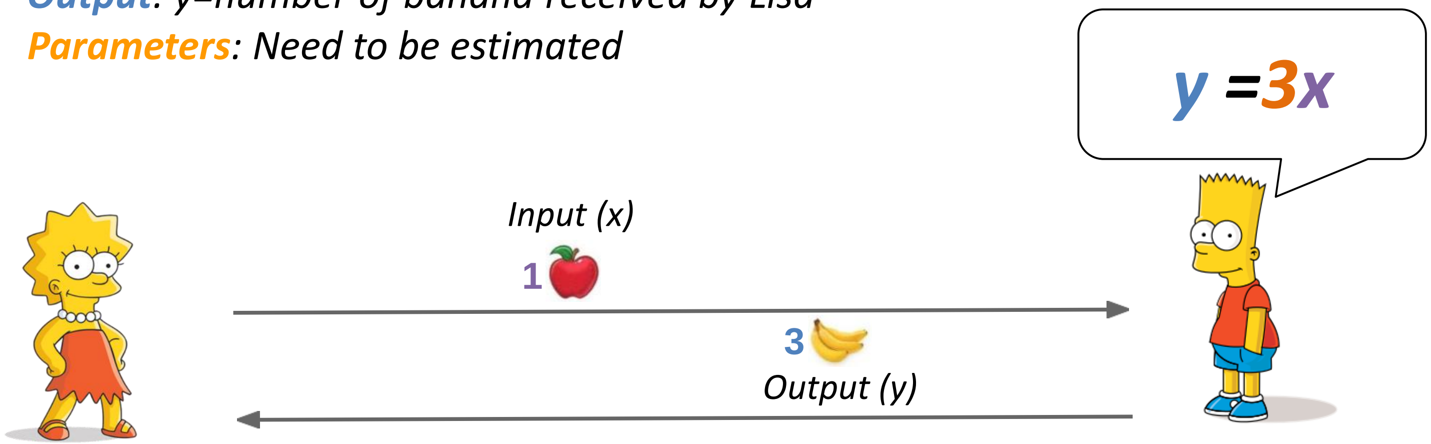
Deep Learning with Neural Network

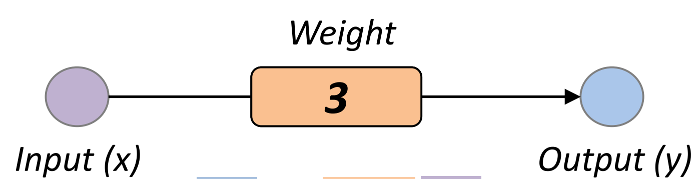
Input: x=number of apple given by Lisa
Output: y=number of banana received by Lisa
Parameters: Need to be estimated



