CSCE 689: Special Topics in Advanced NLP

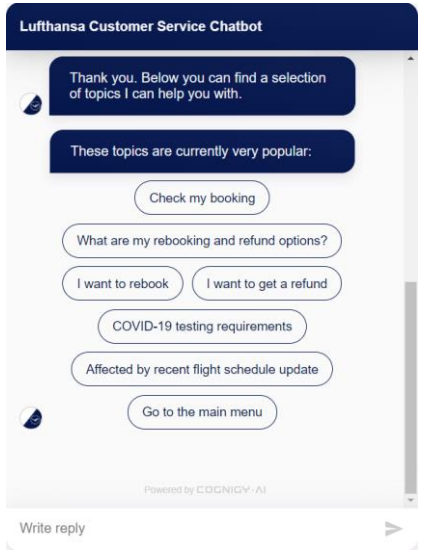
Lecture 2: Machine Learning Basics, Word Representations

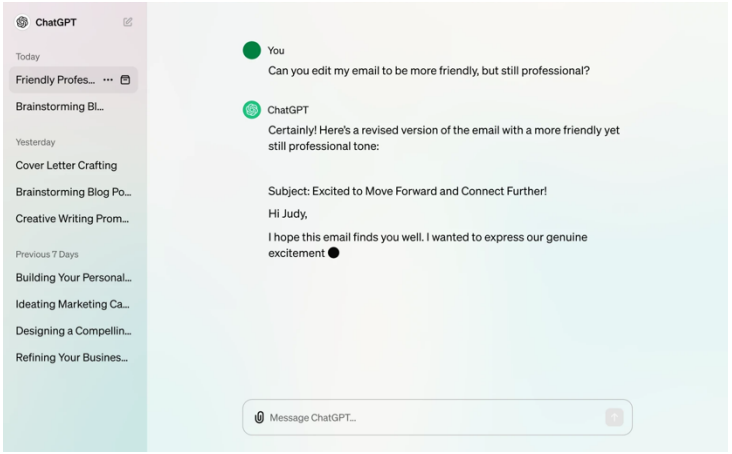
Kuan-Hao Huang

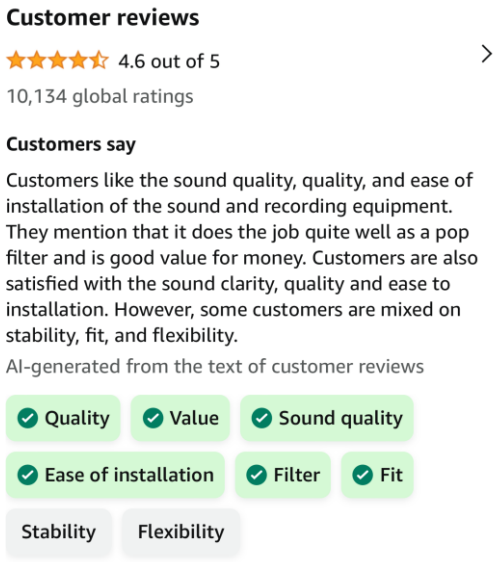
Fall 2026

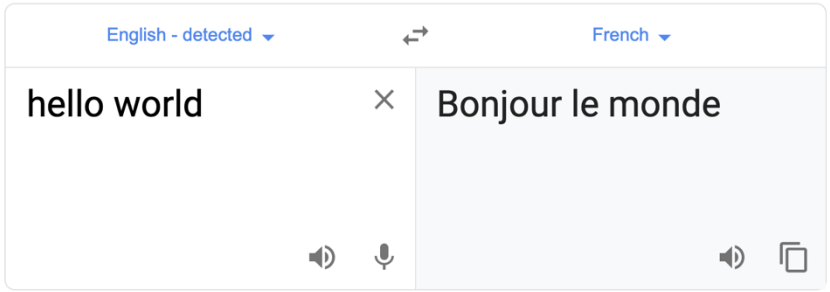
(Some slides adapted from Chris Manning, Dan Jurafsky, Danqi Chen, and Karthik Narasimhan)

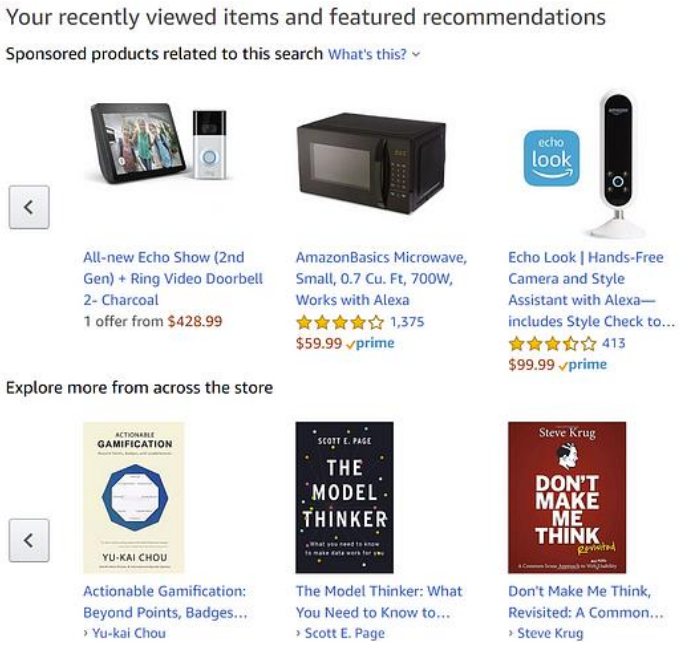
NLP Applications











How to formulate those problems?

Formulation

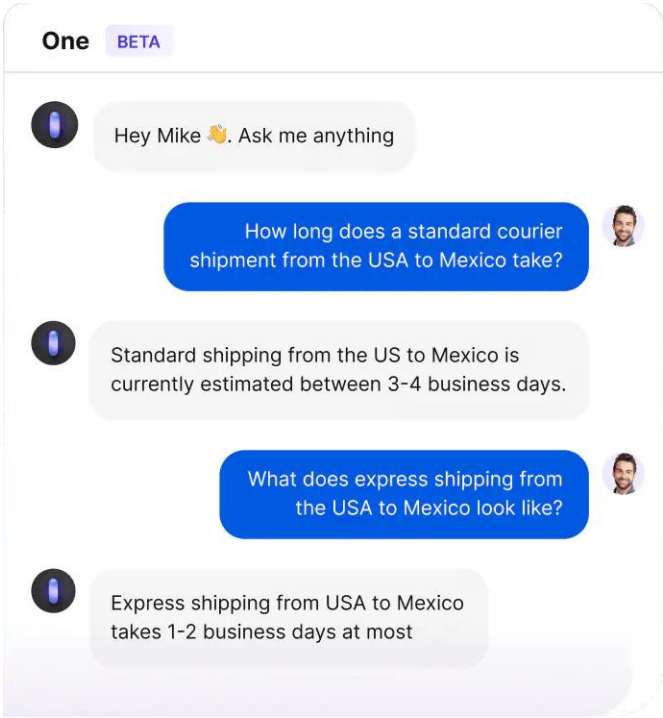
- Build an NLP model to learn the association between input x and output y
- Input x: a sequence of symbols
  - What's the temperature now?
  - I like this restaurant.
- Output y: label
  - Category
  - Structure
  - Text
  - …

Text Classification

- Input x → Output y (category)

$y \in \{pos, neg\}$

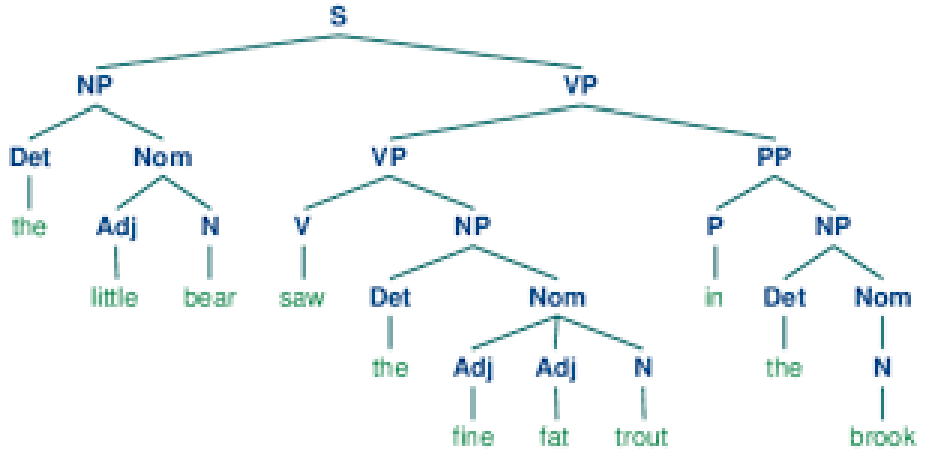
$y \in \{normal, fraud\}$



$y \in \{engineer, business, marketing, IT\ service\}$

Structured Classification

- Input x → Output y (structure)
  - Multiple labels with dependency



Event

Car-Accident	
Location	city hall
Person	foreigner
Age	26
Time	Yesterday

Yesterday, a car accident occurred in front of the city hall, involving a 26-year-old foreigner as the driver. The collision resulted in significant damage to both the vehicles involved and the city hall's facade. Emergency services swiftly responded to the scene and the injured driver was transported to the hospital directly from the site. The extent of the driver's injuries remains undisclosed. Witnesses described the aftermath as chaotic, with visible signs ...

Event

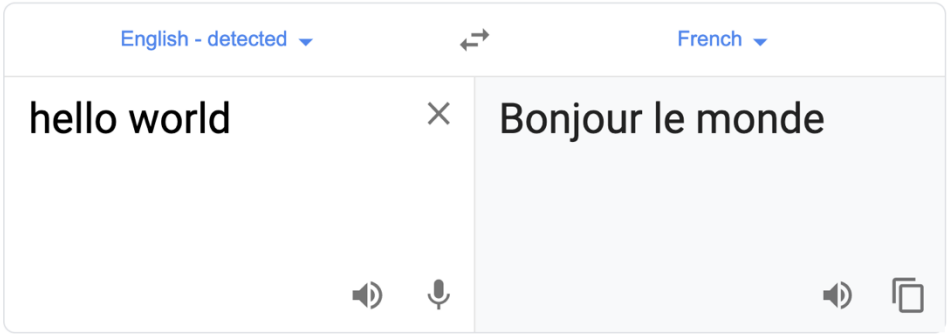
Damage	
Object	vehicles
Object	city hall's facade

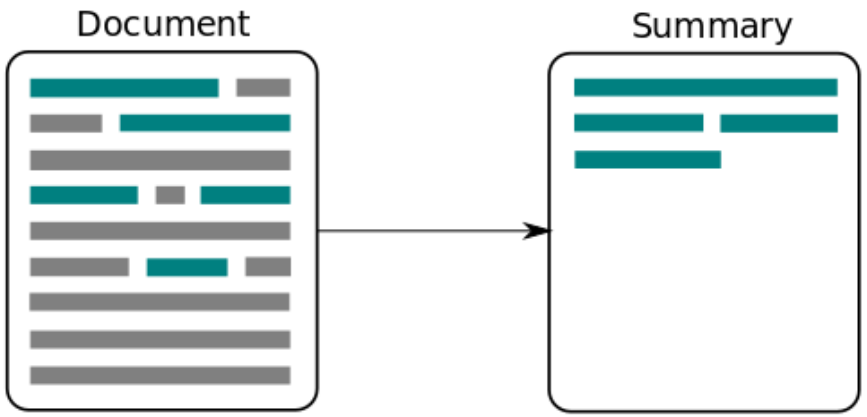
Event

Transport-Person	
Person	injured driver
Origin	city hall
Destination	hospital

Generation

- Input x → Output y (text)
  - Also called sequence-to-sequence tasks





The first recorded travels by Europeans to China and back date from this time. The most famous traveler of the period was the Venetian Marco Polo, whose account of his trip to "Cambaluc," the capital of the Great Khan, and of life there astounded the people of Europe. The account of his travels, Il milione (or, The Million, known in English as the Travels of Marco Polo), appeared about the year 1299. Some argue over the accuracy of Marco Polo's accounts due to the lack of mentioning the Great Wall of China, tea houses, which would have been a prominent sight since Europeans had yet to adopt a tea culture, as well the practice of foot binding by the women in capital of the Great Khan. Some suggest that Marco Polo acquired much of his knowledge through contact with Persian traders since many of the places he named were in Persian.

How did some suspect that Polo learned about China instead of by actually visiting it?

Answer: through contact with Persian traders

Classification vs. Generation

- There is no clear boundary between classification and generation
- Generation = Structured Token Classification

$y \in \{all\ possible\ words\}$ $y \in \{all\ possible\ words\}$

Classification vs. Generation

- There is no clear boundary between classification and generation
- Classification problems can be solved by generation

What's the sentiment of the following text: I very like this restaurant.

The sentiment is positive.

Supervised Learning

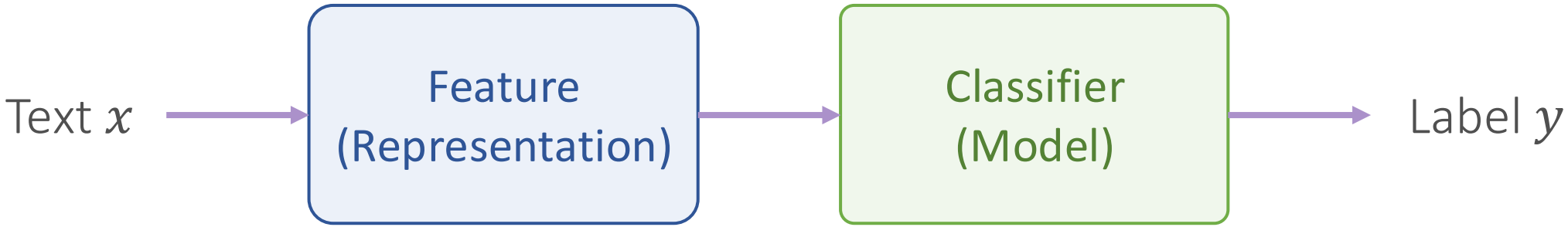
Training Stage

- Training data $\mathcal{D}_{train} = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$
  - Example $x_i \in \mathcal{X}$, label $y_i \in C$
- Train a classifier(model) $f: \mathcal{X} \to C$

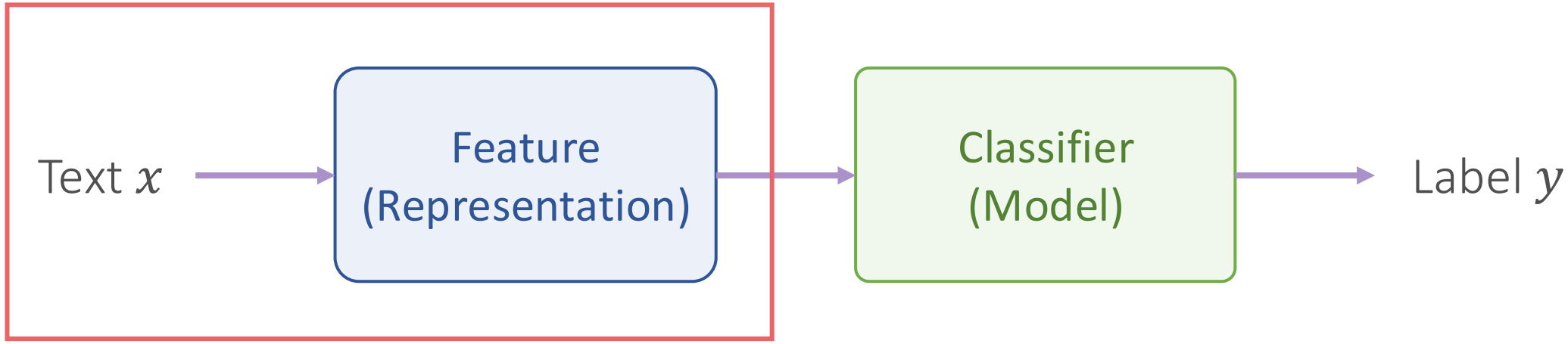
Testing Stage

- Testing data $\mathcal{D}_{test} = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$
- Make predictions $\tilde{y}_i = f(x_i)$
- Evaluate performance $\frac{1}{\mathrm{n}} \sum_i S(y_i, \tilde{y}_i)$

A General Framework for Text Classification



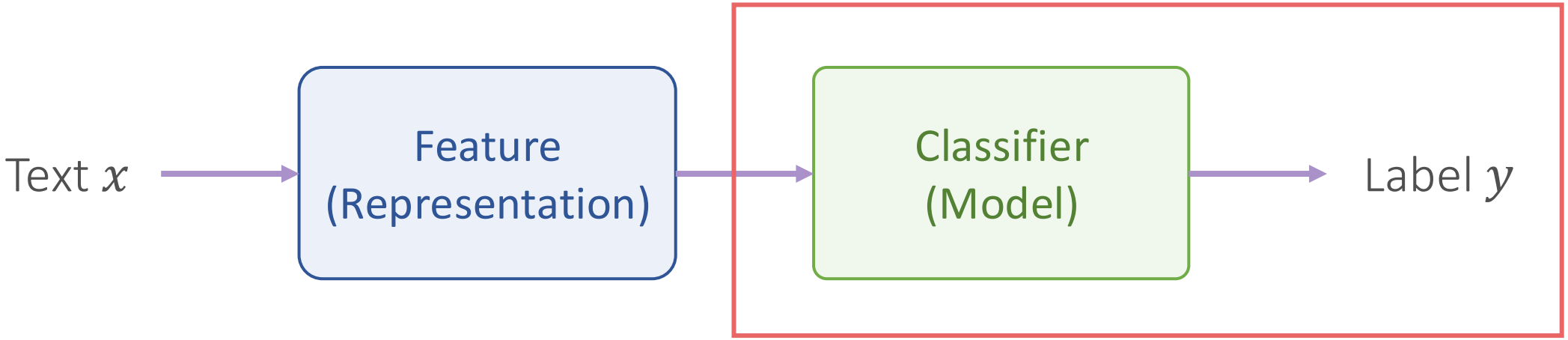
A General Framework for Text Classification



- Teach the model how to understand example x
- Convert the text to a mathematical form
  - The mathematical form captures essential characteristics of the text
- Bag-of-words, n-grams, word embeddings, etc.

We will talk about them later!

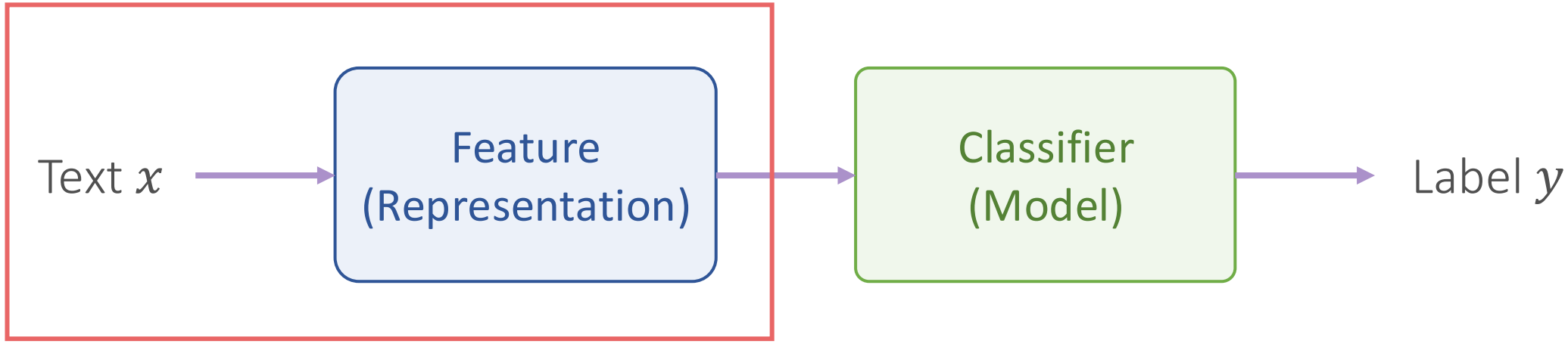
A General Framework for Text Classification



- Teach the model how to make prediction y
- Logistic regression, neural networks, CNN, RNN, LSTM, Transformers

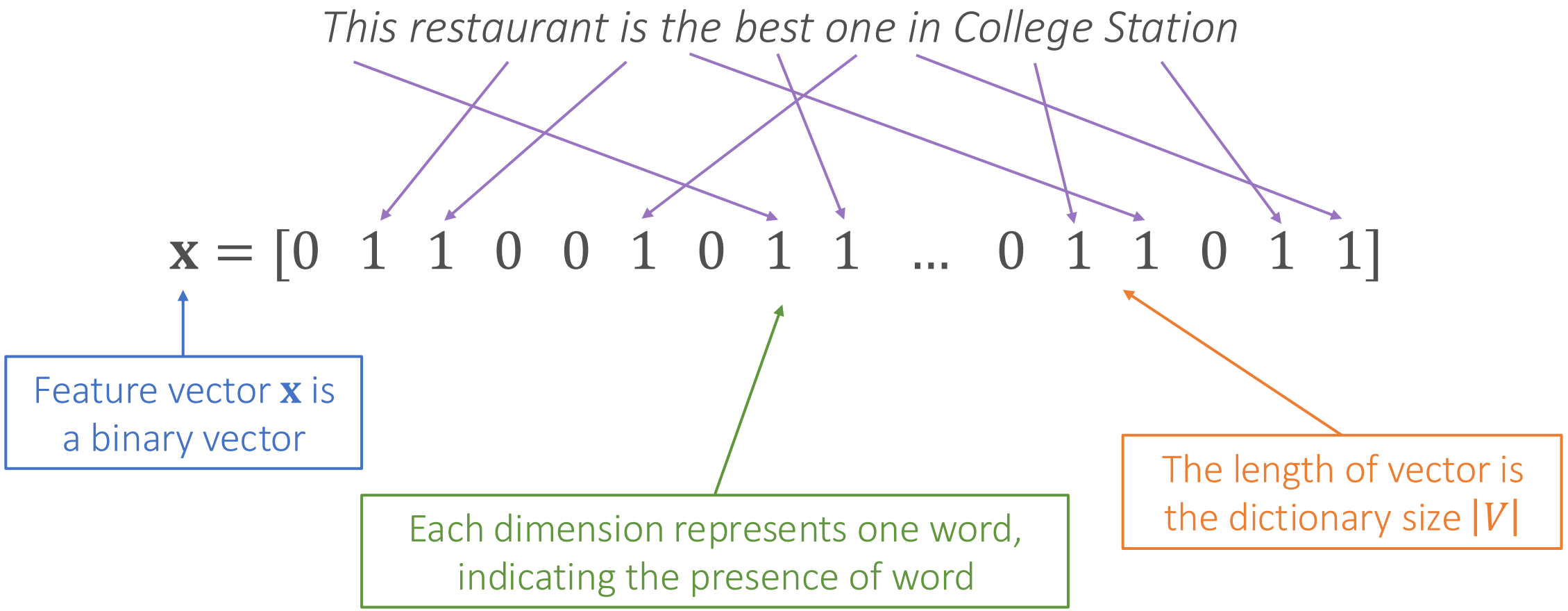
We will talk about them later!

Bag-of-Words (BoW)



- Bag-of-Words (BoW)
  - Consider text as a set of words
- Easy, no effort required

Bag-of-Words (BoW)



Advantages and disadvantages?

Bag-of-Words (BoW)

Bob likes Alice very much

Alice likes Bob very much

They will have the same BoW vector!

$$\mathbf{x} = [0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad \ldots \quad 0 \quad 1]$$

BoW fails to capture sentential structure

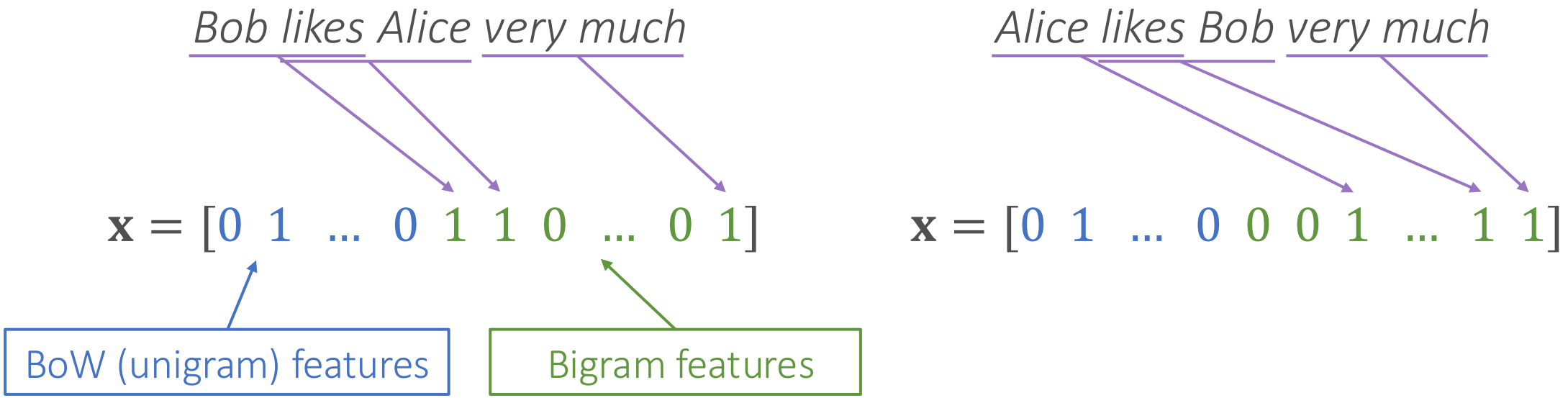
Any solutions?

N-Grams

Bob likes Alice very much

Unigram	*{Bob, likes, Alice, very, much}*
Bigram	*{Bob likes, likes Alice, Alice very, very much}*
Trigram	*{Bob likes Alice, likes Alice very, Alice very much}*
4-gram	*{Bob likes Alice very, likes Alice very much}*

Bag-of-N-Grams



We can consider trigrams, 4-grams, …

N-gram features capture more sentential structure

Other Variants

Binary BoW $\mathbf{x} = [0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad \dots \quad 0 \quad 1]$

Word Count $\mathbf{x} = [0 \quad 2 \quad 1 \quad 0 \quad 0 \quad 4 \quad \dots \quad 0 \quad 3]$

Word Frequency $\mathbf{x} = [0 \quad 0.16 \quad 0.08 \quad 0 \quad 0 \quad 0.32 \quad \dots \quad 0 \quad 0.24]$

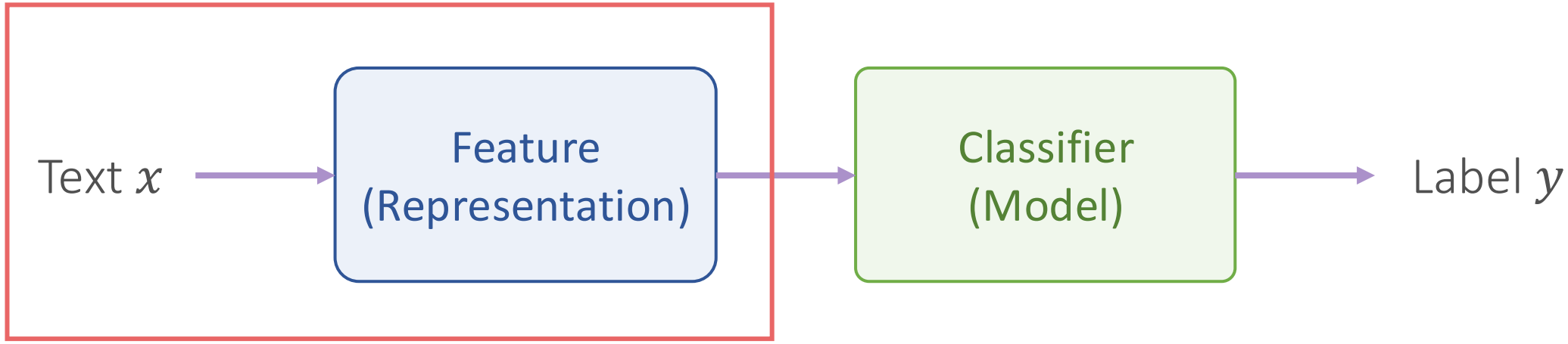
TF-IDF $\mathbf{x} = [0 \quad 0.48 \quad 0.02 \quad 0 \quad 0 \quad 0.15 \quad \dots \quad 0 \quad 0.88]$

$$f_w \cdot \log \frac{N}{n_t}$$

Term Frequency (TF)

Inverse Document Frequency (IDF)

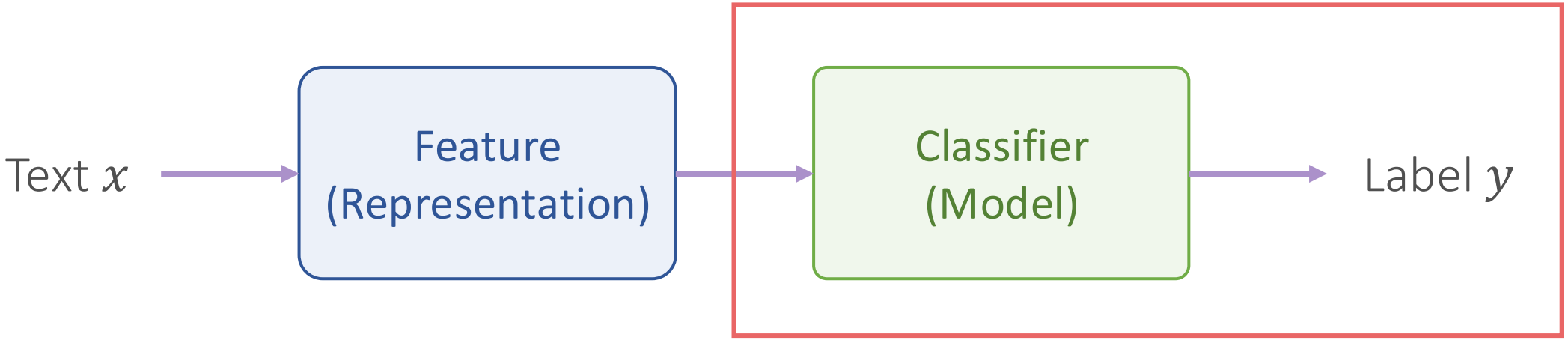
Bag-of-Words and Bag-of-N-Grams



- Bag-of-Words (BoW)
  - A set of words
- Bag-of-N-Grams
  - A set of n-grams

We will discuss “learnable” features later!