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*





$$y = Wx$$

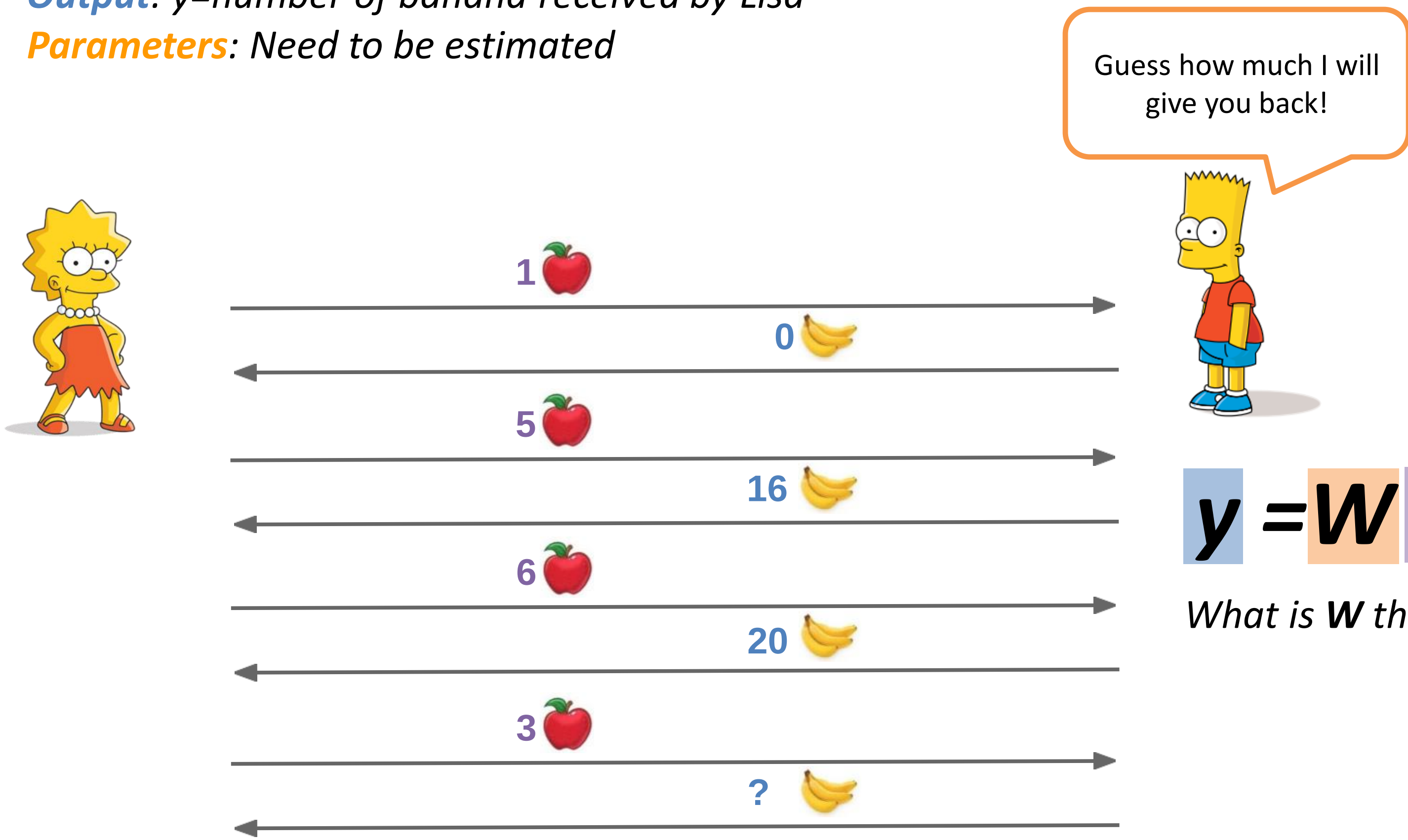
3 Deep Learning for NLP

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*

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

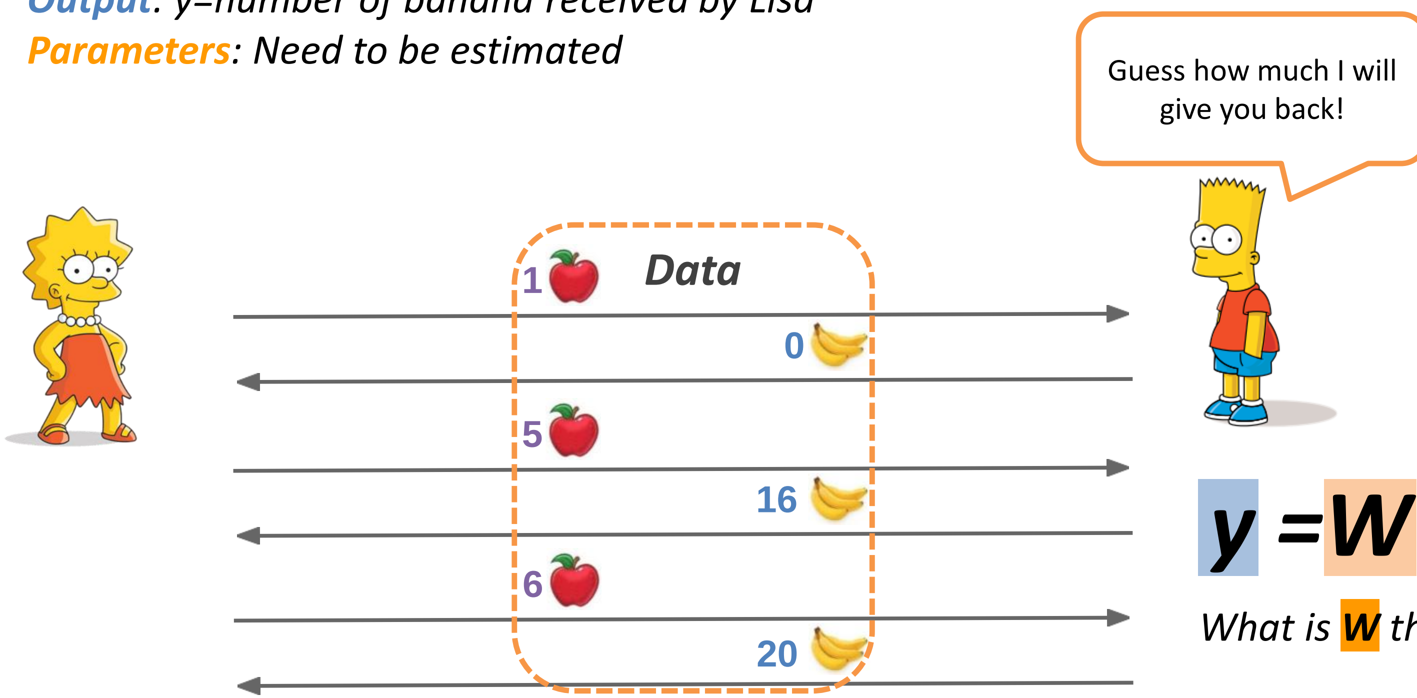


$$y = W x$$

*What is **W** then?*

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



$$y = W x$$

*What is **W** then?*

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

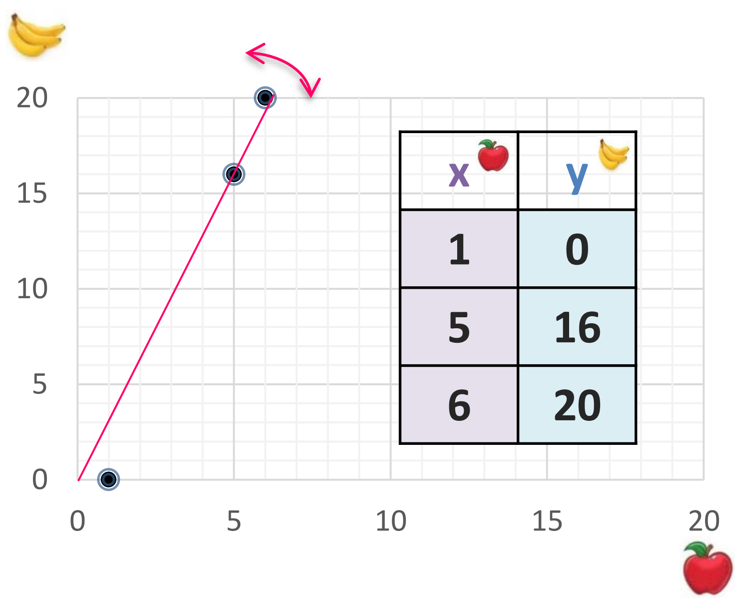
x 🍎	y 🍌
1	0
5	16
6	20

$$y = Wx$$

*What is **W** then?*

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



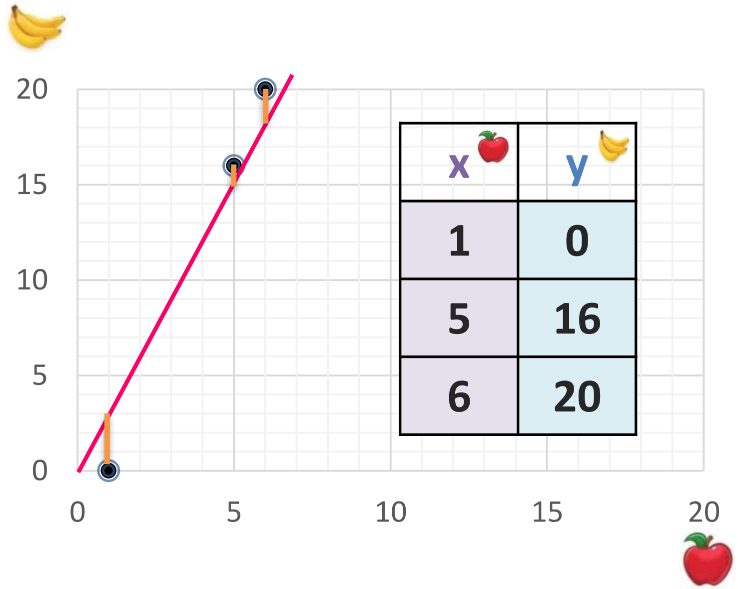
x	y
1	0
5	16
6	20

$$y = W x$$

*What is **W** then?*

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*



x	y
1	0
5	16
6	20

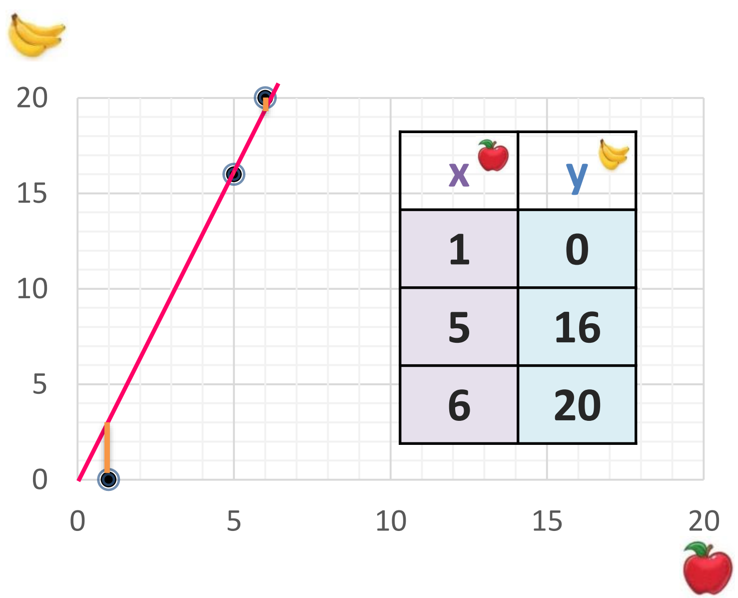
$$y = W x$$

*What if W is **3**?*

3 = 3 x 1
15 = 3 x 5
20 = 3 x 6

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



x	y
1	0
5	16
6	20

$$y = W x$$

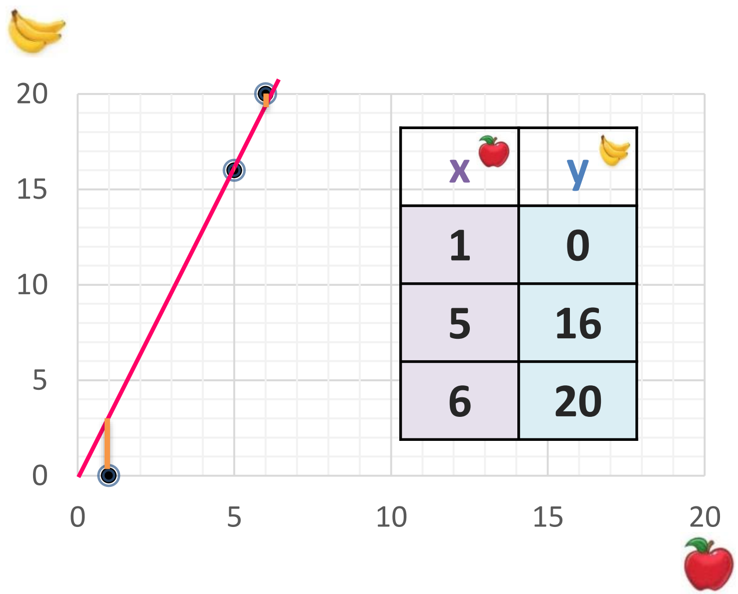
What if W is ***3.2****?*

3.2 = 3.2 x 1
16 = 3.2 x 5
19.2 = 3.2 x 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



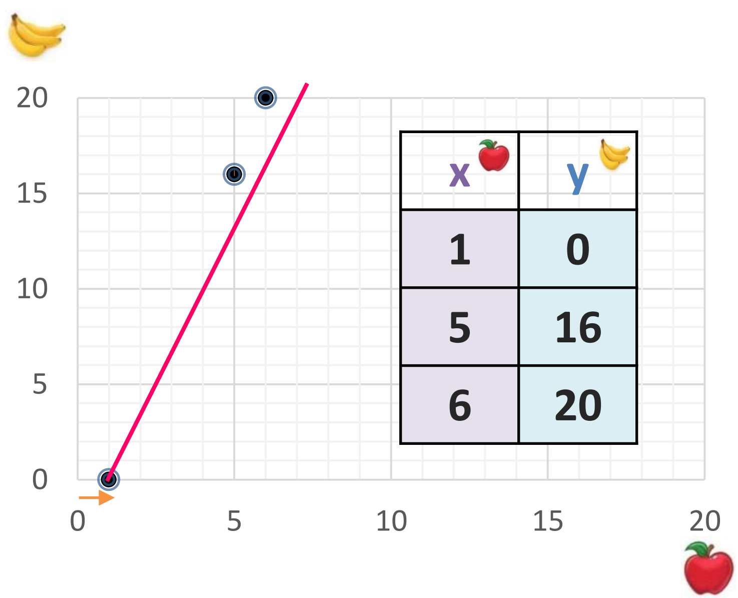
x	y
1	0
5	16
6	20

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

Weight is not enough…

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



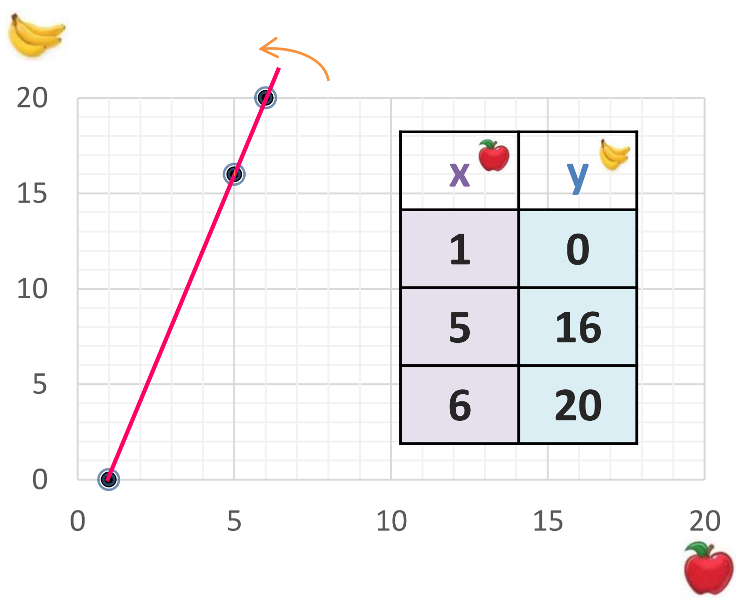
x	y
1	0
5	16
6	20

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

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

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



x	y
1	0
5	16
6	20

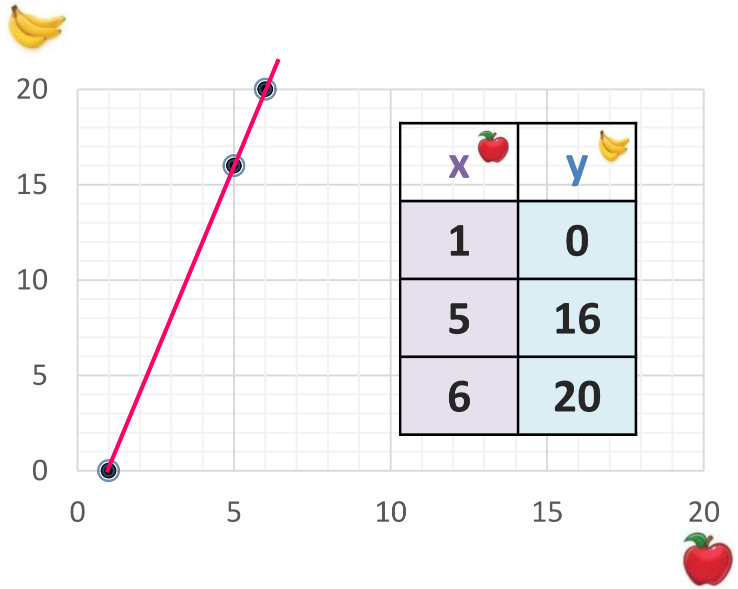
weight *bias*

$$y = W x + b$$

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

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*



x	y
1	0
5	16
6	20

Model $y = W x + b$ (weight: W; bias: b)

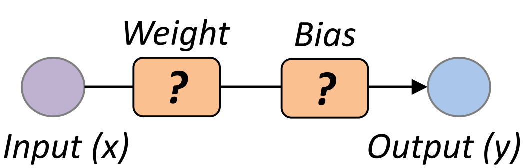
How can we find the parameters, ***w*** *and* ***b****?*

Deep Learning with Neural Network - Cost

Actual Data

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

x 🍎	y 🍌
1	0
5	16
6	20



Model Ex#1

$$\hat{y} = \overset{\text{weight}}{1}\, x + \overset{\text{bias}}{0}$$

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

Model Ex#2

$$\hat{y} = \overset{\text{weight}}{2}\, x + \overset{\text{bias}}{2}$$

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

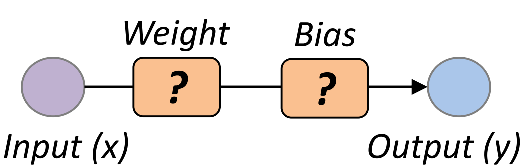
🤔 ***Which one is closer?***

Deep Learning with Neural Network - Cost

Actual Data

$$\underbrace{y}_{} = \overset{weight}{?}\ x + \overset{bias}{?}$$

x	y
1	0
5	16
6	20



Model Ex#1

$$\hat{y} = \overset{weight}{1}\ x + \overset{bias}{0}$$

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

Model Ex#2

$$\hat{y} = \overset{weight}{2}\ x + \overset{bias}{2}$$

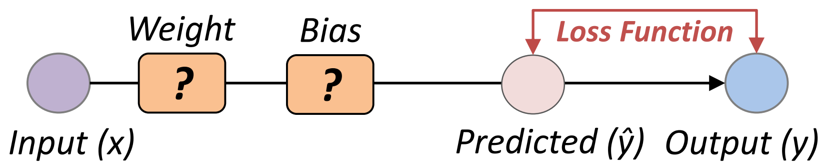
x	predicted $\hat{y}$	actual y	cost $(y-\hat{y})^2$
1	4	0	
5	12	16	
6	14	20	

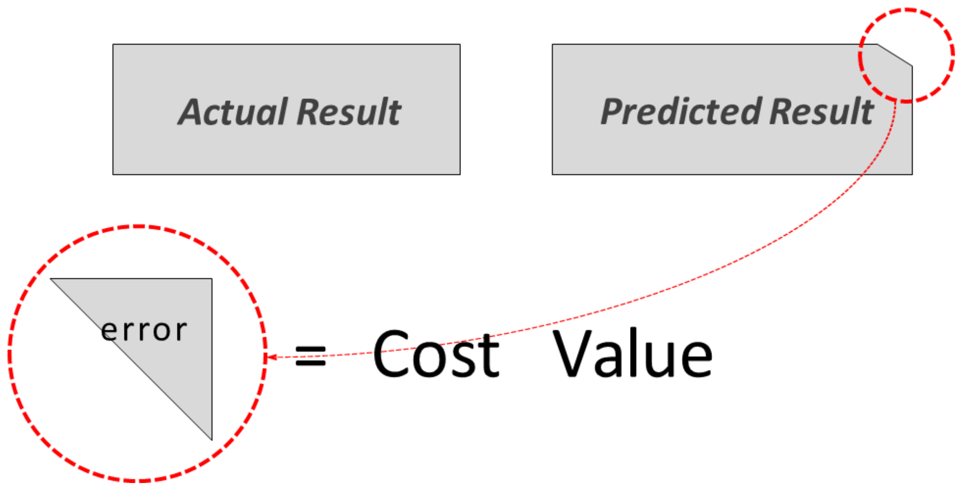
Let's calculate the cost(loss)!

Mean Squared Error (MSE)

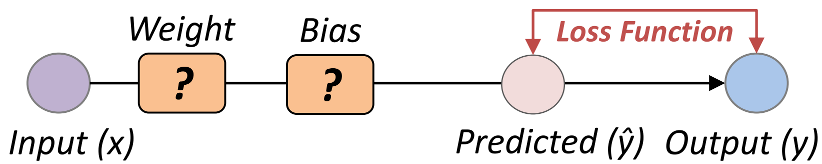
$$C(w,b) = \sum_{n\in\{0,1,2\}} (y_n - \hat{y}_n)$$

WAIT! Loss Function? Cost Calculation?





WAIT! Loss Function? Cost Calculation?



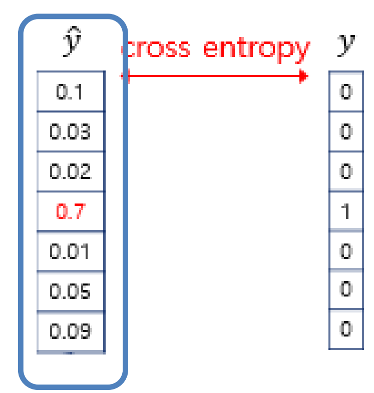
1) Mean Squared Error (MSE): measures the average of the squares of the errors

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

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

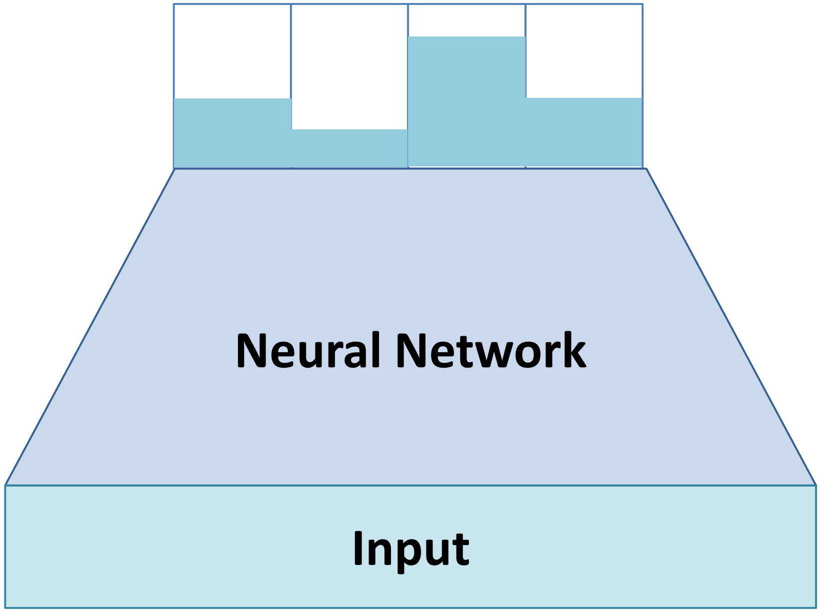
$$L_{\text{cross-entropy}}(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_{i} y_i \log(\hat{y}_i)$$

softmax



3 Deep Learning for NLP

WAIT! Softmax!!



1. Define N Class

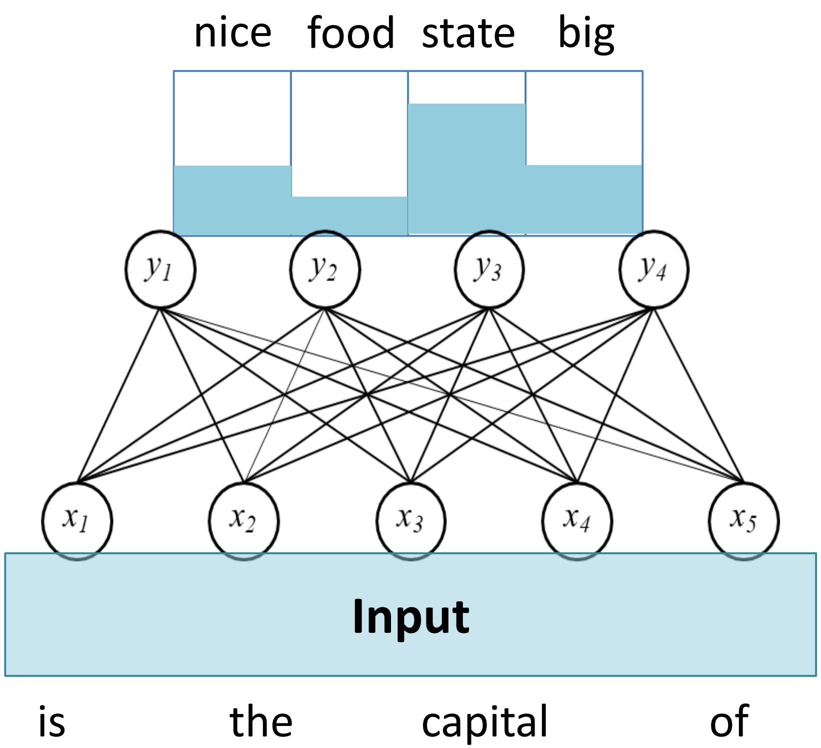
5. Score given classes

2. Construct Network

4. Classification

3. Input

WAIT! Softmax!!



1. Define N Class

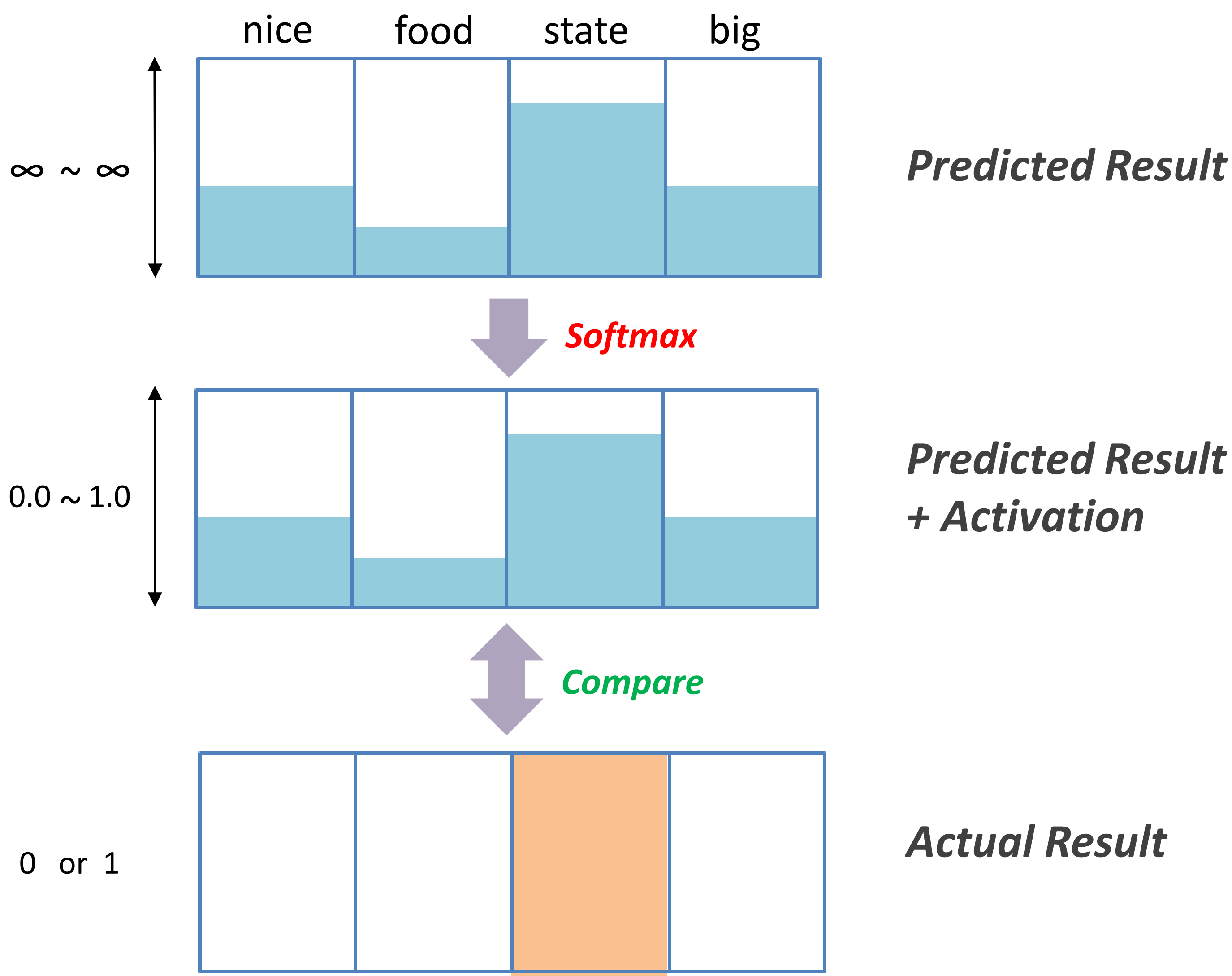
5. Score given classes

2. Construct Network

4. Classification

3. Input

WAIT! Softmax!!



THE UNIVERSITY OF SYDNEY

WAIT! Softmax!!

***Depends on the dataset**

Problem type	Last-layer activation	Loss (Cost) function	Example
Binary classification	sigmoid	Binary Cross Entropy	Sentiment analysis (Positive/Negative)
Multi-class, single-label classification	softmax	Categorical Cross Entropy	Part-of-Speech tagging Named Entity Recognition
Multi-class, multi-label classification	sigmoid	Binary Cross Entropy	Multi-topic classification, one can have multiple topics
Regression to arbitrary values	None	MSE (Mean Squared Error)	Predict house price
Regression to values between 0 and 1	sigmoid	MSE or Binary Cross Entropy	Engine health assessment where 0 is broken, 1 is new

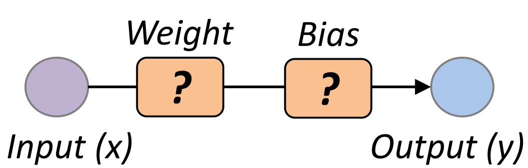
THE UNIVERSITY OF SYDNEY

Deep Learning with Neural Network - Cost

Actual Data

$$y = \underbrace{?}_{\text{weight}}\, x + \underbrace{?}_{\text{bias}}$$

x 🍎	y 🍌
1	0
5	16
6	20



Model Ex#1

$$\hat{y} = \underbrace{1}_{\text{weight}}\, x + \underbrace{0}_{\text{bias}}$$

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

Model Ex#2

$$\hat{y} = \underbrace{2}_{\text{weight}}\, x + \underbrace{2}_{\text{bias}}$$

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

😎 ***Let's calculate the cost(loss)!***

Mean Squared Error (MSE)

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

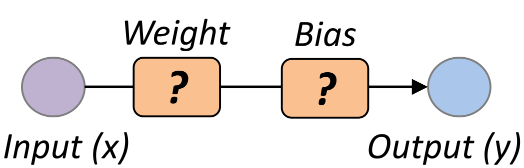
Deep Learning with Neural Network - Cost

Actual Data

weight, bias

$$y = ?\,x + ?$$

x	y
1	0
5	16
6	20



Model Ex#1

weight, bias

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

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

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

Model Ex#2

weight, bias

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

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

C(2,2) = **68**

Let's calculate the cost(loss)!

Mean Squared Error (MSE)

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

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

THE UNIVERSITY OF SYDNEY

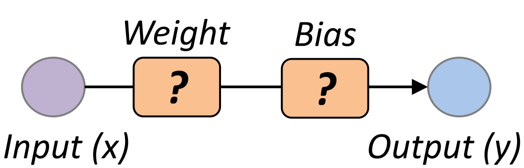
Deep Learning with Neural Network - Cost

Actual Data

weight *bias*

$$y = ?\,x + ?$$

x	y
1	0
5	16
6	20



Model Ex#1

weight *bias*

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

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

C(1,0) = 318

Model Ex#2

weight *bias*

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

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

C(2,2) = 68 Winner

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

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

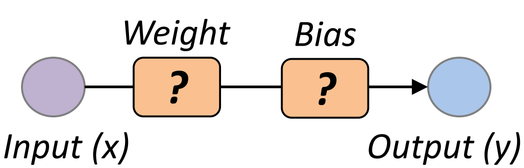
Deep Learning with Neural Network - Optimizer

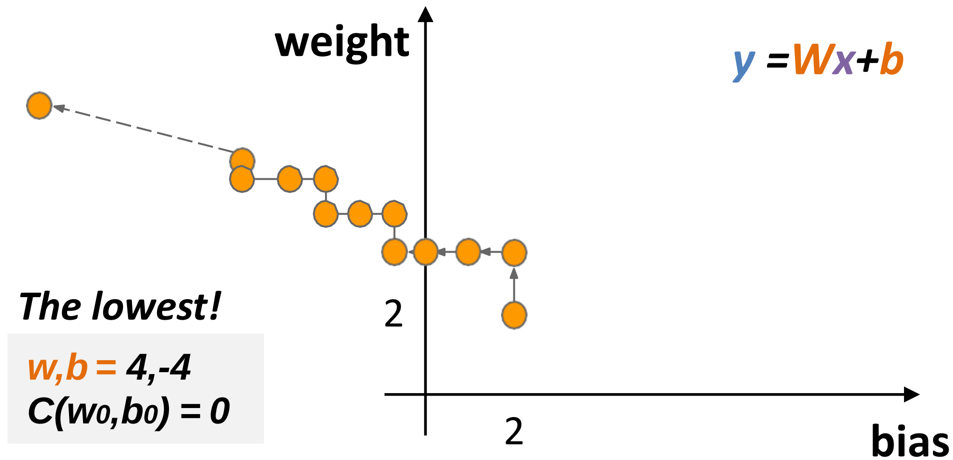
Actual Data

weight *bias*

$$y = ?\,x + ?$$

x 🍎	y 🍌
1	0
5	16
6	20





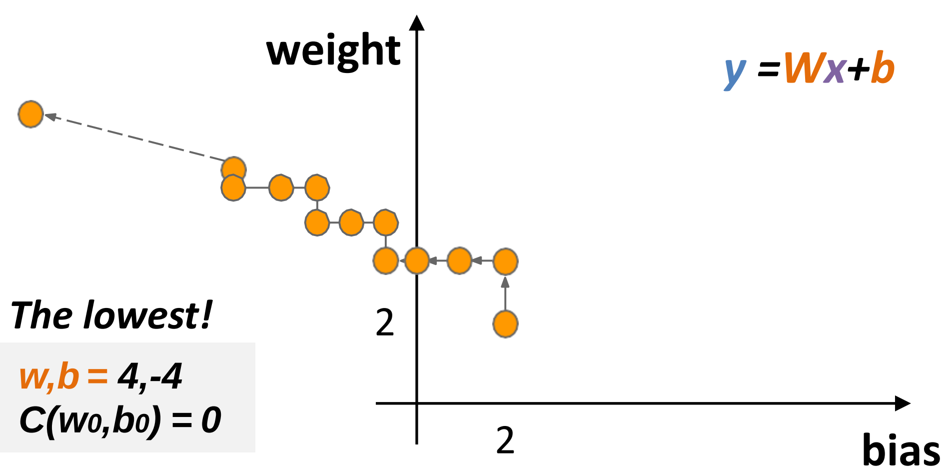
😎 ***Let's calculate the costs and get the lowest one!***