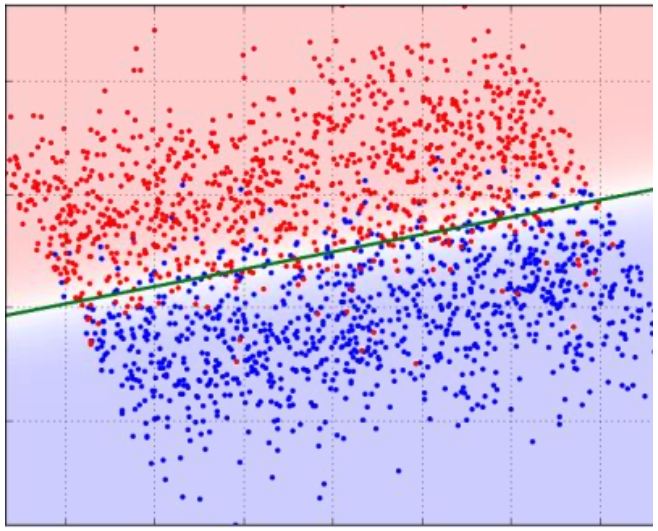
Logistic Regression

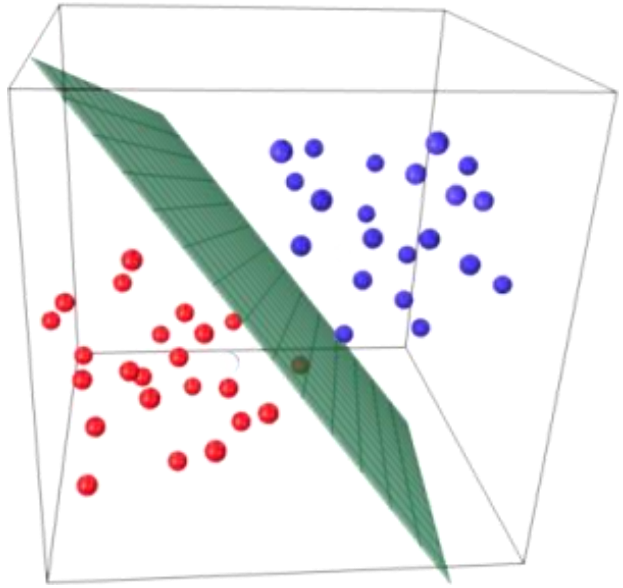


- Logistic regression
  - Find linear weights to map feature vector $\mathbf{x}$ to label y

Logistic Regression

- Let's start from binary classification
  - Input: feature vector $\mathbf{x} = [x_1, x_2, x_3, \dots, x_d]$
  - Output: label $y \in \{0, 1\}$
- Find a linear decision boundary to classify $\mathbf{x}$ into $\{0, 1\}$





https://blog.bigml.com/2016/09/28/logistic-regression-versus-decision-trees/
https://codesachin.wordpress.com/2015/08/16/logistic-regression-for-dummies/

Logistic Regression

Feature Vector $\mathbf{x} = [x_1, x_2, x_3, \ldots, x_d]$ Label $y = 0 \; or \; 1$

Weight Vector $\mathbf{w} = [w_1, w_2, w_3, \ldots, w_d]$ Bias b

Learnable parameters

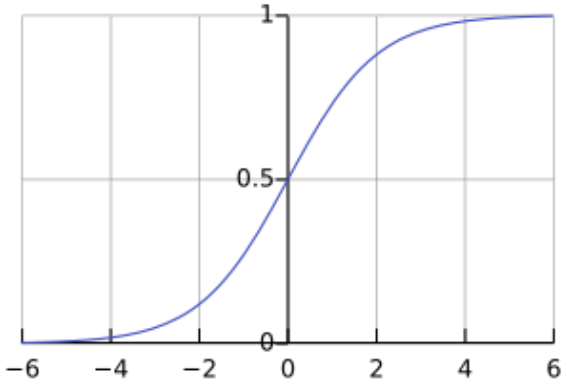
$$z = \mathbf{w} \cdot \mathbf{x} + b$$

$$\tilde{y} = P(y = 1 | \mathbf{x}) = \sigma(z)$$

Convert to probability

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

Sigmoid Function



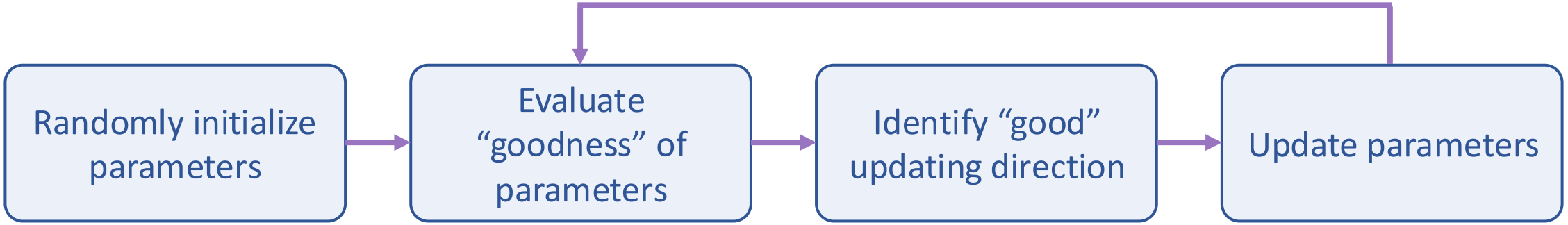
Decision boundary: $= \begin{cases} 1 & \text{If } \tilde{y} \geq 0.5 \\ 0 & \text{If } \tilde{y} < 0.5 \end{cases}$

How to Find The Best Parameters?

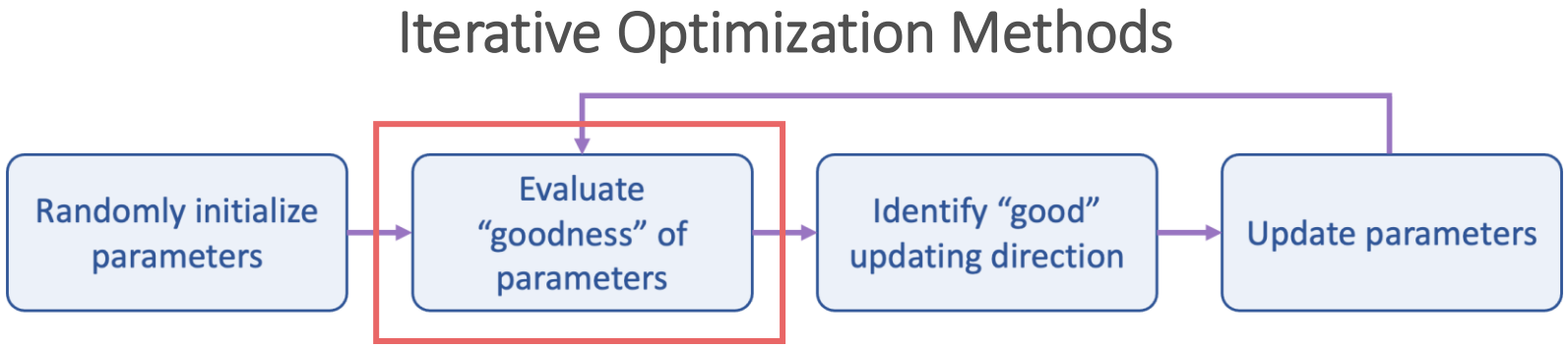
Weight Vector $\mathbf{w} = [w_1, w_2, w_3, \ldots, w_d]$ Bias b

Learnable parameters

Iterative Optimization Methods



Loss Function



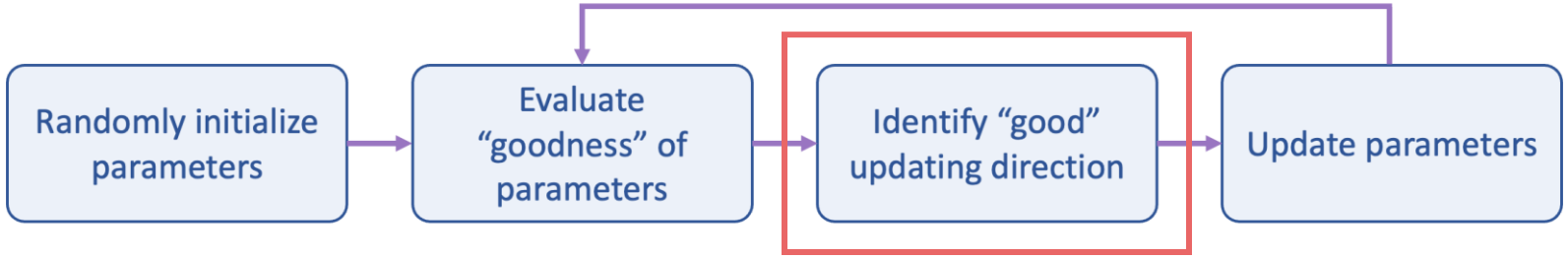
- Training data $\mathcal{D}_{train} = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$
- Output labels probabilities $\tilde{y}_1, \tilde{y}_2, \dots, \tilde{y}_m$

Cross Entropy Loss

$$\mathcal{L}_{total} = -\frac{1}{m}\sum_i \mathcal{L}_{CE}(y_i, \tilde{y}_i) = -\frac{1}{m}\sum_i [y_i \log \tilde{y}_i + (1 - y_i)\log(1 - \tilde{y}_i)]$$

Optimization Objective

Iterative Optimization Methods



Cross Entropy Loss

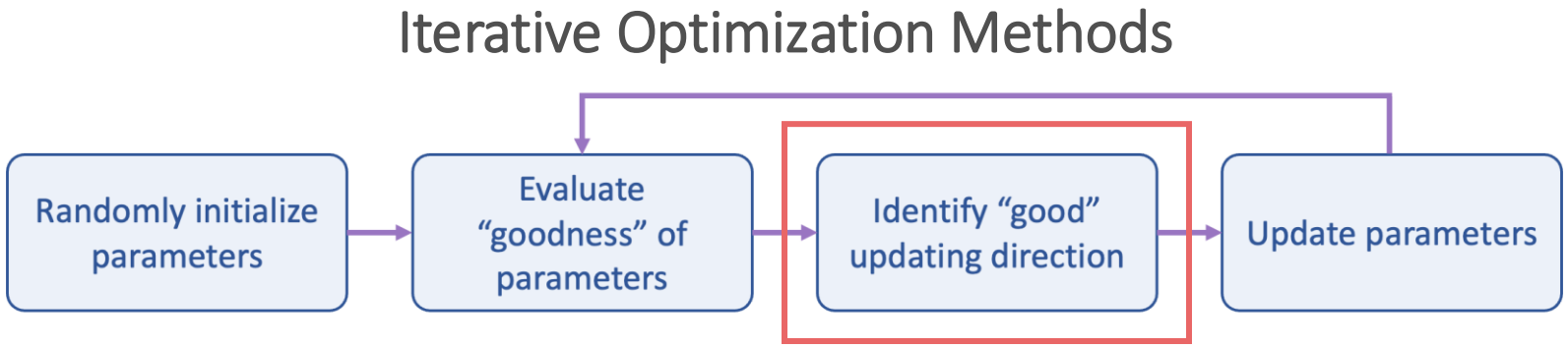
$$\mathcal{L}_{total} = -\frac{1}{m}\sum_{i}\mathcal{L}_{CE}(y_i, \tilde{y}_i)$$