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

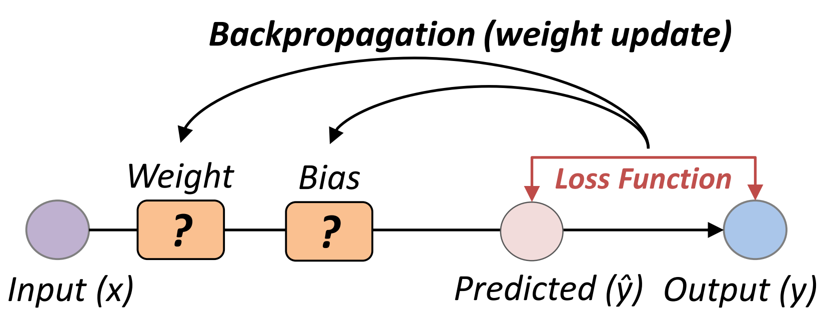
Deep Learning with Neural Network - Optimizer

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

x 🍎	y 🍌
1	0
5	16
6	20



$y = Wx + b$



arg min C(w,b)

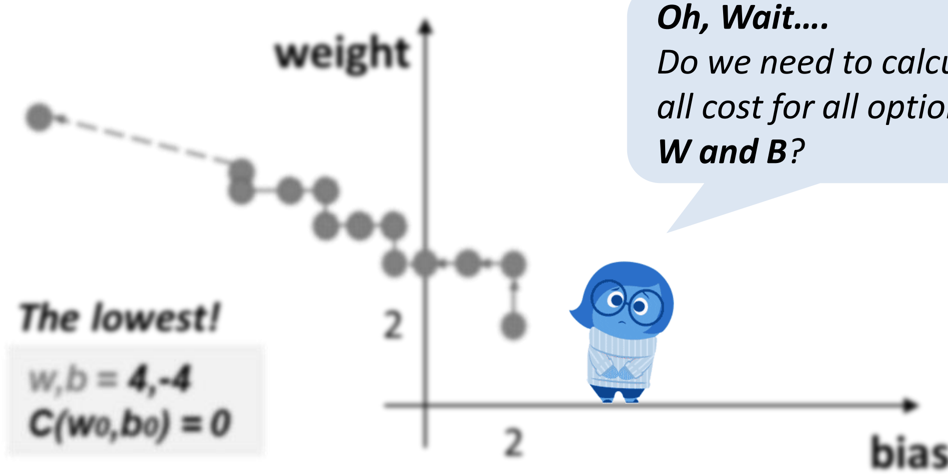
w,b∈[-∞,∞]

Deep Learning with Neural Network - Optimizer

weight *bias*

$$y = 4x - 4$$

x	y
1	0
5	16
6	20



Oh, Wait….
*Do we need to calculate all cost for all options of **W and B**?*

<u>Expensive to compute (hours or days)</u>

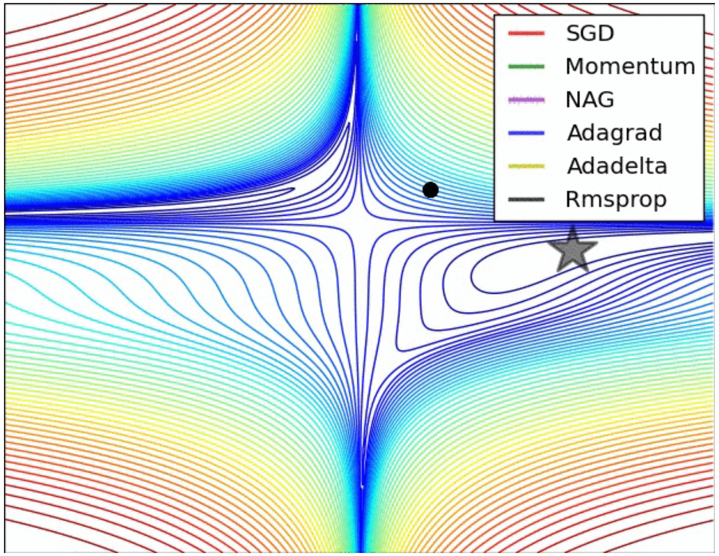
$$\underset{w,b\in[-\infty,\infty]}{\arg\min}\ C(w,b)$$

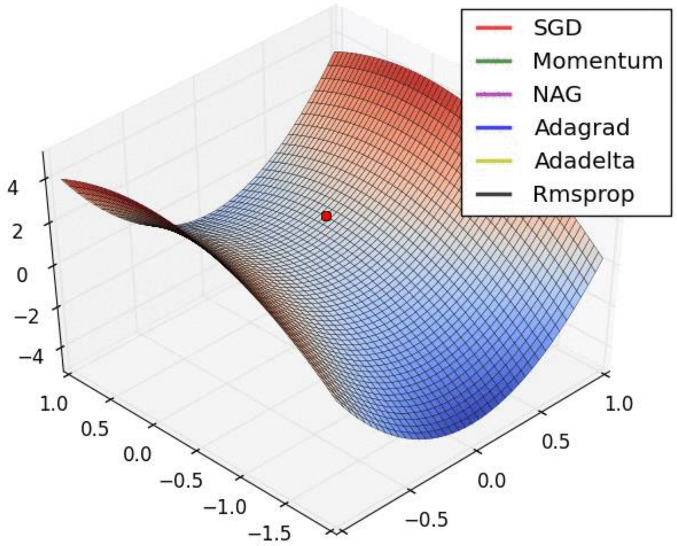
Finding the Optimal weight and bias – Gradient Descent

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

3 Deep Learning for NLP

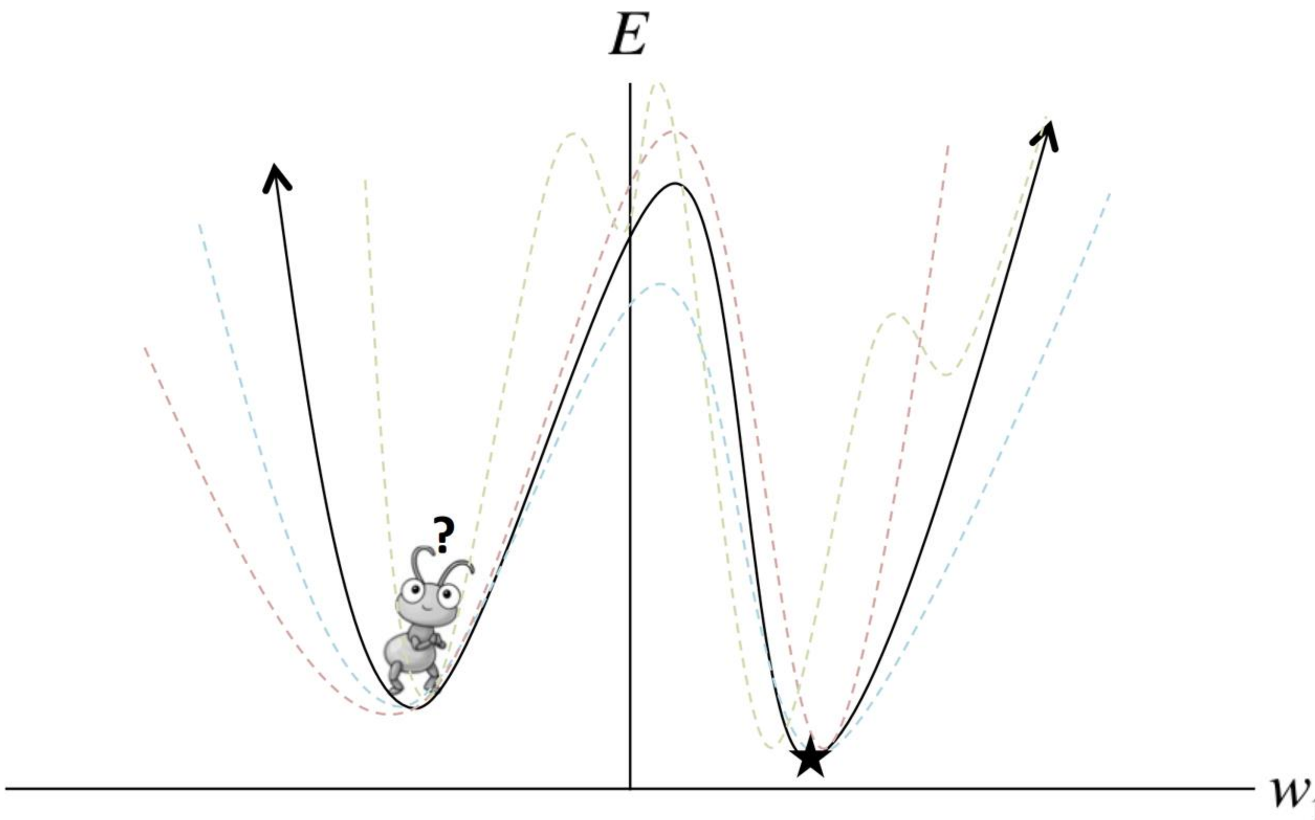
Finding the Optimal weight and bias – Gradient Descent





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

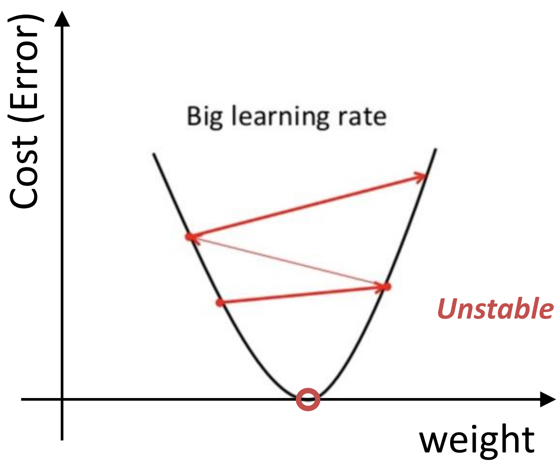
Finding the Optimal weight and bias – Gradient Descent



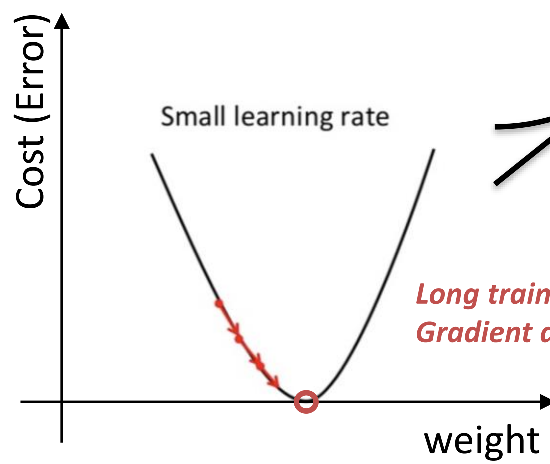
There are different types of Gradient descent optimization algorithms:
Batch Gradient Descent, Stochastic Gradient Descent, Momentum, Adam, etc.

Choose the optimal Learning Rate!

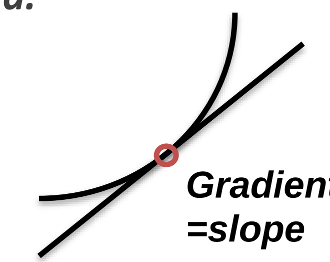
Learning Rate: *a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated.*



Unstable performance



Long training time
Gradient descent can be slow



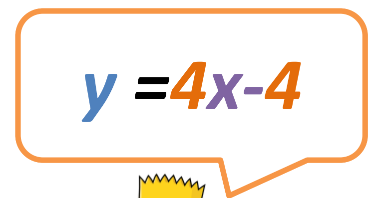
Gradient =slope

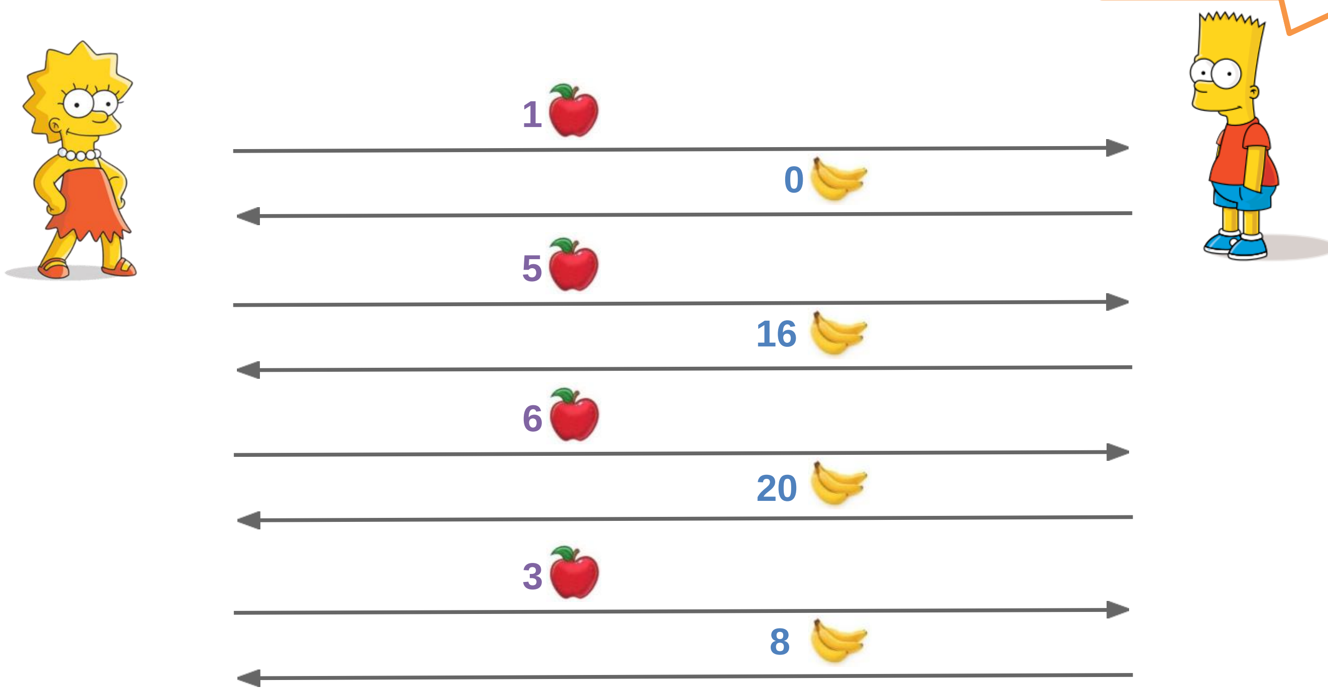
*new_weight = existing_weight — learning_rate * gradient*

*new_weight = existing_weight — learning_rate * (current_output – desired output) *gradient(current output) * existing_input*

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

pixel(1,1) pixel(1,2)

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

Data

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

3

Deep Learning for NLP

Deep Learning with Neural Network

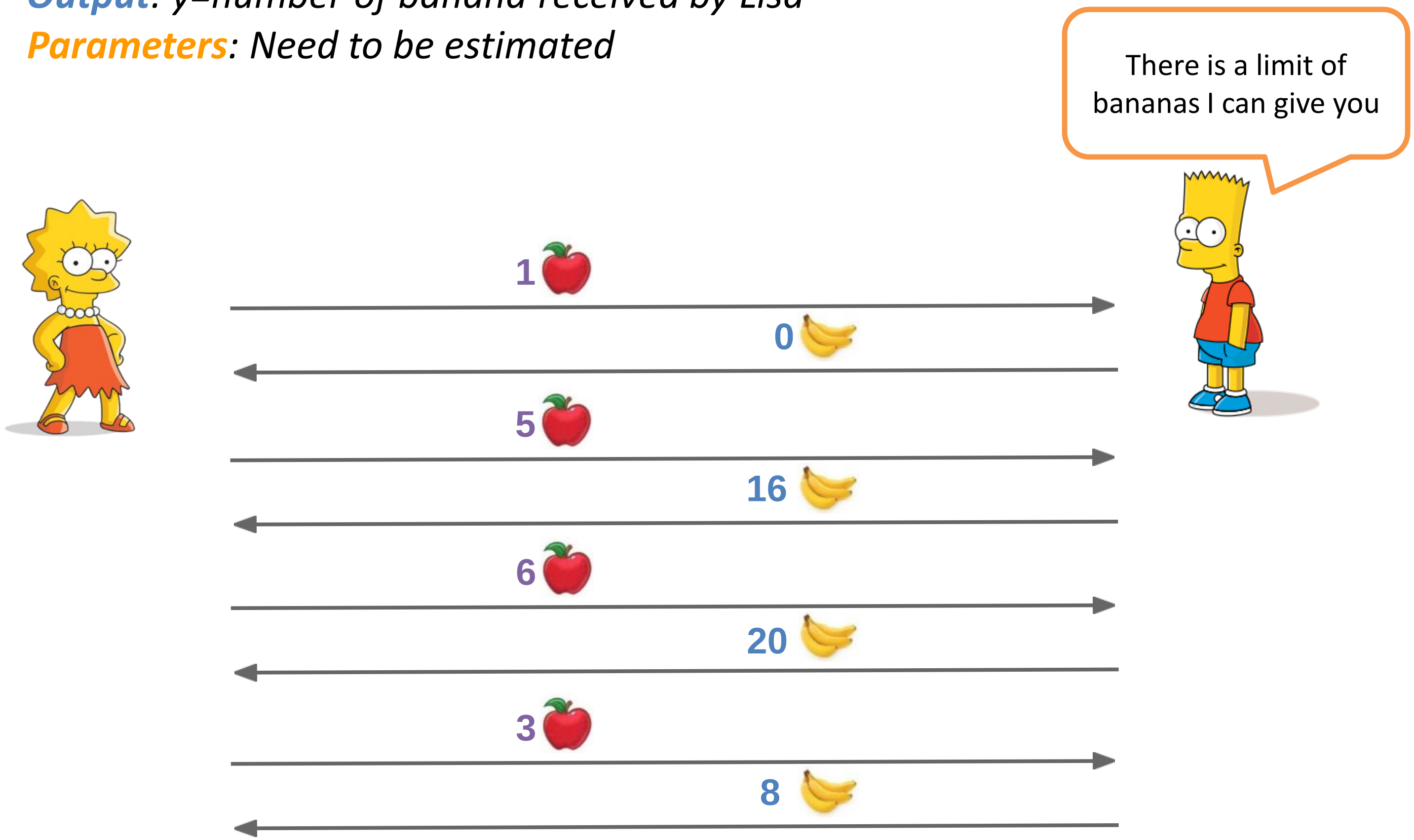
$$y = w_1\overset{\textit{Vector1}}{x_1} + w_2\overset{\textit{Vector2}}{x_2} + w_3x_3 + w_4x_4 + \ldots + w_nx_n + b$$

*Millions of **Parameters***

*Millions of **Samples***

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

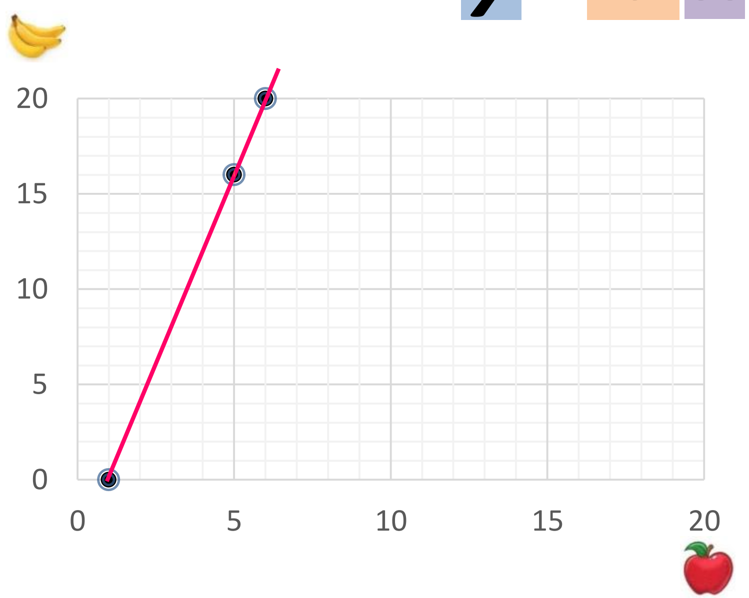


Deep Learning with Neural Network

Nonlinear Neural Network

Data

x	y
1	0
5	16
6	20



weight *bias*

$$y = 4x - 4$$

3 Deep Learning for NLP

Deep Learning with Neural Network

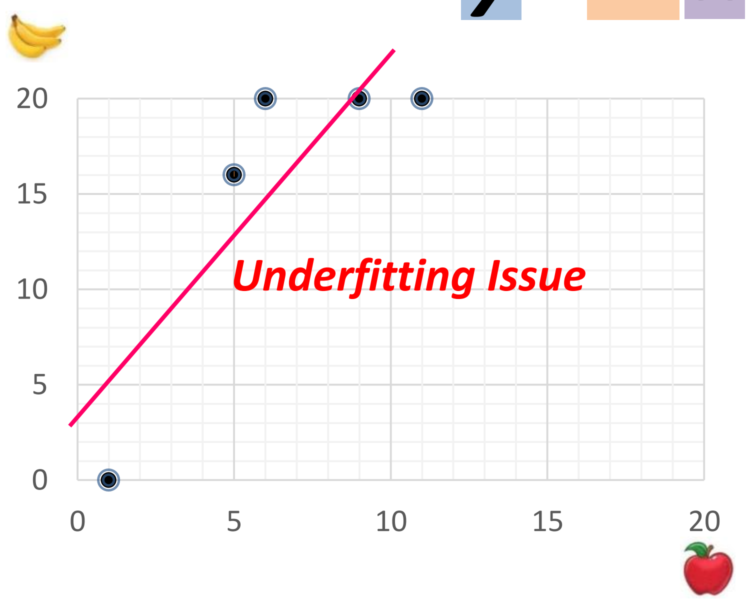
Nonlinear Neural Network

weight *bias*

$$y = 2x + 3$$

Data

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



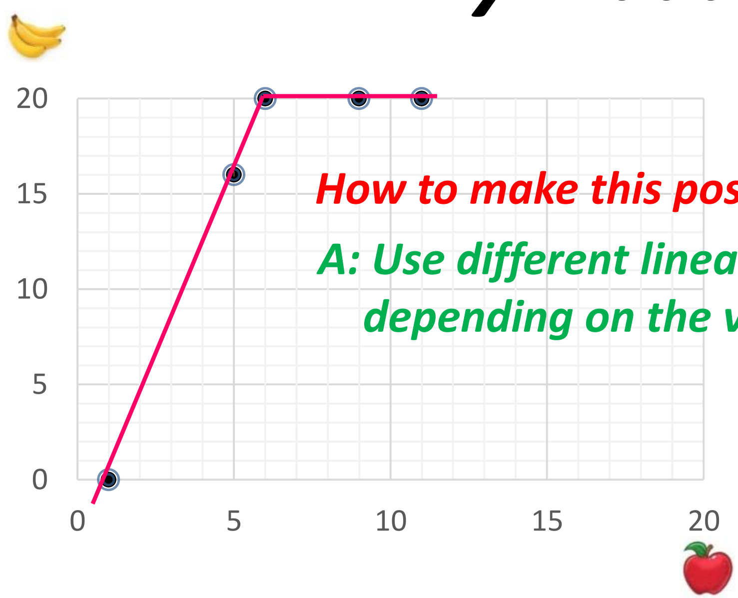
Deep Learning with Neural Network

Nonlinear Neural Network

y = ???

Data

x 🍎	y 🍌
1	0
5	16
6	20
9	20
11	20



How to make this possible?

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

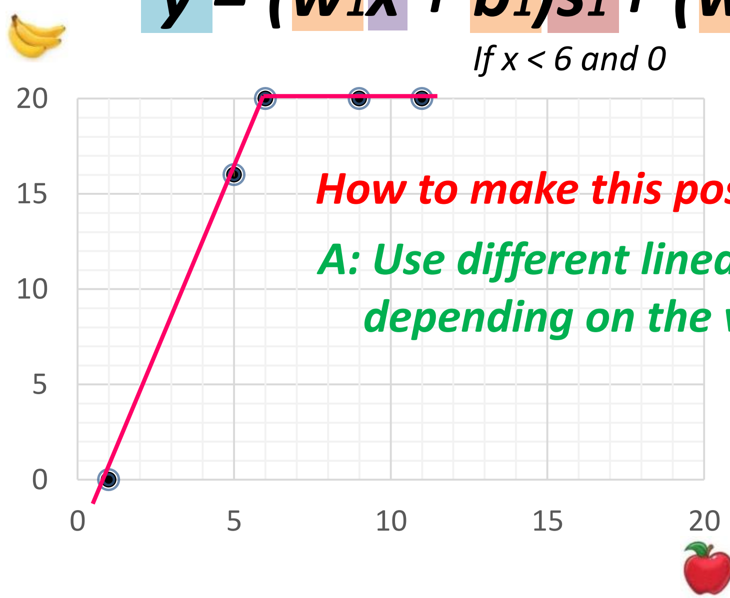
Deep Learning with Neural Network

Nonlinear Neural Network

Data

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

$$y = (\overset{\text{weight}}{w_1} x + \overset{\text{bias}}{b_1})\underset{\text{If } x < 6 \text{ and } 0}{s_1} + (\overset{\text{weight}}{w_2} x + \overset{\text{bias}}{b_2})\underset{\text{If } x >= 6 \text{ and } 0}{s_2}$$



How to make this possible?