Parameters $\theta =$ Weight Vector $\mathbf{w} = [w_1, w_2, w_3, \ldots, w_d]$ Bias b

$$[\mathbf{w}^*; b^*] = \theta^* = \arg\min_{\theta} \mathcal{L}_{total}$$

Optimization objective

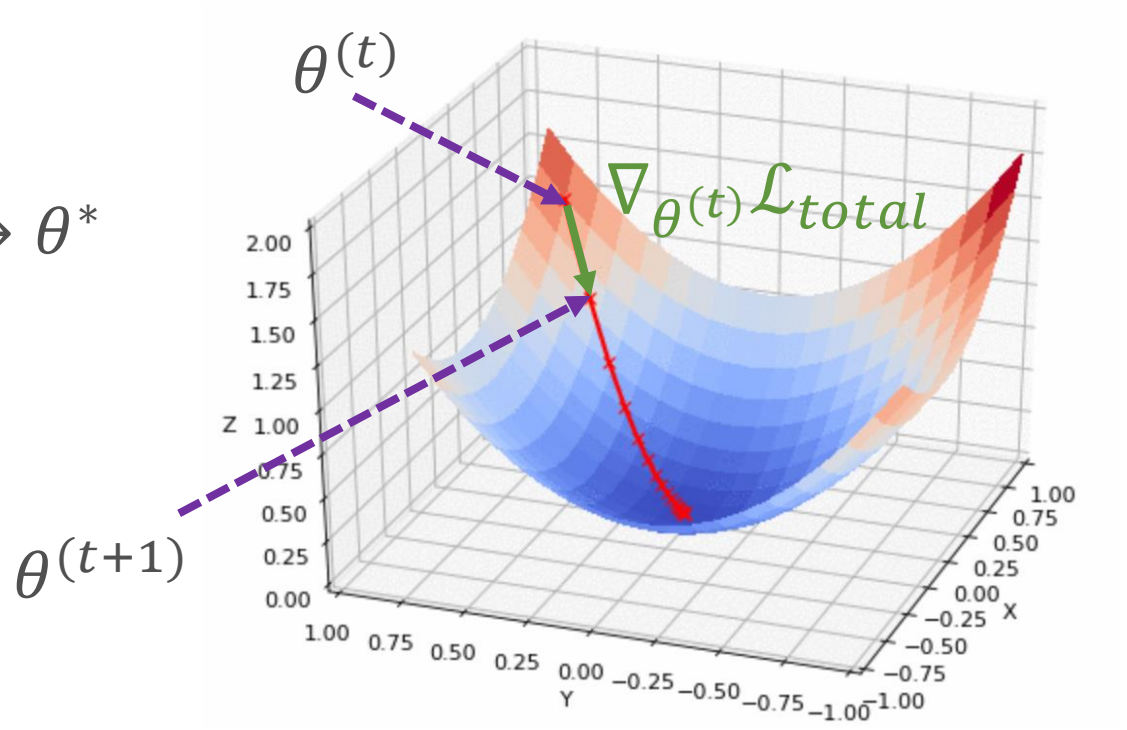
Gradient



$$\theta^* = \arg\min_{\theta} \mathcal{L}_{total}$$

$$\theta^{(0)} \rightarrow \theta^{(1)} \rightarrow \theta^{(2)} \rightarrow \cdots \rightarrow \theta^{(k)} \rightarrow \cdots \rightarrow \theta^*$$

$\nabla_{\theta^{(t)}} \mathcal{L}_{total}$ is a "good" direction to minimize the objective



https://www.digitalocean.com/community/tutorials/intro-to-optimization-in-deep-learning-gradient-descent

Gradient

$$\nabla_\theta \mathcal{L}_{total} \qquad \frac{\partial \mathcal{L}_{total}}{\partial \mathbf{w}} \qquad \frac{\partial \mathcal{L}_{total}}{\partial b}$$

$$\frac{\partial \mathcal{L}_{total}}{\partial \mathbf{w}_j} = \frac{\partial \left(-\frac{1}{m} \sum_i [y_i \log \tilde{y}_i + (1 - y_i) \log(1 - \tilde{y}_i)] \right)}{\partial \mathbf{w}_j}$$

$$= \frac{\partial \left(-\frac{1}{m} \sum_i [y_i \log \sigma(z_i) + (1 - y_i) \log(1 - \sigma(z_i))] \right)}{\partial \mathbf{w}_j}$$

$$= -\frac{1}{m} \sum_i \left[y_i \frac{\partial \log \sigma(z_i)}{\partial \mathbf{w}_j} + (1 - y_i) \frac{\partial \log(1 - \sigma(z_i))}{\partial \mathbf{w}_j} \right]$$

$$\tilde{y}_i = \sigma(z_i)$$
$$z_i = \mathbf{w} \cdot \mathbf{x}_i + b$$

Gradient

$$\frac{\partial \mathcal{L}_{total}}{\partial \mathbf{w}_j} = -\frac{1}{m}\sum_i \left[y_i \frac{\partial \log \sigma(z_i)}{\partial \mathbf{w}_j} + (1-y_i)\frac{\partial \log(1-\sigma(z_i))}{\partial \mathbf{w}_j} \right]$$

$$\frac{\partial \log \sigma(z_i)}{\partial \mathbf{w}_j} = \frac{1}{\sigma(z_i)} \cdot \left[\sigma(z_i)\big(1-\sigma(z_i)\big)\right] \cdot \mathbf{x}_{i,j} = \big(1-\sigma(z_i)\big)\mathbf{x}_{i,j}$$

$$\sigma'(z) = \sigma(z)(1-\sigma(z))$$

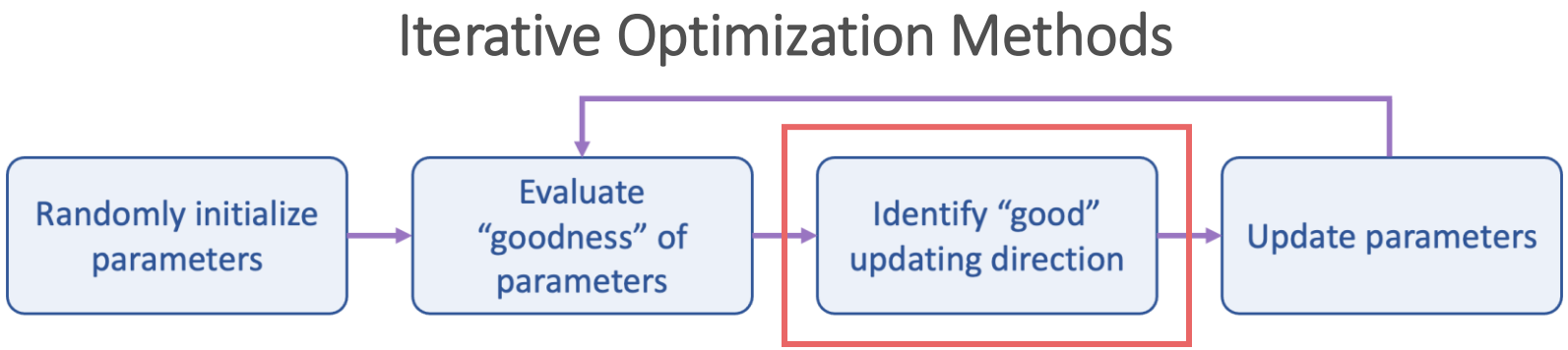
$$\frac{\partial \log(1-\sigma(z_i))}{\partial \mathbf{w}_j} = \frac{1}{1-\sigma(z_i)} \cdot \left[-\sigma(z_i)\big(1-\sigma(z_i)\big)\right] \cdot \mathbf{x}_{i,j} = -\sigma(z_i)\mathbf{x}_{i,j}$$

$$\big(1-\sigma(z)\big)' = -\sigma(z)(1-\sigma(z))$$

$$\frac{\partial \mathcal{L}_{total}}{\partial \mathbf{w}_j} = -\frac{1}{m}\sum_i \left[y_i\big(1-\sigma(z_i)\big)\mathbf{x}_{i,j} + (1-y_i)\big(-\sigma(z_i)\mathbf{x}_{i,j}\big) \right]$$

$$= -\frac{1}{m}\sum_i \big(y_i - \sigma(z_i)\big)\mathbf{x}_{i,j} = \frac{1}{m}\sum_i (\tilde{y}_i - y_i)\mathbf{x}_{i,j}$$

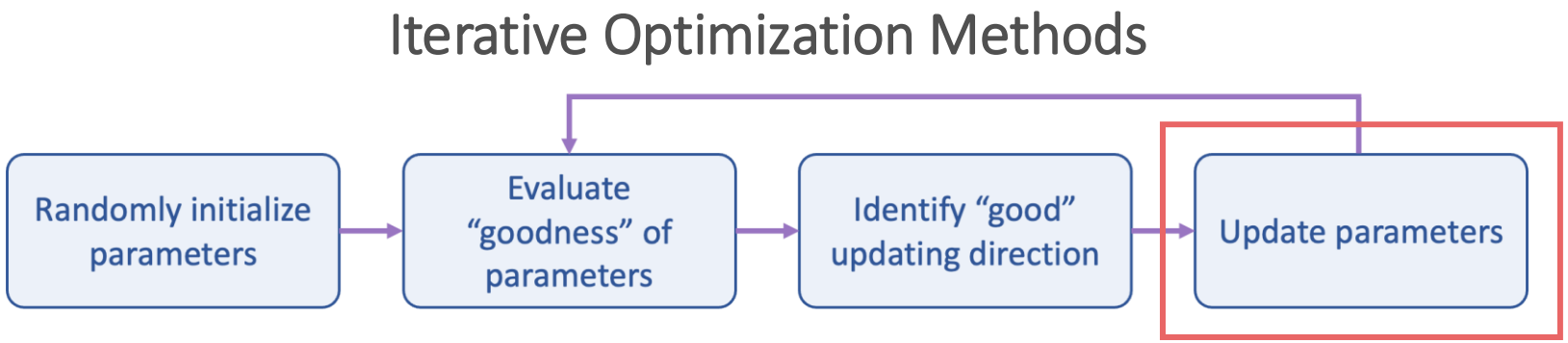
Gradient



$$\frac{\partial \mathcal{L}_{total}}{\partial \mathbf{w}} = \sum_{i=1}^{m} (\tilde{y}_i - y_i)\mathbf{x}_i$$

$$\frac{\partial \mathcal{L}_{total}}{\partial b} = \sum_{i=1}^{m} (\tilde{y}_i - y_i)$$

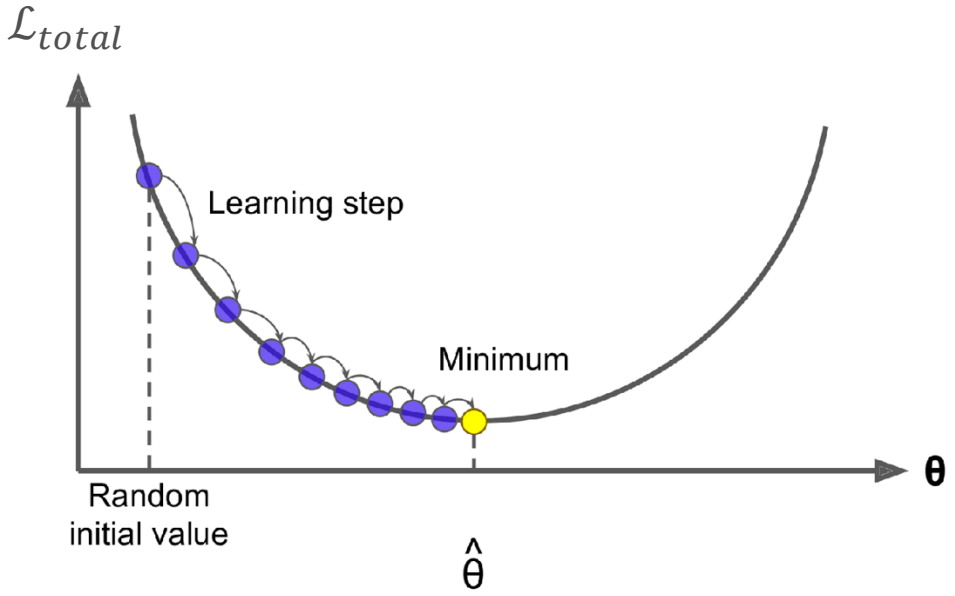
Gradient Descent



$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \, \nabla_{\mathbf{w}} \mathcal{L}_{total}$$

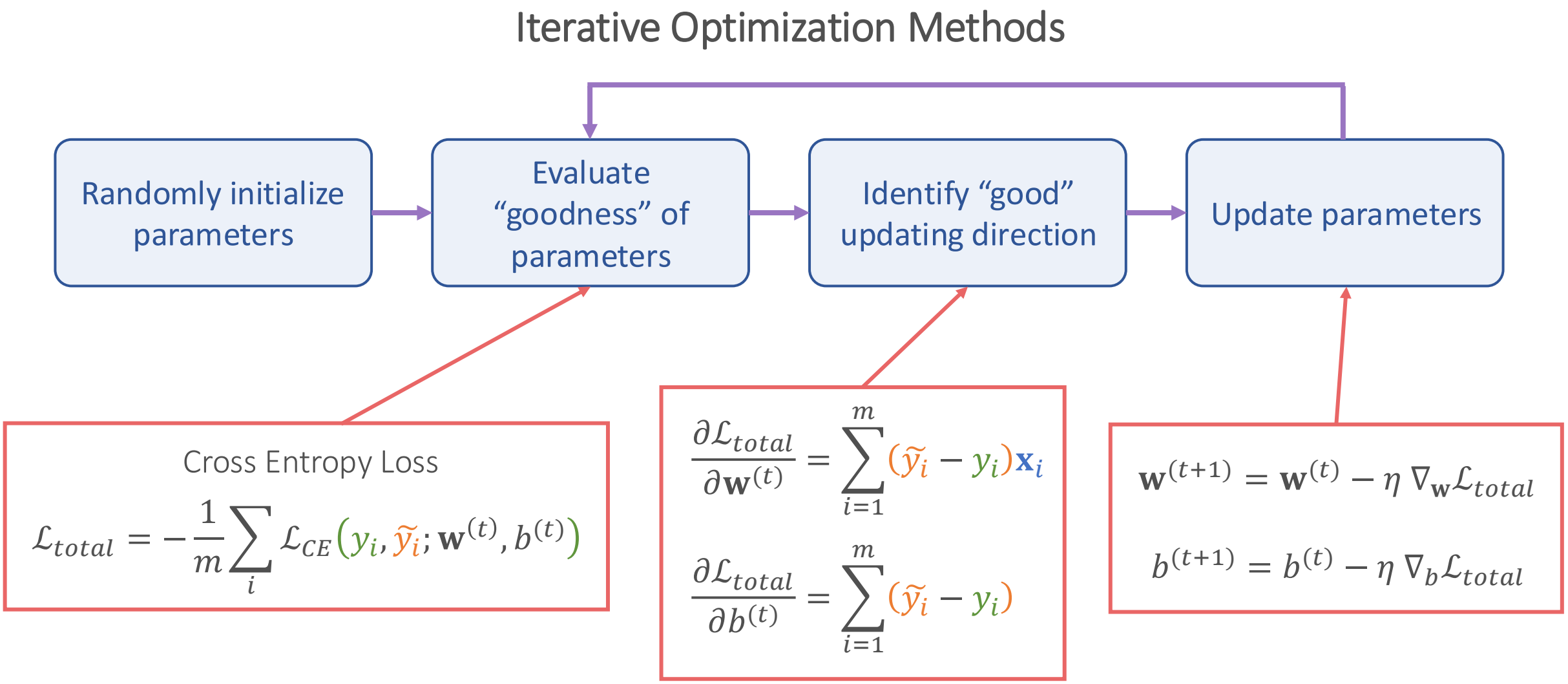
$$b^{(t+1)} = b^{(t)} - \eta \, \nabla_{b} \mathcal{L}_{total}$$