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

Deep Learning with Neural Network

Nonlinear Neural Network

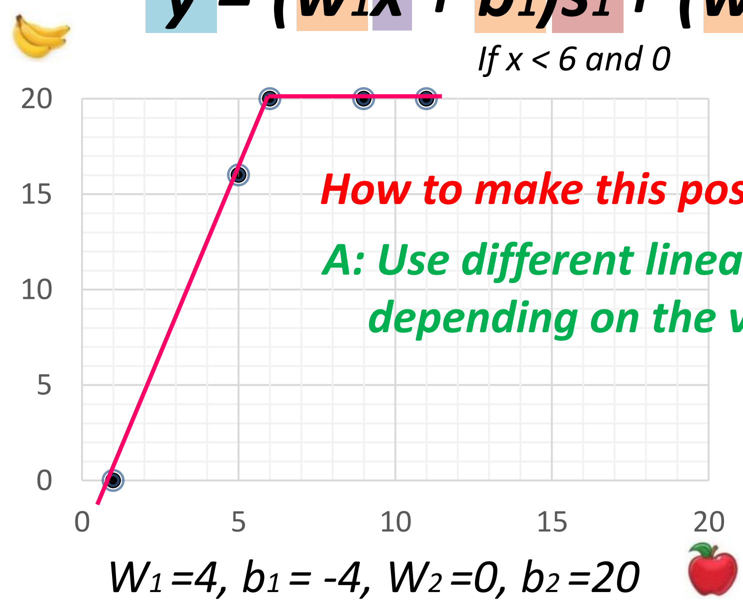
Data

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

weight *bias* *weight* *bias*

$$y = (w_1 x + b_1)s_1 + (w_2 x + b_2)s_2$$

If x < 6 and 0 *If x >= 6 and 0*

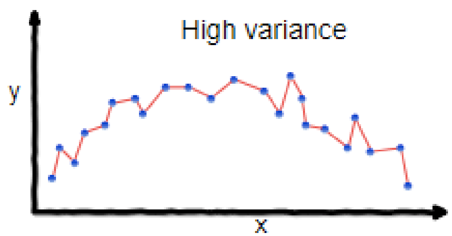


How to make this possible?

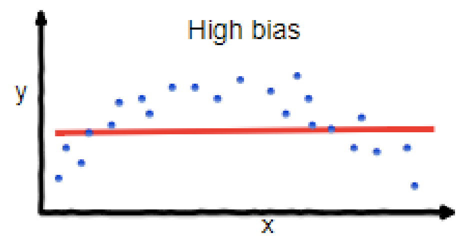
A: Use different linear functions depending on the value of x

$W_1 = 4$, $b_1 = -4$, $W_2 = 0$, $b_2 = 20$

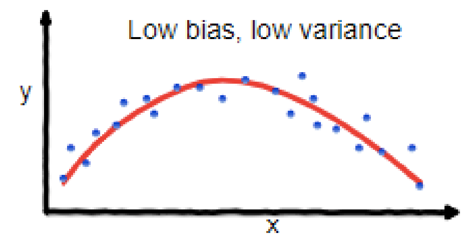
Deep Learning with Neural Network



overfitting

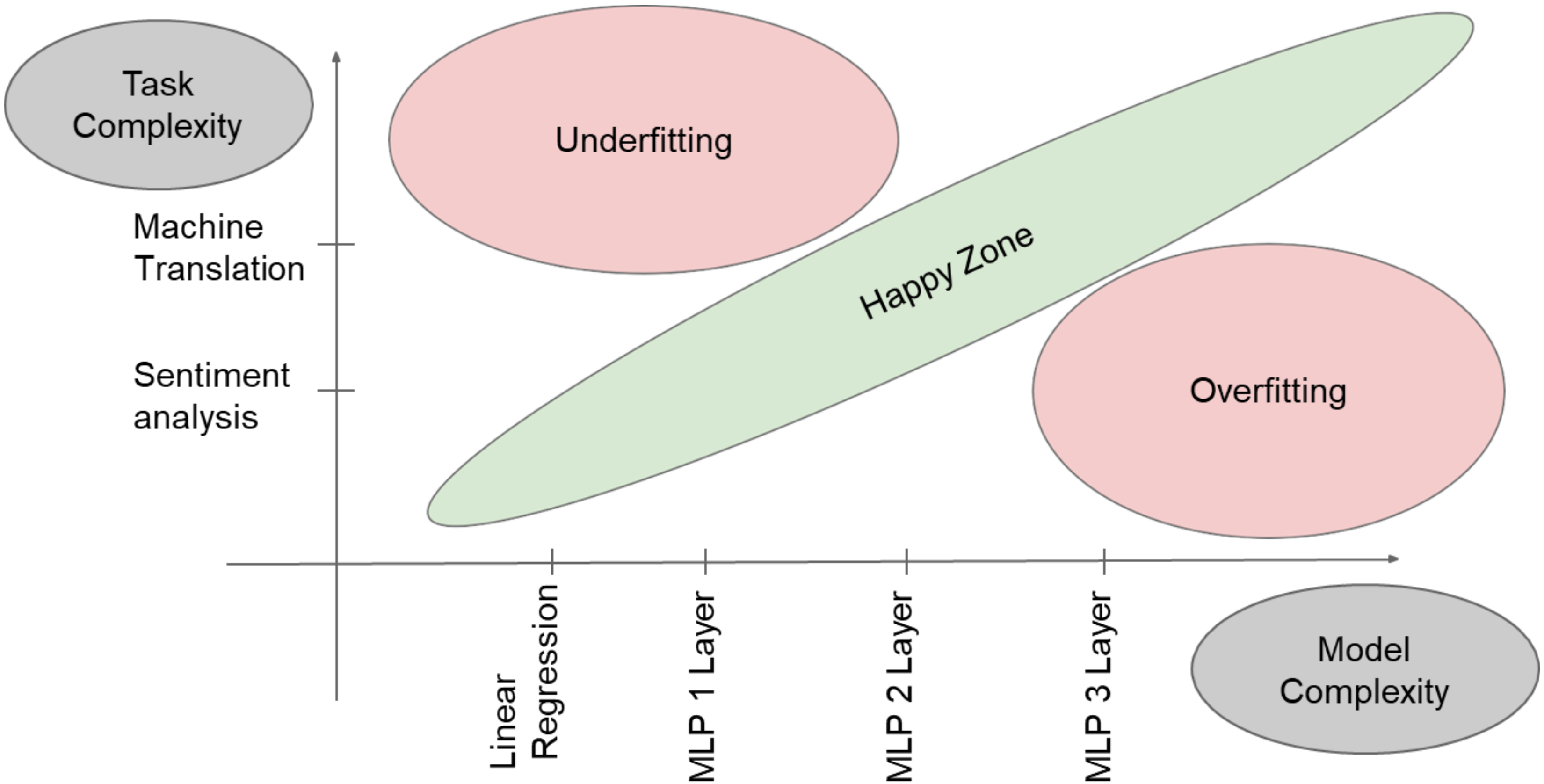


underfitting

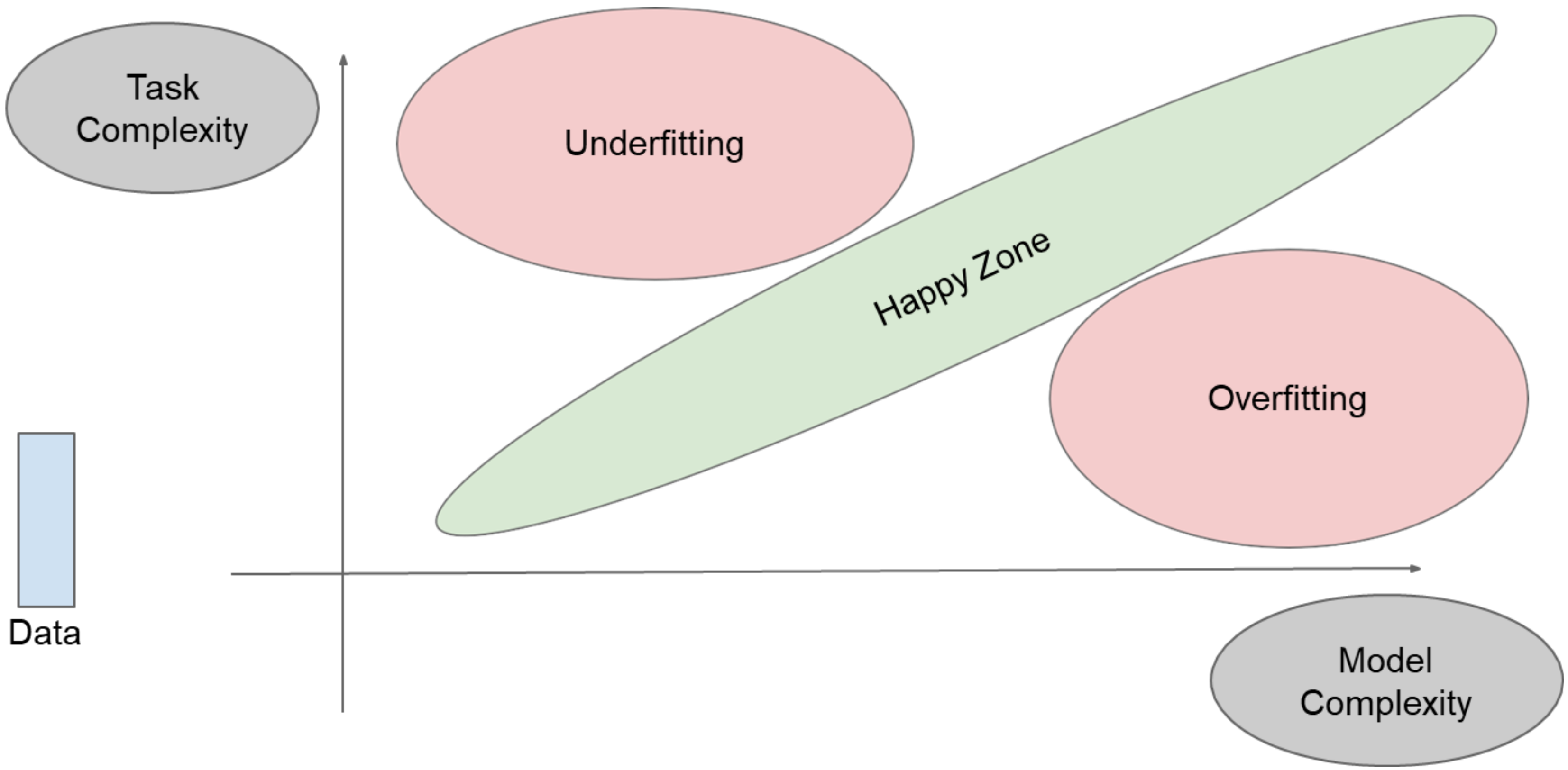


Good balance

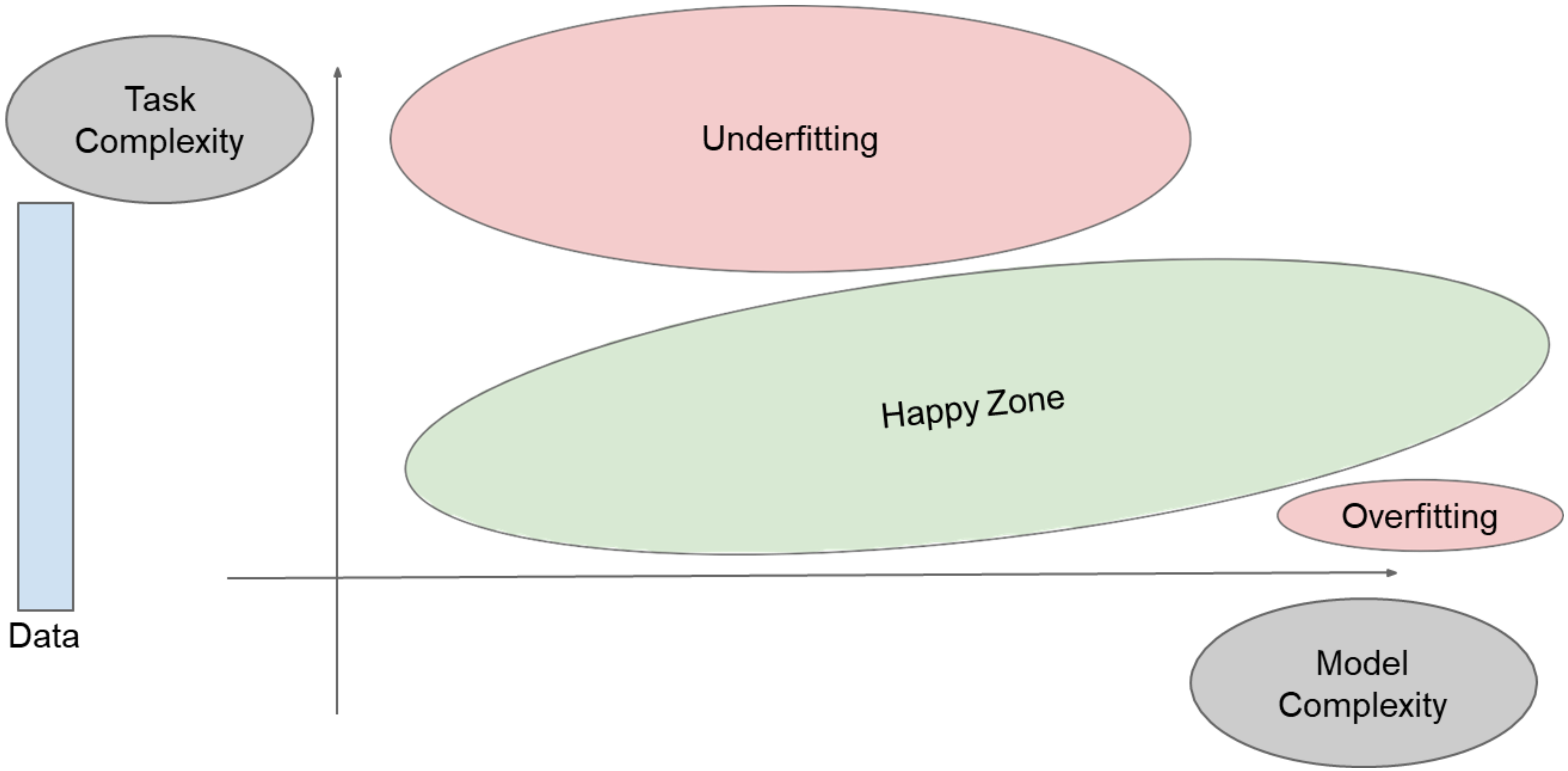
3 Deep Learning for NLP



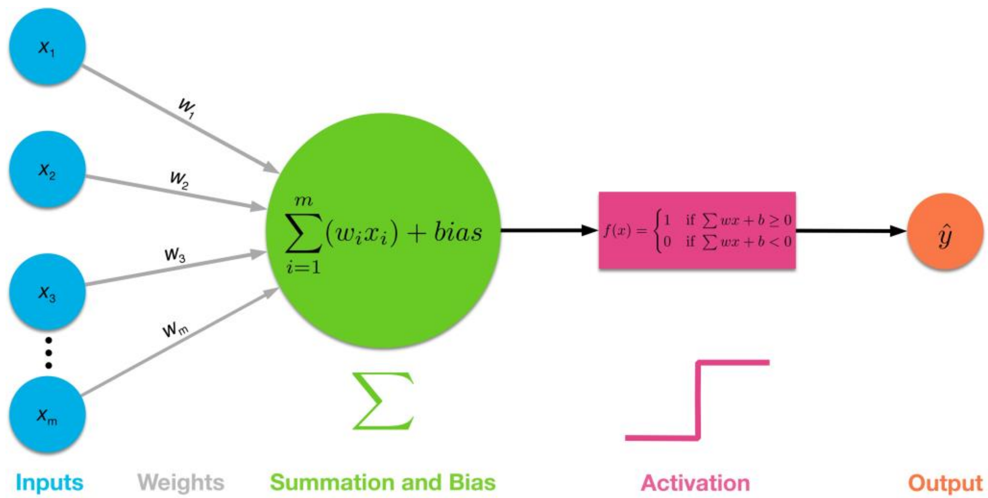
3 Deep Learning for NLP

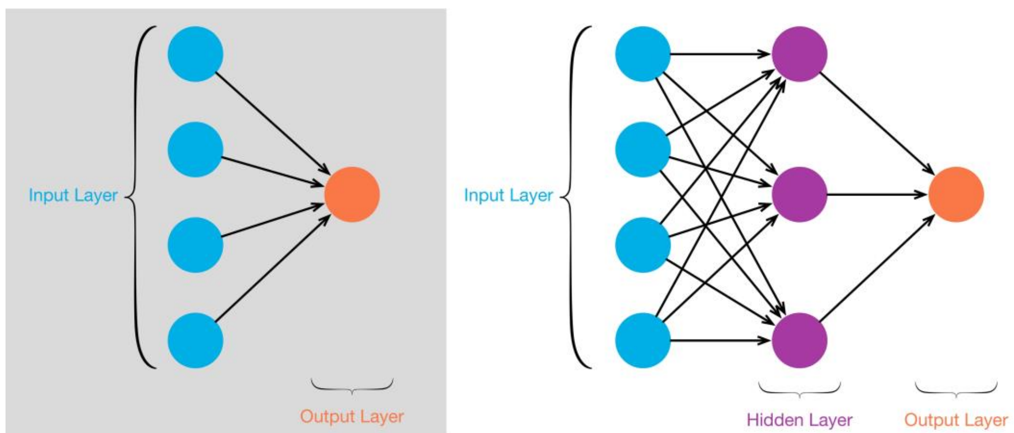


3 Deep Learning for NLP



Single Neuron VS Multilayer



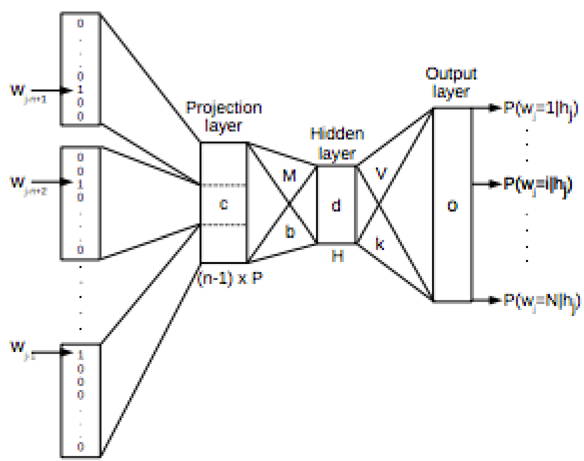


3 Deep Learning for NLP

Application

Previous Word Embedding Models

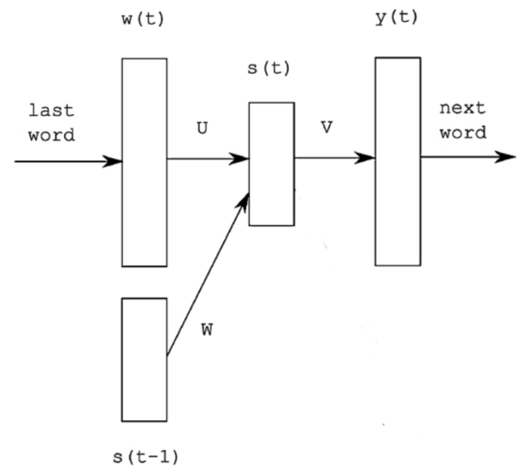
NxP NxPxH HxV



Neural Net Language Model (NNLM)

- The first word embedding model!
- N – how many previous words will be checked
- Only care about the previous words
- Long Computation time! Really slow

H HxH HxV



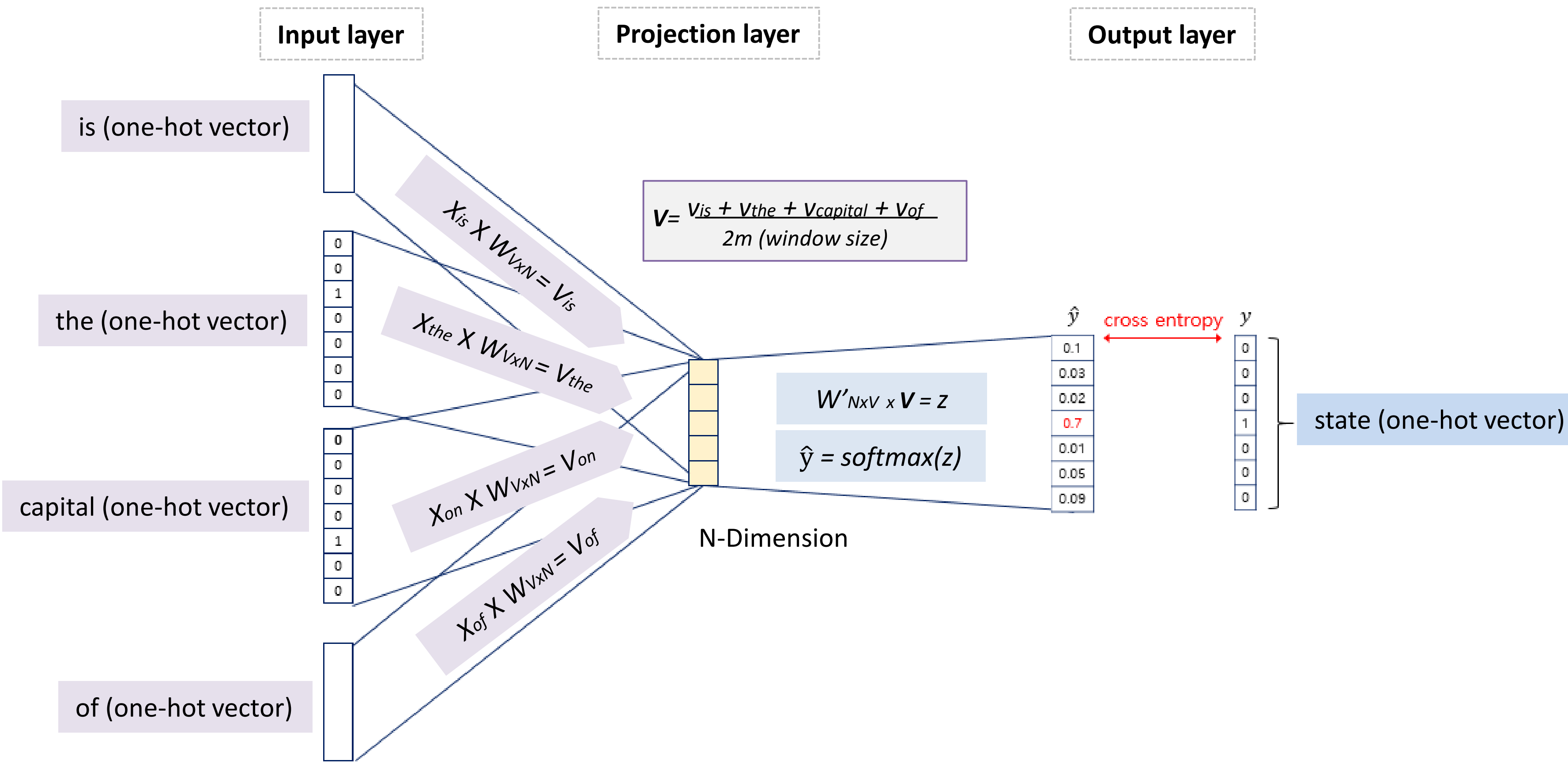
Recurrent Neural Net Language Model (RNNLM)

- No need to setup the No of previous words
- No projection layer
- Only care about the previous word
- A bit faster than NNLM

3 Deep Learning for NLP

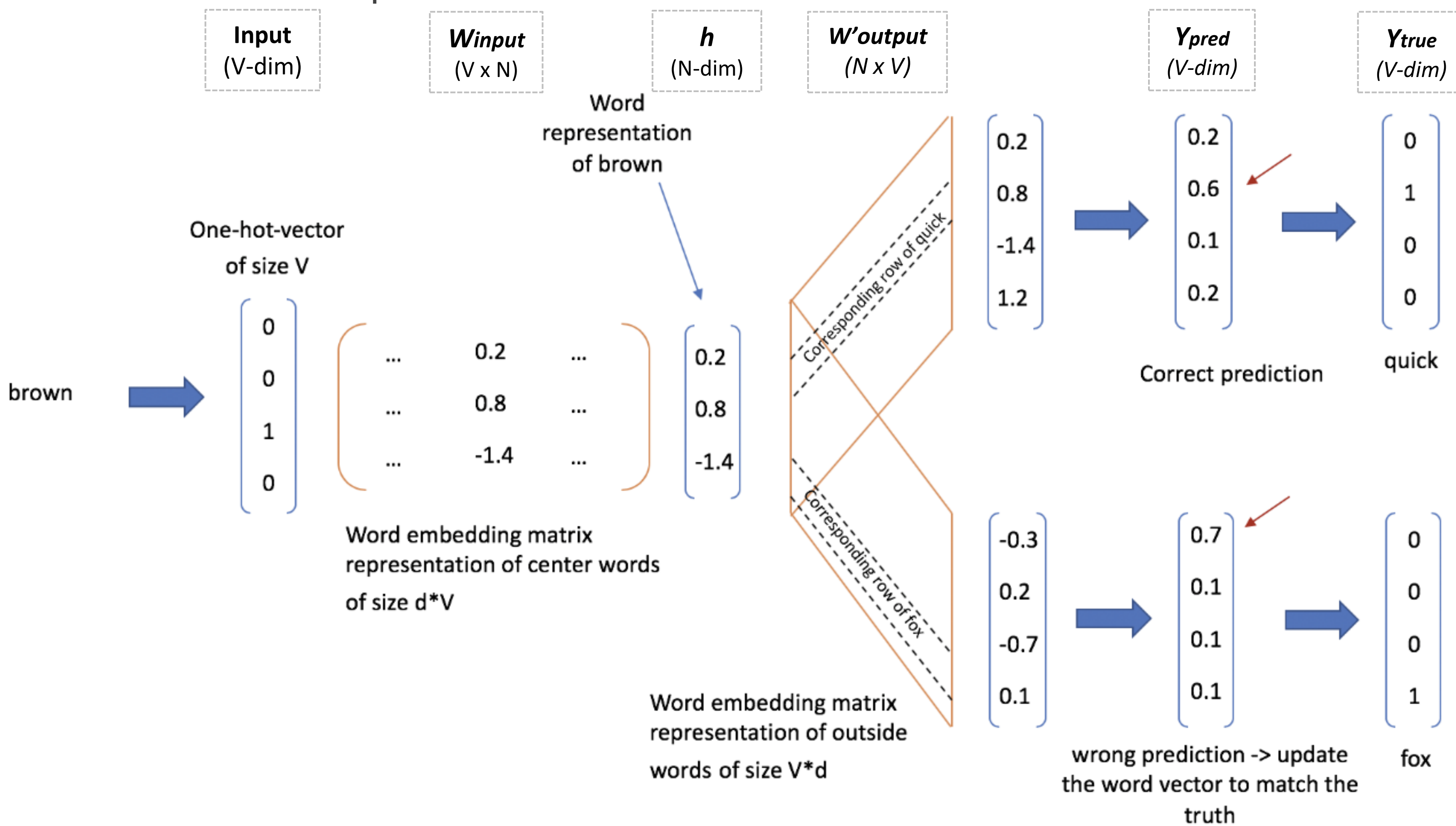
Application

Word2Vec with CBOW



Application

Word2Vec with Skip Gram



3 Deep Learning for NLP

Negative Sampling

new_weight = existing_weight — learning_rate * gradient

W_output (old)

-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

(11X3)

—

Learning R. $\boxed{0.05}$

×

grad_W_output

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

(11X3)

=

W_output (new)

-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**

(11X3)

W_output (old)

-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

(11X3)

—

Learning R. $\boxed{0.05}$

×

grad_W_output

Not computed!