Learning step



https://insightfultscript.com/collections/programming/machine-learning/sgd/

Training Process



From Binary to Multiclass Classification

- Logistic Regression for binary classification

Feature Vector $\mathbf{x} = [x_1, x_2, x_3, \dots, x_d]$ Label $y = 0 \; or \; 1$

Weight Vector $\mathbf{w} = [w_1, w_2, w_3, \dots, w_d]$ Bias b

Learnable Parameters

$$z = \mathbf{w} \cdot \mathbf{x} + b$$

$$P(y = 1 | \mathbf{x}) = \sigma(z)$$

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

Sigmoid Function

$$\text{Prediction} = \begin{cases} 1 & \text{If } P(y = 1 | \mathbf{x}) \geq 0.5 \\ 0 & \text{If } P(y = 1 | \mathbf{x}) < 0.5 \end{cases}$$

From Binary to Multiclass Classification

- Logistic Regression for multiclass classification

Feature Vector $\mathbf{x} = [x_1, x_2, x_3, \dots, x_d]$ Label $y = 0,1, \dots, C-1$

Weight Vectors $\mathbf{w}_c = [w_{c,1}, w_{c,2}, w_{c,3}, \dots, w_{c,d}]$ Bias b_c

Learnable Parameters

$$z_c = \mathbf{w}_c \cdot \mathbf{x} + b_c$$

$$P(y = c|\ \mathbf{x}) = \text{softmax}(z_c)$$

$$\text{softmax}(z_c) = \frac{e^{z_c}}{\sum_t e^{z_t}}$$

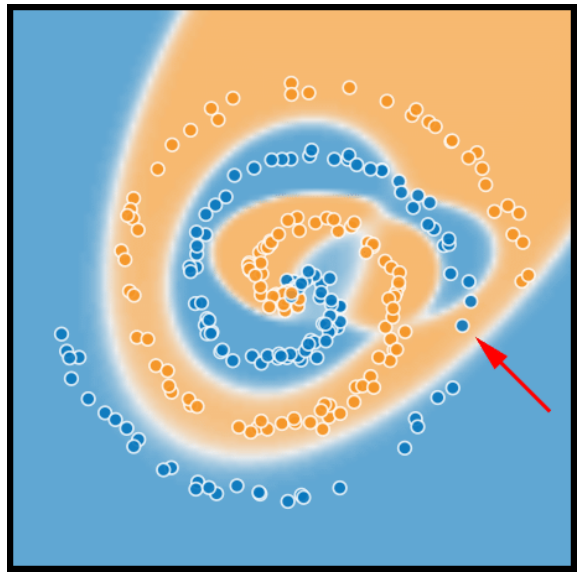
Softmax Function

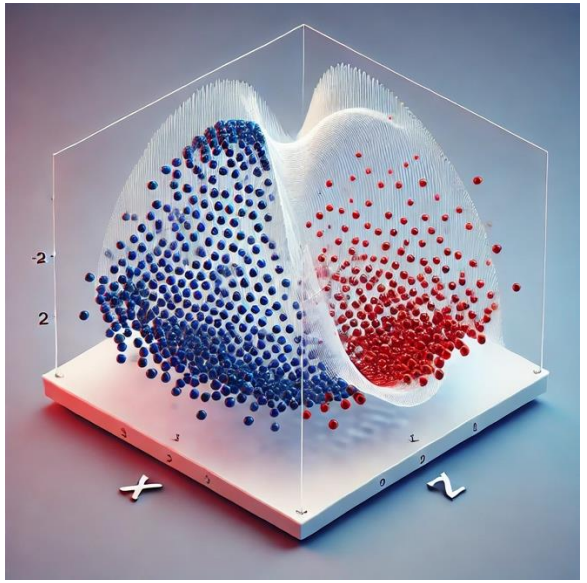
$$\text{Prediction} = \arg\max_{\text{c}} P(y = c|\ \mathbf{x})$$

Logistic Regression

- Logistic regression
  - Find linear weights to map feature vector $\mathbf{x}$ to label y

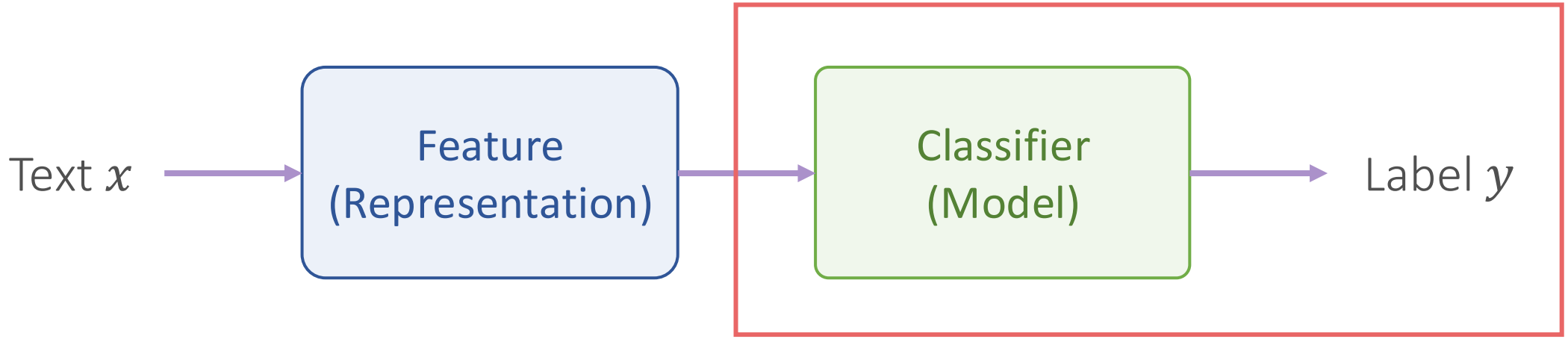
What if linear weights are not powerful enough?





https://www.researchgate.net/publication/344636757_CQNN_Convolutional_Quadratic_Neural_Networks
https://chatgpt.com/

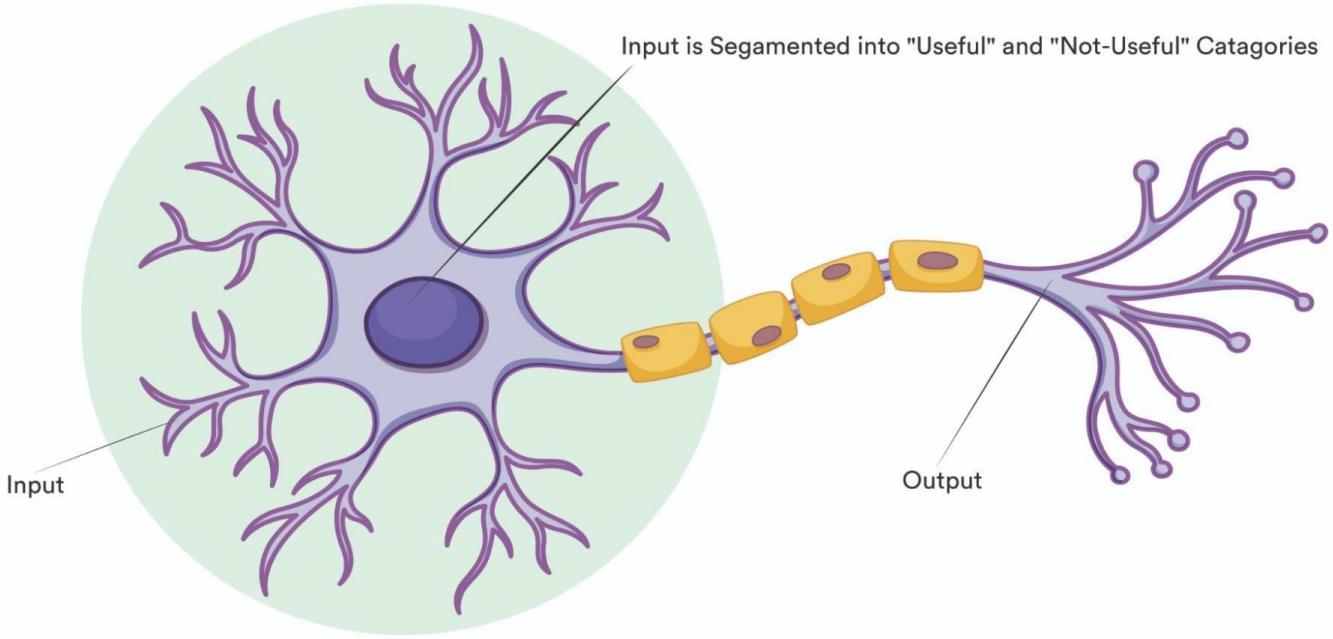
Neural Networks



- Neural Networks
  - Find a non-linear decision boundary to map feature vector $\mathbf{x}$ to label y

Biological Neurons

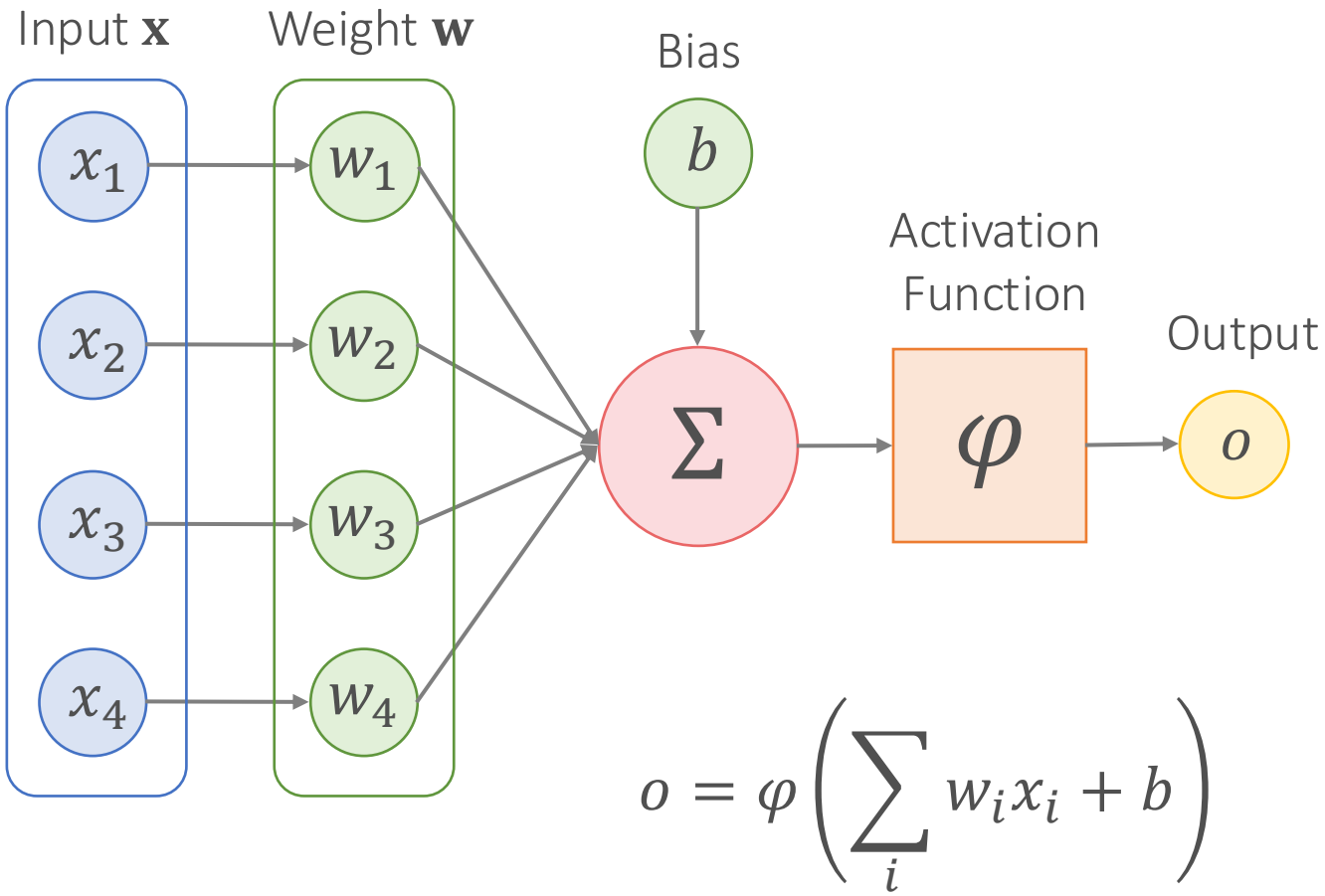
Neuron activation: A neuron becomes active to transmit information when it receives sufficient input from other neurons



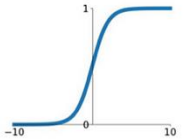
https://www.kdnuggets.com/2022/06/activation-functions-work-deep-learning.html

Neurons in Neural Networks

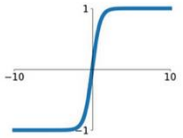
Mimic the behavior of neurons to transmit information



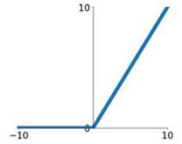
Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$



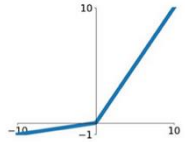
tanh
$\tanh(x)$



ReLU
$\max(0, x)$

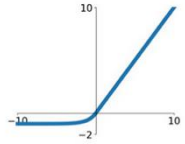


Leaky ReLU
$\max(0.1x, x)$



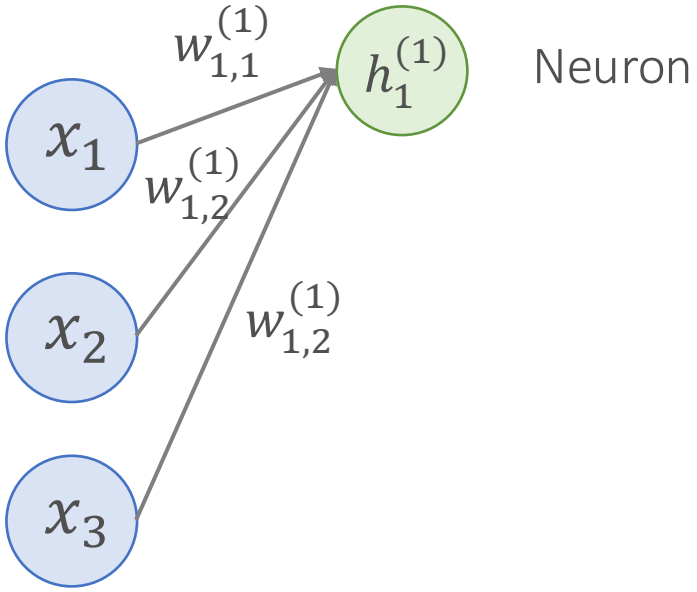
Maxout
$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



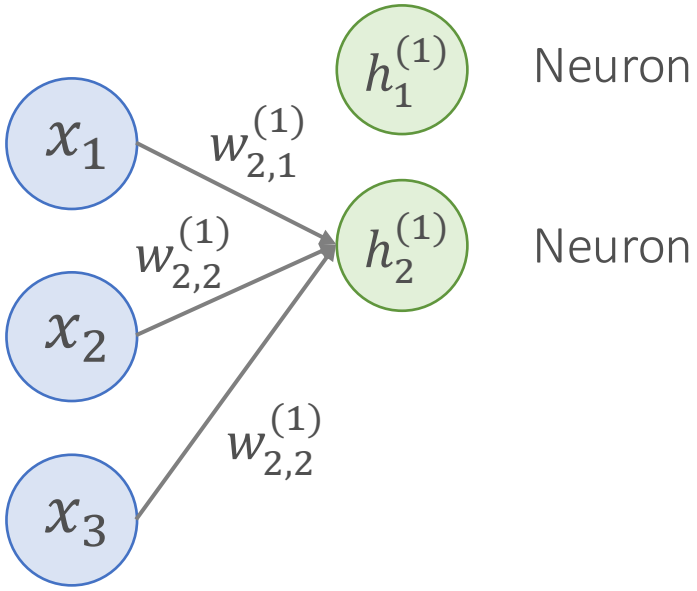
https://blog.devops.dev/exploring-activation-functions-in-deep-learning-properties-derivatives-and-impact-on-model-7585aad8a757

Multilayer Perceptron (MLP)



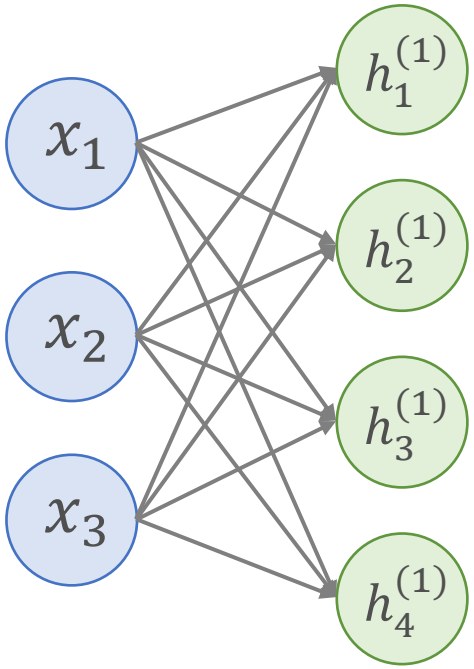
$$h_1^{(1)} = \varphi\left(\sum_i w_{1,i}^{(1)} x_i + b\right) = \varphi\left(\mathbf{w}_1^{(1)} \cdot \mathbf{x} + b\right)$$

Multilayer Perceptron (MLP)



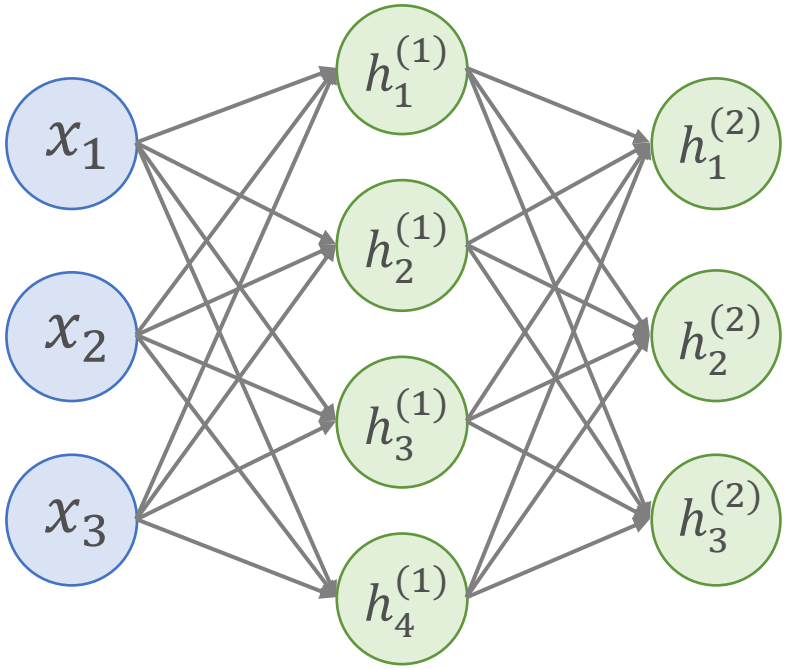
$$h_2^{(1)} = \varphi\left(\sum_i w_{2,i}^{(1)} x_i + b\right) = \varphi\left(\mathbf{w}_2^{(1)} \cdot \mathbf{x} + b\right)$$

Multilayer Perceptron (MLP)



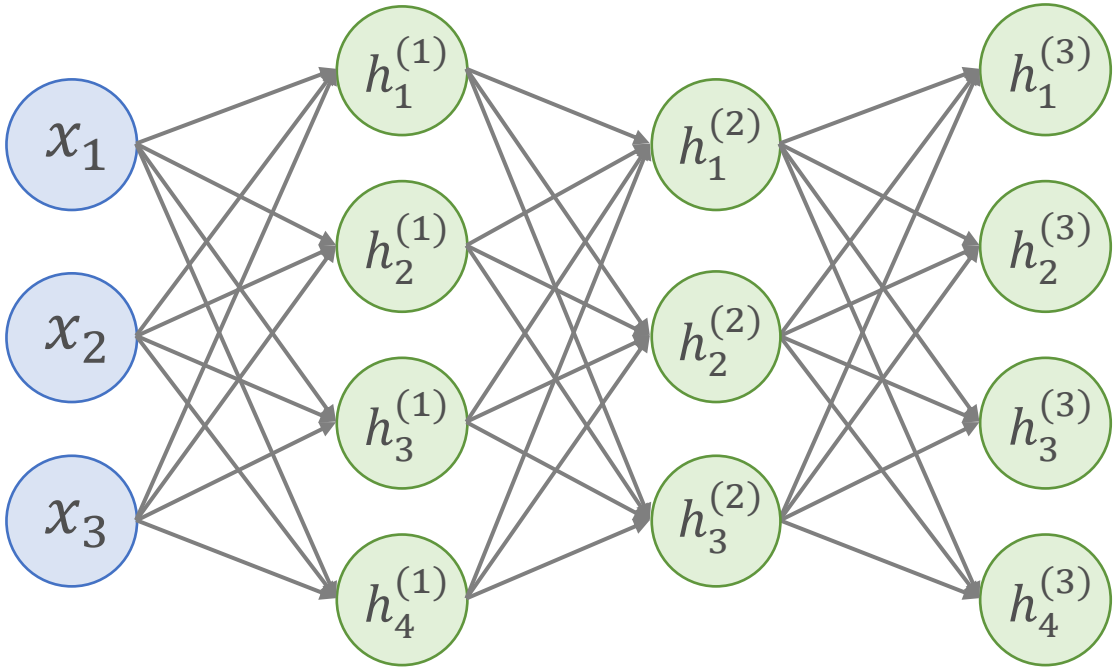
$$\mathbf{h}^{(1)} = \varphi\left(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}\right)$$

Multilayer Perceptron (MLP)



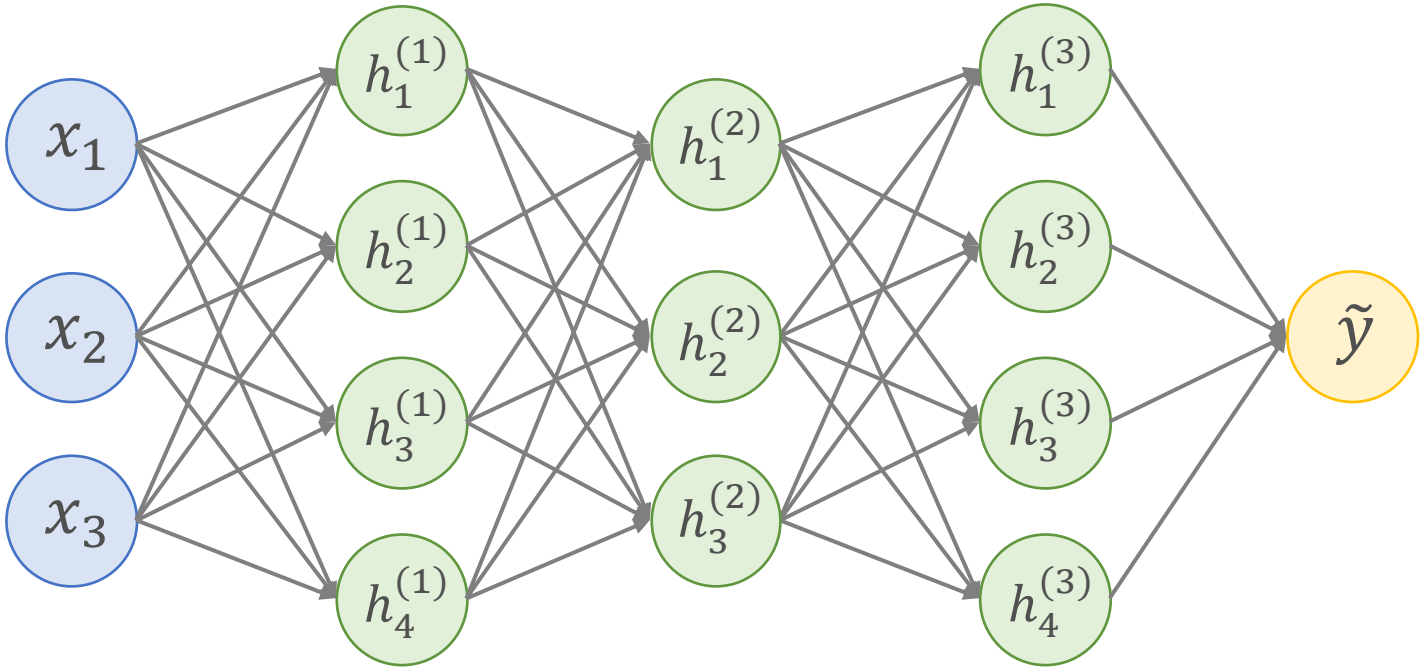
$$\mathbf{h}^{(2)} = \varphi\left(\mathbf{W}^{(2)}\mathbf{h}^{(1)} + \mathbf{b}^{(2)}\right)$$

Multilayer Perceptron (MLP)



$$\mathbf{h}^{(3)} = \varphi\left(\mathbf{W}^{(3)}\mathbf{h}^{(2)} + \mathbf{b}^{(3)}\right)$$

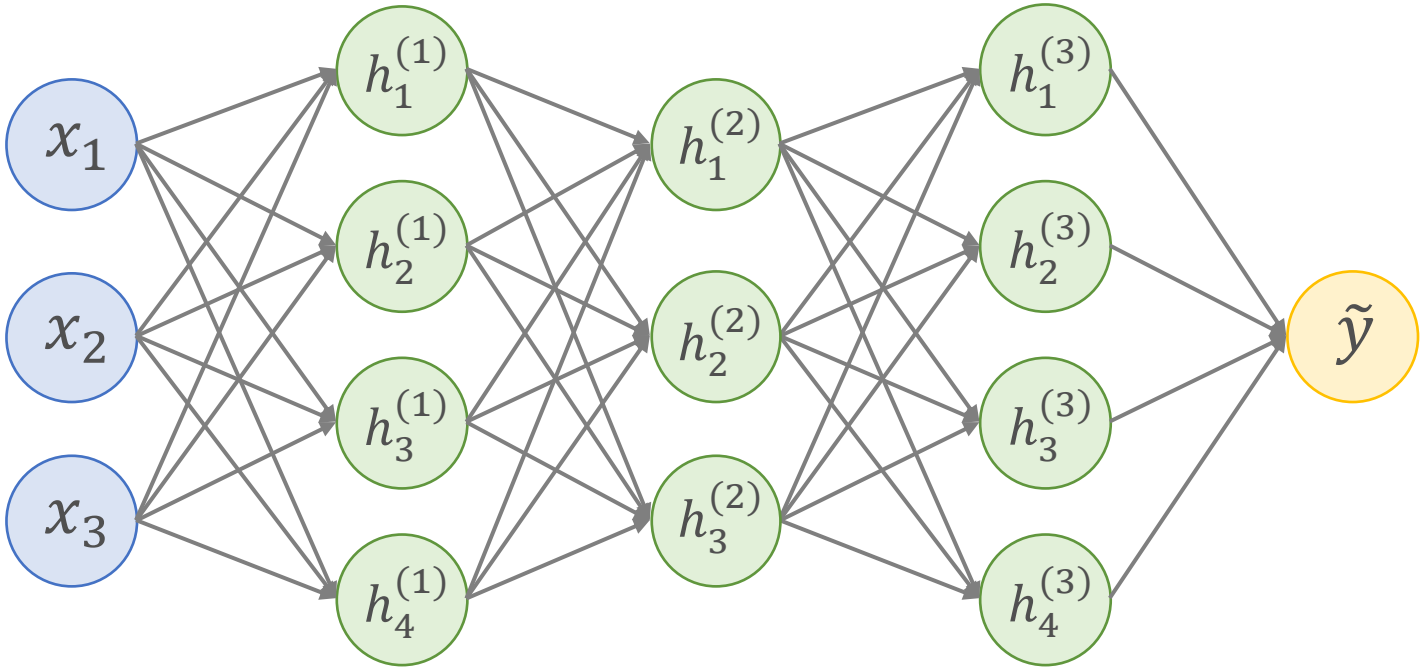
Multilayer Perceptron (MLP)



Decision boundary: $= \begin{cases} 1 & \text{If } \tilde{y} \geq 0.5 \\ 0 & \text{If } \tilde{y} < 0.5 \end{cases}$

$$\tilde{y} = \sigma\left(\mathbf{W}^{(o)}\mathbf{h}^{(3)} + \mathbf{b}^{(o)}\right)$$