Positive sample, w_o	0.031	0.030	0.041
Negative sample, k=1	-0.090	0.031	-0.065
Negative sample, k=2	0.056	0.098	-0.061
Negative sample, k=3	0.069	0.084	-0.044

(11X3)

=

W_output (new)

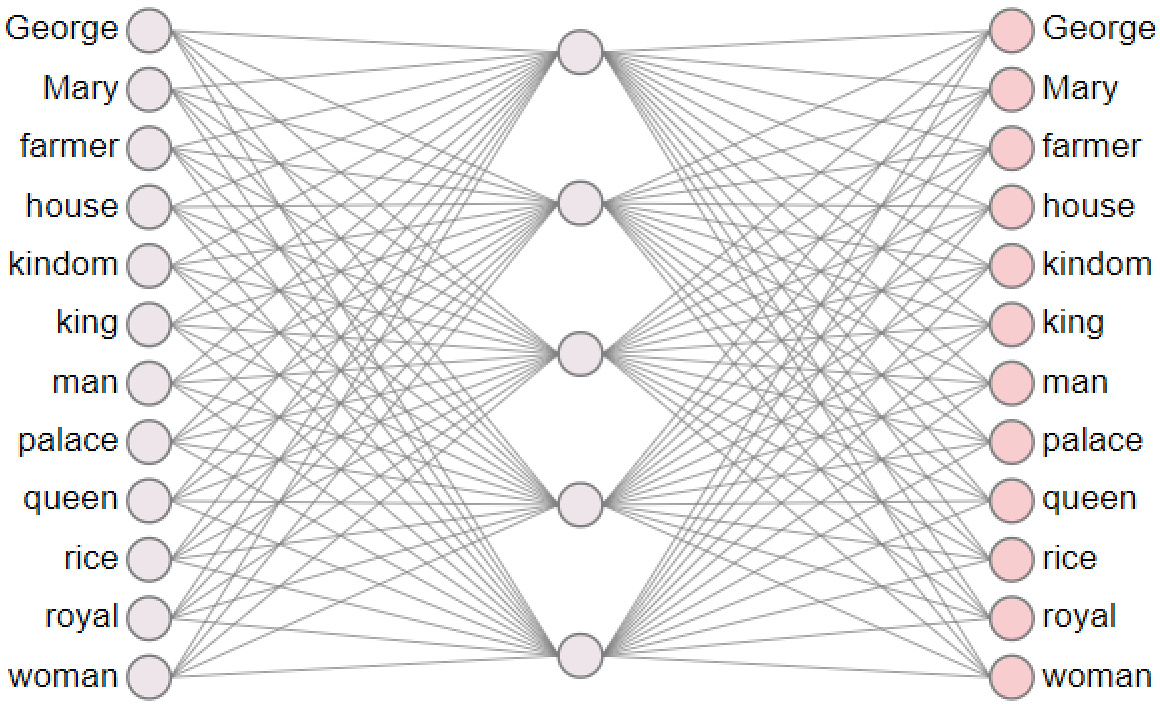
-0.560	0.340	0.160
-0.910	-0.440	1.560
-1.210	-0.130	-1.320
1.670	-0.150	-1.030
1.720	-1.460	0.730
0.000	1.390	-0.120
-0.060	1.520	-0.790
0.798	**1.849**	**-1.672**
-1.366	**1.318**	**-0.477**
0.667	**1.985**	**-1.847**
-1.523	**-1.744**	**-1.858**

(11X3)

https://aegis4048.github.io/optimize_computational_efficiency_of_skip-gram_with_negative_sampling

Application

Application #1: Embedding Pretraining



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

Application

Application #2: 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

Application

Application #2: Translation Rescoring

Lisa gosta de comer Macas e bananas **source**

$\boxed{\text{Lisa}}$ likes to eat apples and bananas *translation#2*

0.2 0.1

4 Deep Learning for NLP

Application

Application #2: 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

Application

Application #2: Translation Rescoring

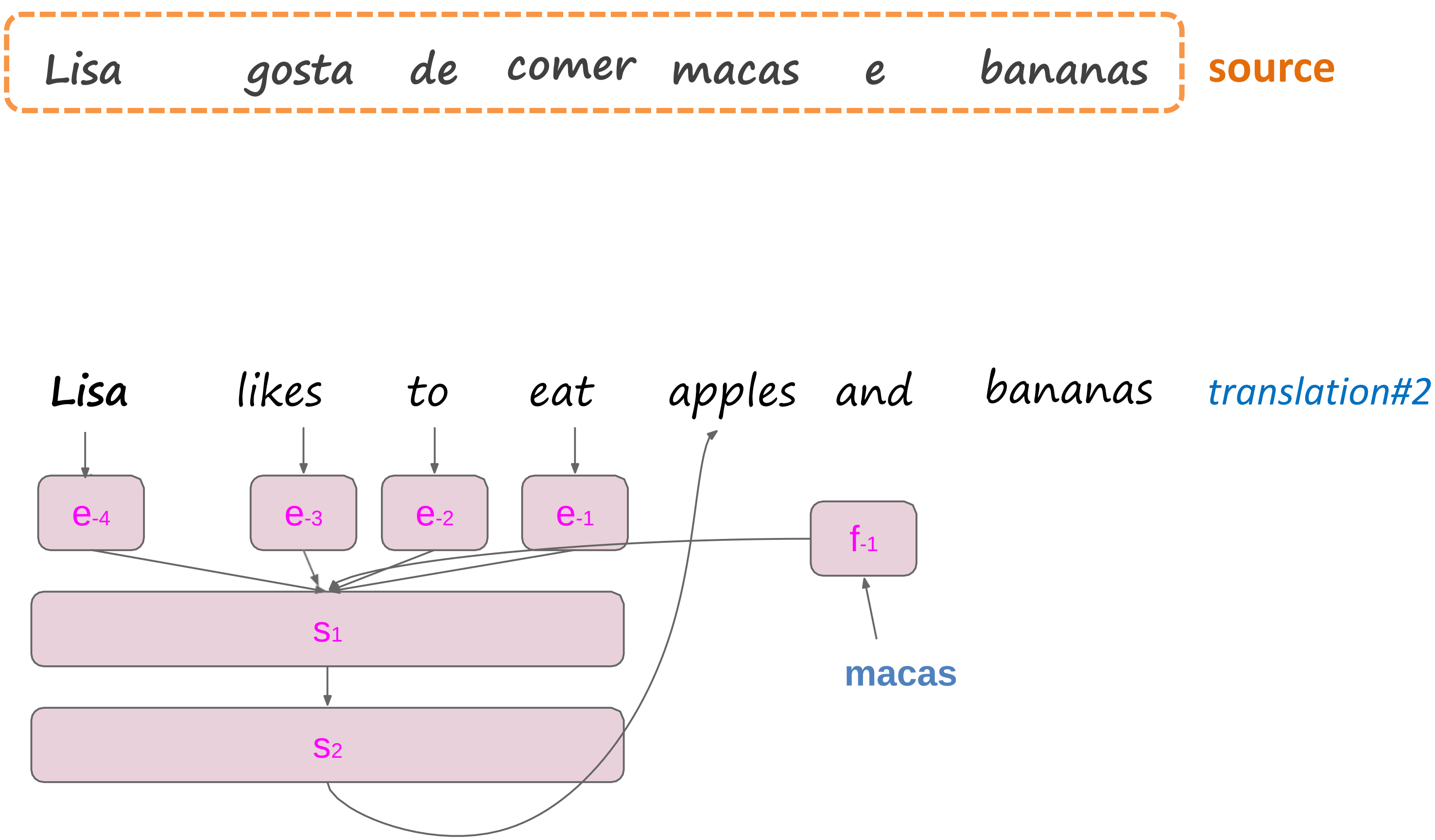
Lisa gosta de comer Macas e bananas source

Lisa likes to eat apples and bananas translation#2

0.2 0.1 0.3

Application

Application #2: Translation Rescoring



4 Deep Learning for NLP

Application

Application #2: 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

0 LECTURE PLAN

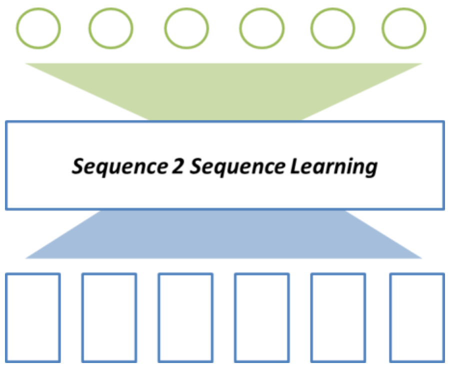
Lecture 3: Word Classification and Machine Learning

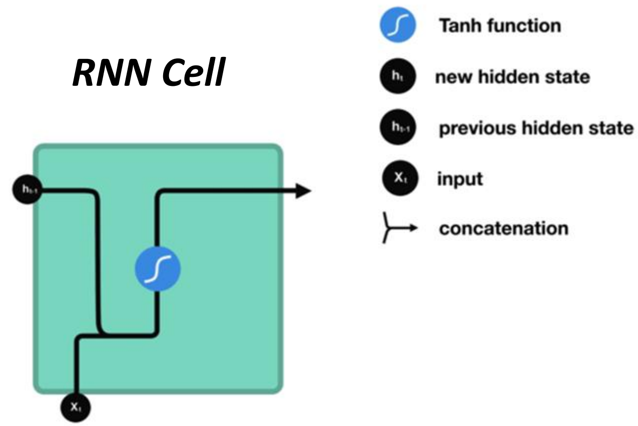
1. Previous Lecture: Word Embedding Review
2. Word Embedding Evaluation
3. Deep Neural Network for Natural Language Processing
    1. Perceptron and Neural Network (NN)
    2. Multilayer Perceptron
    3. Applications
4. **Next Week Preview**

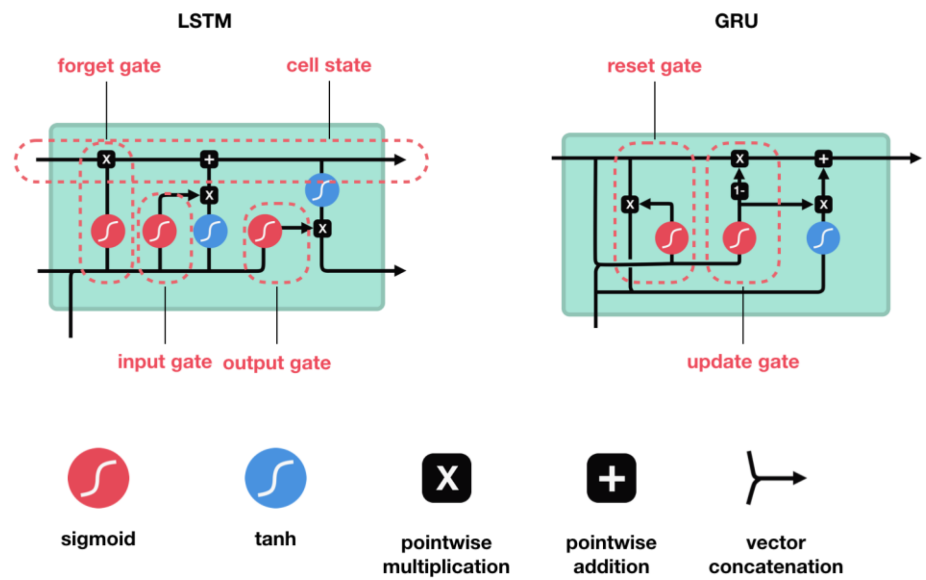
    See how the Deep Learning can be used for NLP

    - Text Classification, etc.

4 Next Week Preview







/ 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