Optimization Objective



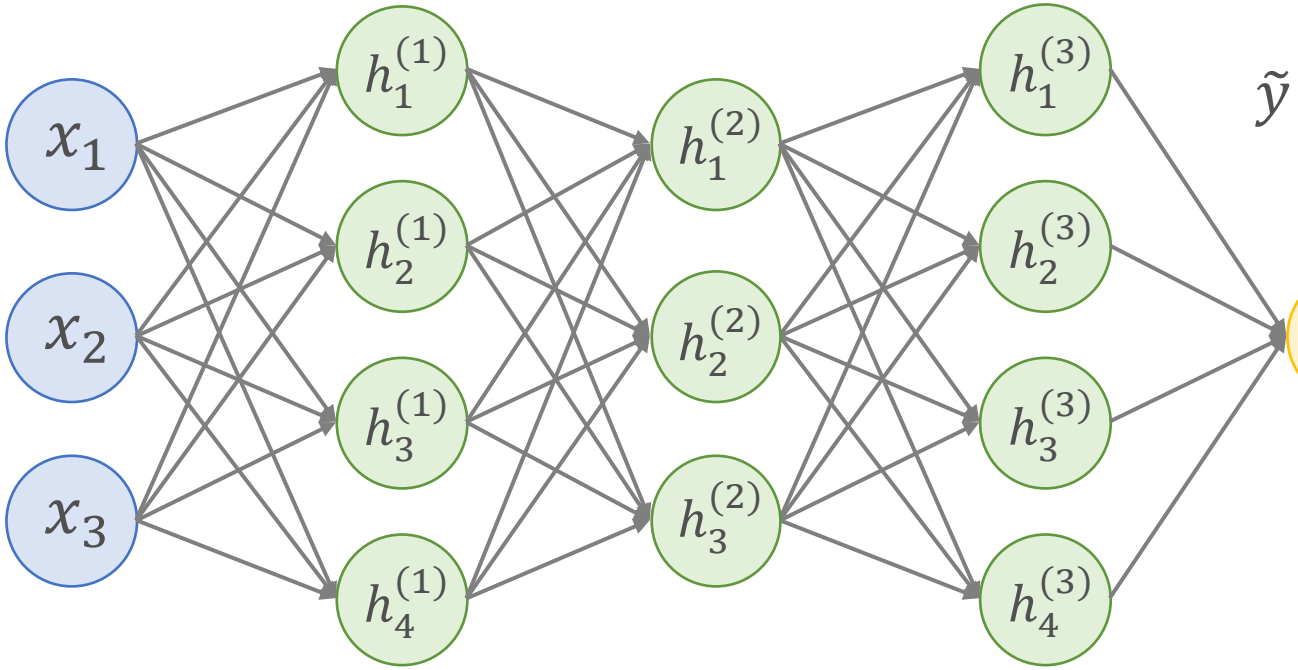
Cross Entropy Loss

$$\mathcal{L}_{total} = -\frac{1}{m}\sum_{i}\mathcal{L}_{CE}(y_i, \tilde{y}_i)$$

Parameters $\theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{W}^{(3)}, \mathbf{W}^{(o)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}, \mathbf{b}^{(3)}, \mathbf{b}^{(o)}\}$,

$$\theta^* = \arg\min_{\theta} \mathcal{L}_{total}$$

Back-Propagation



$$\tilde{y} = \varphi\left(\mathbf{W}^{(o)}\mathbf{h}^{(3)} + \mathbf{b}^{(o)}\right)$$

$$\frac{\partial \mathcal{L}}{\partial \tilde{y}}$$

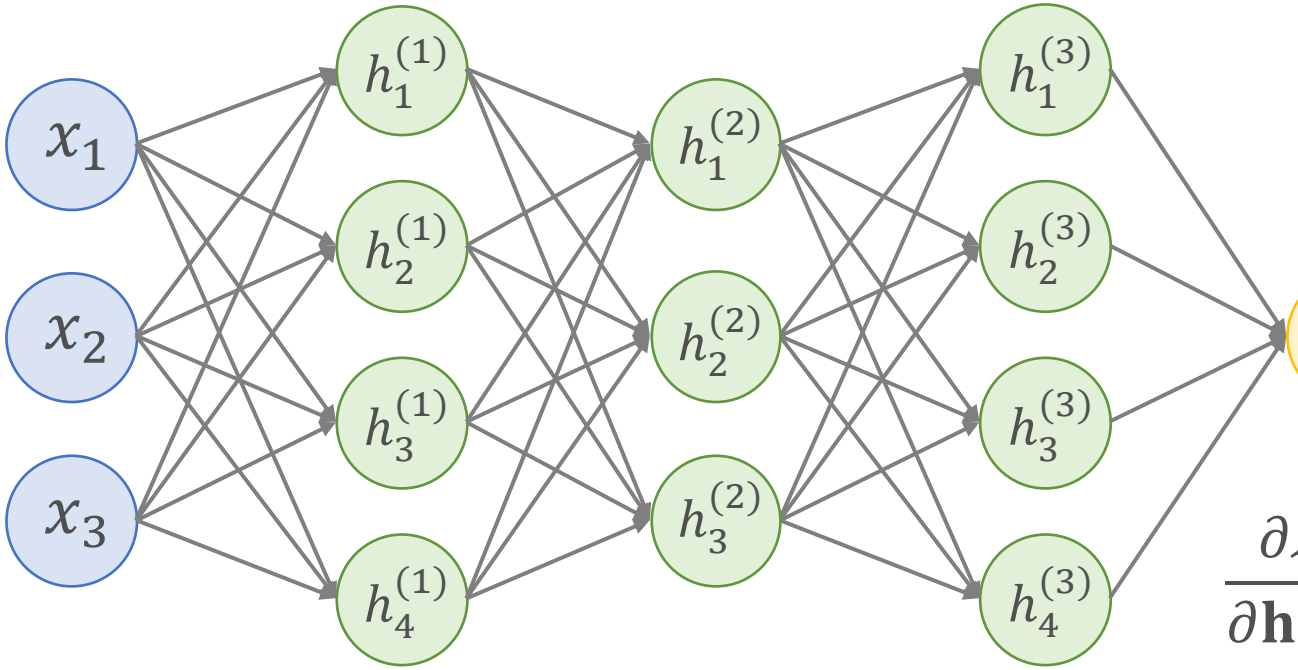
$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(3)}} = \frac{\partial \mathcal{L}}{\partial \tilde{y}} \cdot \frac{\partial \tilde{y}}{\partial \mathbf{h}^{(3)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(o)}} = \frac{\partial \mathcal{L}}{\partial \tilde{y}} \cdot \frac{\partial \tilde{y}}{\partial \mathbf{W}^{(o)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}^{(o)}} = \frac{\partial \mathcal{L}}{\partial \tilde{y}} \cdot \frac{\partial \tilde{y}}{\partial \mathbf{b}^{(o)}}$$

Back-Propagation

$$\mathbf{h}^{(3)} = \varphi\left(\mathbf{W}^{(3)}\mathbf{h}^{(2)} + \mathbf{b}^{(3)}\right)$$



$$\frac{\partial \mathcal{L}}{\partial \tilde{y}}$$

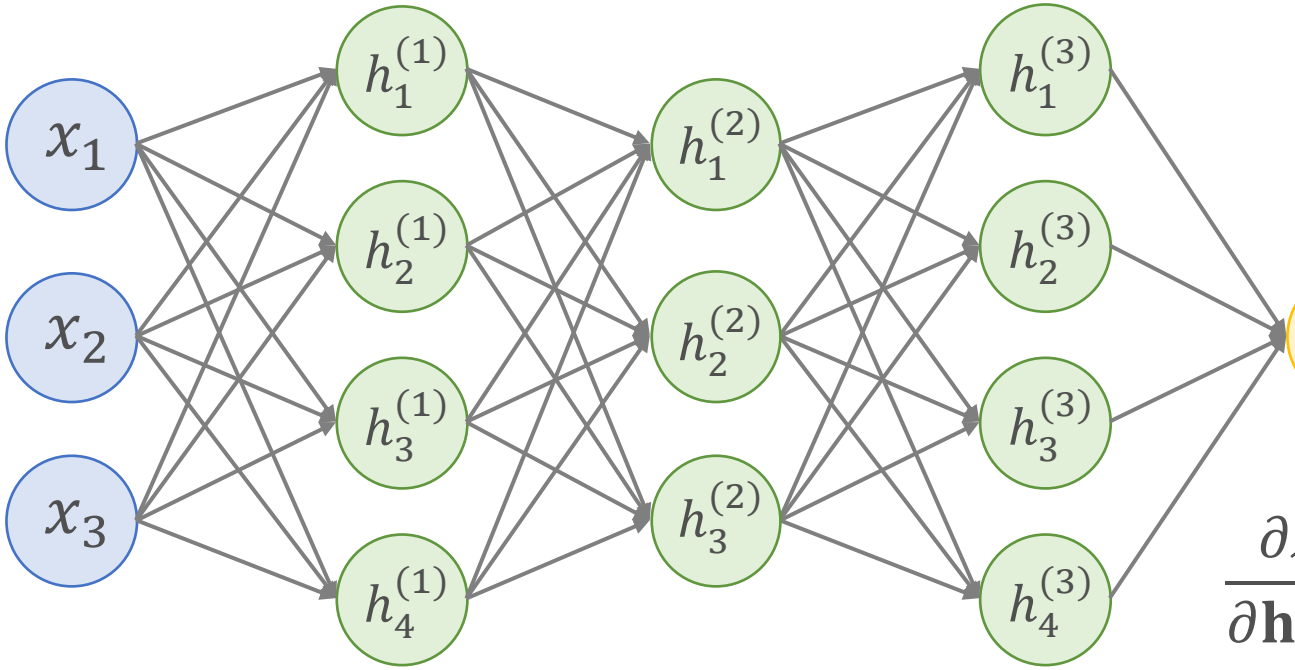
$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(3)}} = \frac{\partial \mathcal{L}}{\partial \tilde{y}} \cdot \frac{\partial \tilde{y}}{\partial \mathbf{h}^{(3)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(3)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(3)}} \cdot \frac{\partial \mathbf{h}^{(3)}}{\partial \mathbf{W}^{(3)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(2)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(3)}} \cdot \frac{\partial \mathbf{h}^{(3)}}{\partial \mathbf{h}^{(2)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}^{(3)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(3)}} \cdot \frac{\partial \mathbf{h}^{(3)}}{\partial \mathbf{b}^{(3)}}$$

Back-Propagation



$$\frac{\partial \mathcal{L}}{\partial \tilde{y}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(3)}} = \frac{\partial \mathcal{L}}{\partial \tilde{y}} \cdot \frac{\partial \tilde{y}}{\partial \mathbf{h}^{(3)}}$$

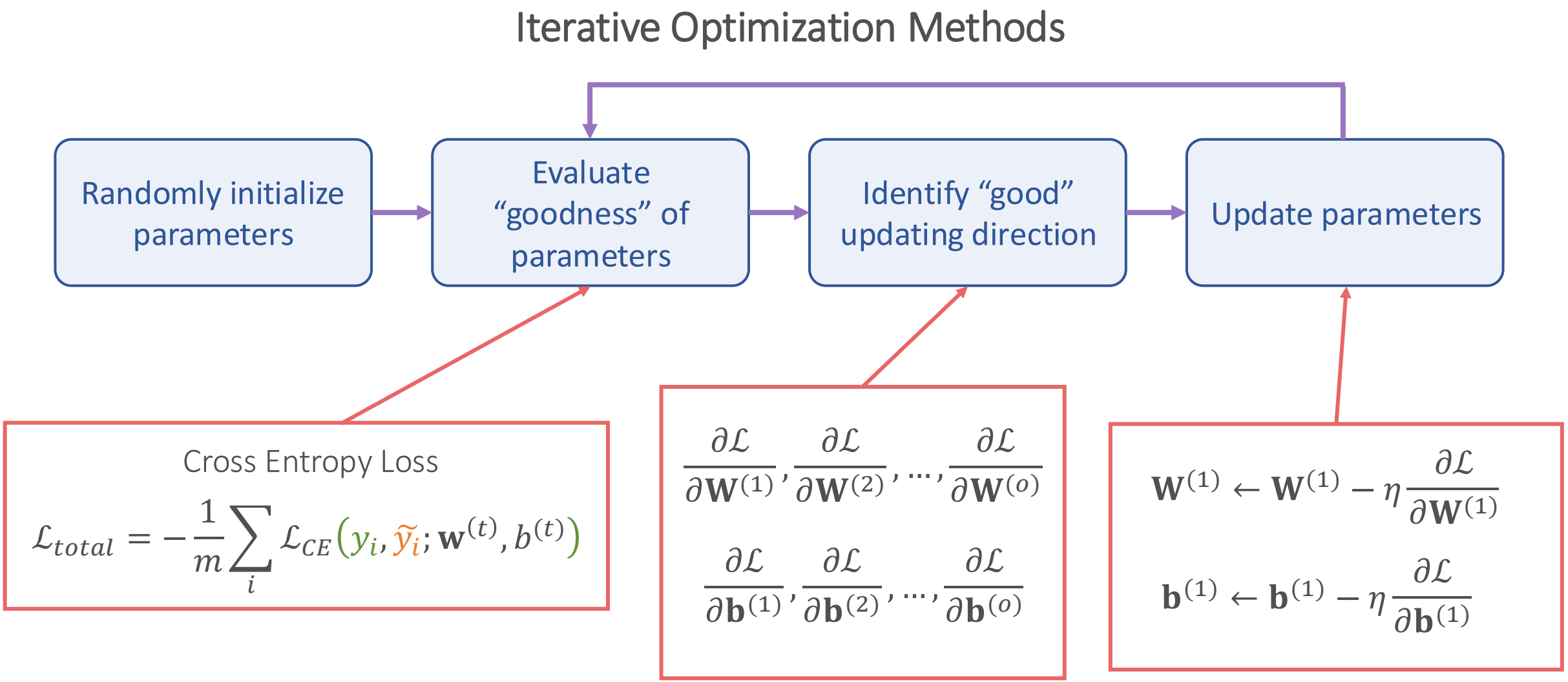
$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(1)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(1)}} \cdot \frac{\partial \mathbf{h}^{(1)}}{\partial \mathbf{W}^{(1)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(2)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(3)}} \cdot \frac{\partial \mathbf{h}^{(3)}}{\partial \mathbf{h}^{(2)}}$$

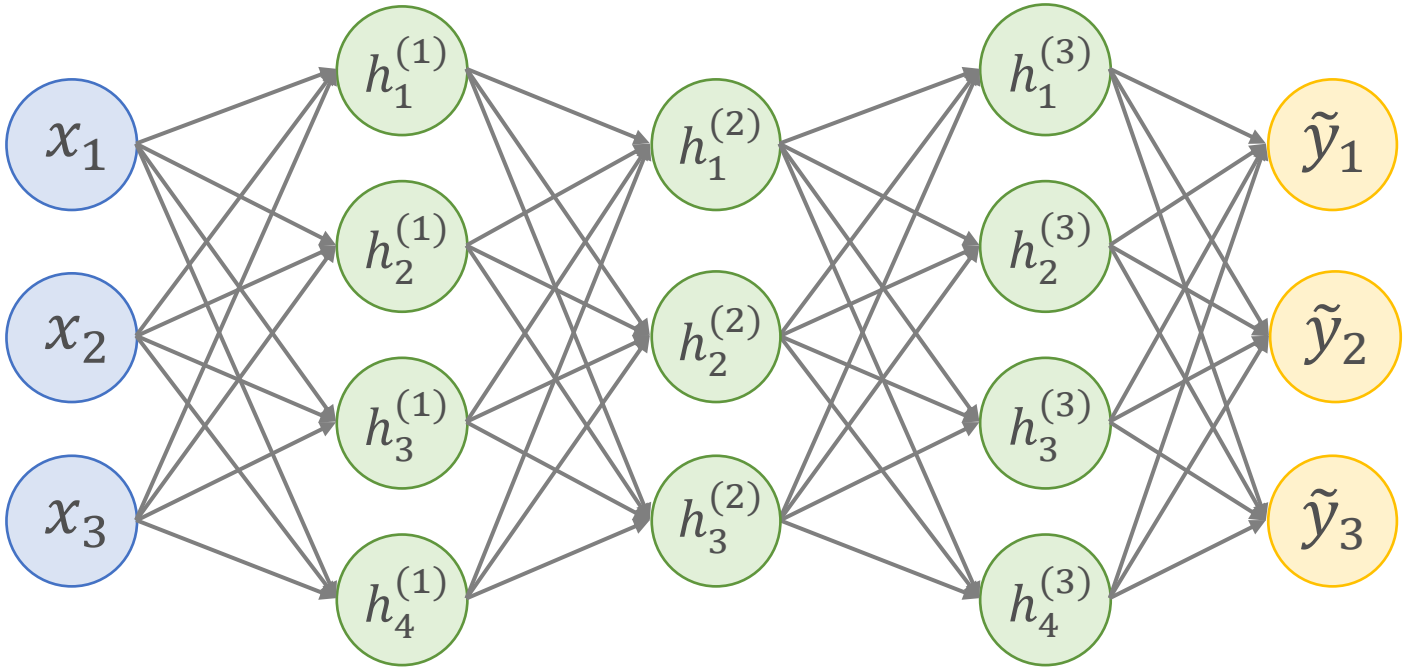
$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}^{(1)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(1)}} \cdot \frac{\partial \mathbf{h}^{(1)}}{\partial \mathbf{b}^{(1)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(1)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(2)}} \cdot \frac{\partial \mathbf{h}^{(2)}}{\partial \mathbf{h}^{(1)}}$$

Training Process



From Binary to Multiclass Classification

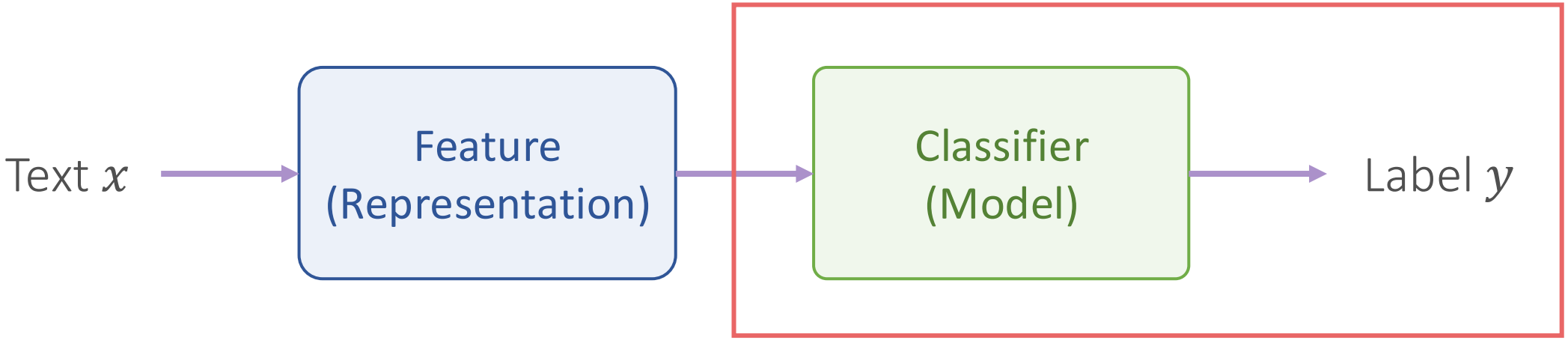


Prediction $= \arg\max_{c} \tilde{y}_c$

Multiclass Cross Entropy Loss

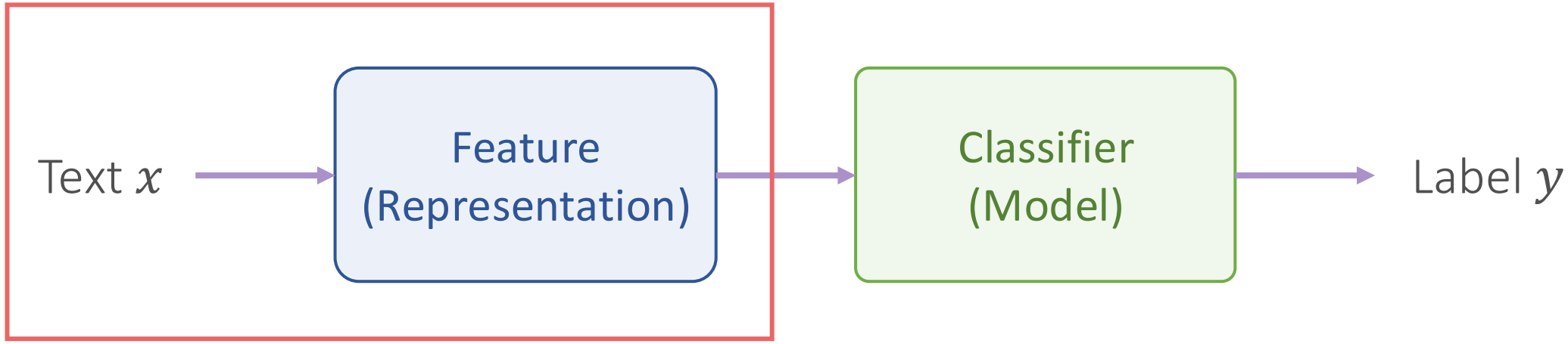
$$\mathcal{L}_{CE}(y, \tilde{y}) = -\sum_{c=0}^{C} y_c \log P(y = c|\, \mathbf{x})$$

Neural Networks



- Neural Networks
  - Find a non-linear decision boundary to map feature vector $\mathbf{x}$ to label y

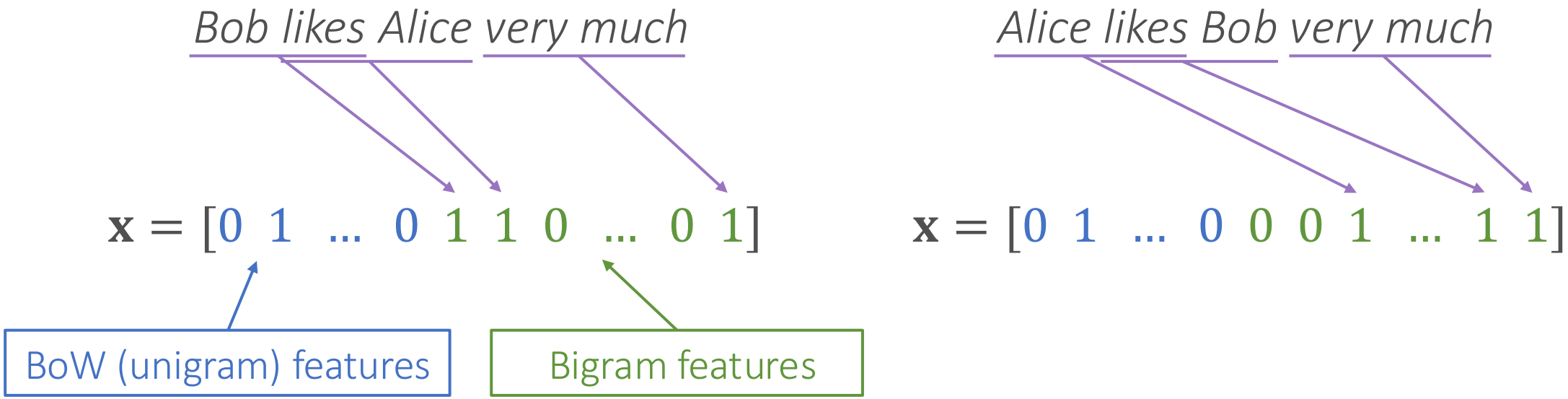
Bag-of-Words and Bag-of-N-Grams



- Bag-of-Words (BoW)
  - A set of words
- Bag-of-N-Grams
  - A set of n-grams

We will discuss “learnable” features later!

Bag-of-Words and N-Gram Features



We can consider trigrams, 4-grams, …

Encode a text to ***one vector***

Word-Level Understanding

great 1 of 2 *adjective*

1 as in *excellent*

of the very best kind

this cake is *great*!

Synonyms & Similar Words

excellent	wonderful	terrific
awesome	fantastic	superb
lovely	beautiful	fabulous
marvelous	stellar	prime
fine	hot	neat
quality	classic	cool
famous	heavenly	good
splendid	exceptional	divine

Antonyms & Near Antonyms

terrible	poor	awful
lousy	pathetic	atrocious
bad	rotten	wretched
vile	unsatisfactory	inferior
substandard	execrable	low-grade
mediocre	second-class	middling

https://www.merriam-webster.com/thesaurus/great

Words as Vectors

Bob *likes* *Alice* *very* *much*

$$W = \begin{bmatrix} | & | & | & | & | \\ w_{bob} & w_{likes} & w_{Alice} & w_{very} & w_{much} \\ | & | & | & | & | \end{bmatrix}$$

Use ***one vector*** to represent ***each word***

Text = A list of vectors

Advantages?

Words as Vectors

Map each word to a vector!

$$\text{similarity}(word1, word2) = \frac{\mathbf{v}_{word1} \cdot \mathbf{v}_{word2}}{\|\mathbf{v}_{word1}\| \times \|\mathbf{v}_{word2}\|}$$

$$\mathbf{v}_{great} = [0.12, 0.38, -0.91, 0.57, -0.64]$$

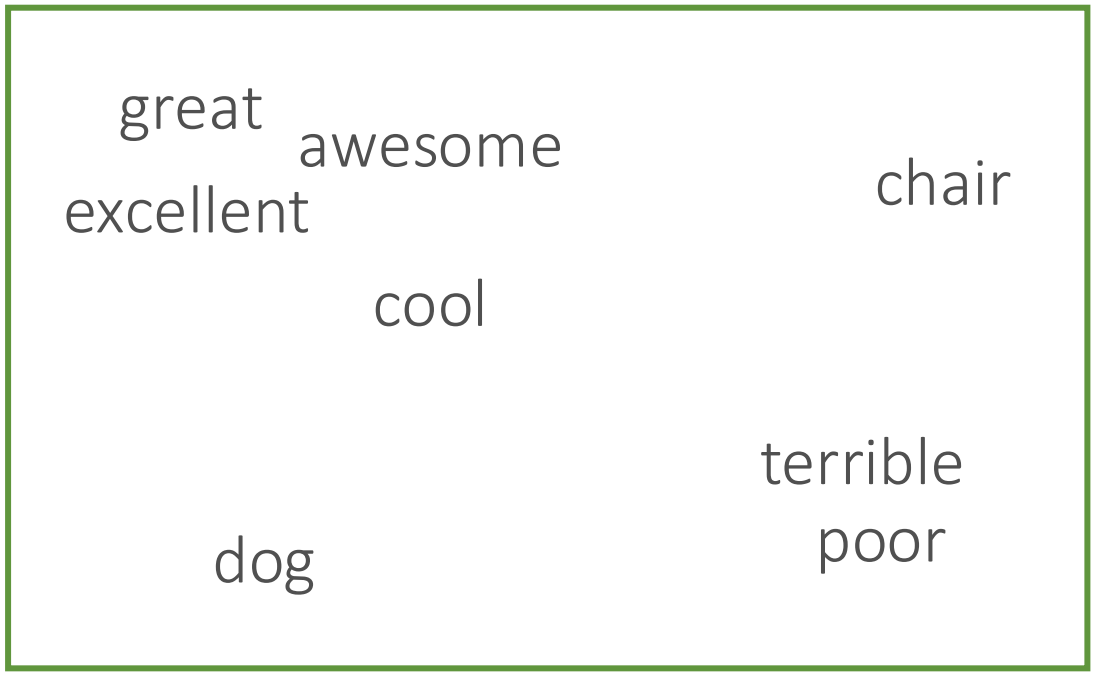
$$\mathbf{v}_{excellent} = [0.16, 0.47, -0.87, 0.50, -0.55]$$

$$\mathbf{v}_{awesome} = [0.08, 0.28, -0.90, 0.61, -0.54]$$

$$\mathbf{v}_{terrible} = [0.92, -0.36, 0.11, -0.24, 0.14]$$

$$\mathbf{v}_{poor} = [0.85, -0.40, 0.02, -0.31, 0.23]$$

Semantic meaning of words



Semantic relationship between words

How to learn those word vectors/embeddings/representations?

Distributional Hypothesis

Distributional hypothesis: A word's meaning is given by the words that frequently appear close-by

J.R.Firth 1957

- "You shall know a word by the company it keeps"
- One of the most successful ideas of modern statistical NLP!

…government debt problems turning into	*banking*	*crises as happened in 2009…*
…saying that Europe needs unified	*banking*	*regulation to replace the hodgepodge…*
…India has just given its	*banking*	*system a shot in the arm…*

These context words will represent banking

Slide adapted from Chris Manning

Distributional Hypothesis: Example

C1: A bottle of ___ is on the table.

C2: Everybody likes ___.

C3: Don't have ___ before you drive.

C4: I bought ___ yesterday.

	C1	C2	C3	C4
wine	1	1	1	1
juice	1	1	0	1
loud	0	0	0	0
apples	0	1	0	1
choices	0	1	0	0
motor-oil	1	0	0	1

A word's meaning is given by the words that frequently appear close-by

Slide adapted from Danqi Chen

Word2Vec: Learning Problem

…government debt problems turning into	*banking*	*crises as happened in 2009…*
…saying that Europe needs unified	*banking*	*regulation to replace the hodgepodge…*
…India has just given its	*banking*	*system a shot in the arm…*

Based on distributional hypothesis

Center word ⟷ Context words

Given the context words, we can predict the most likely center word

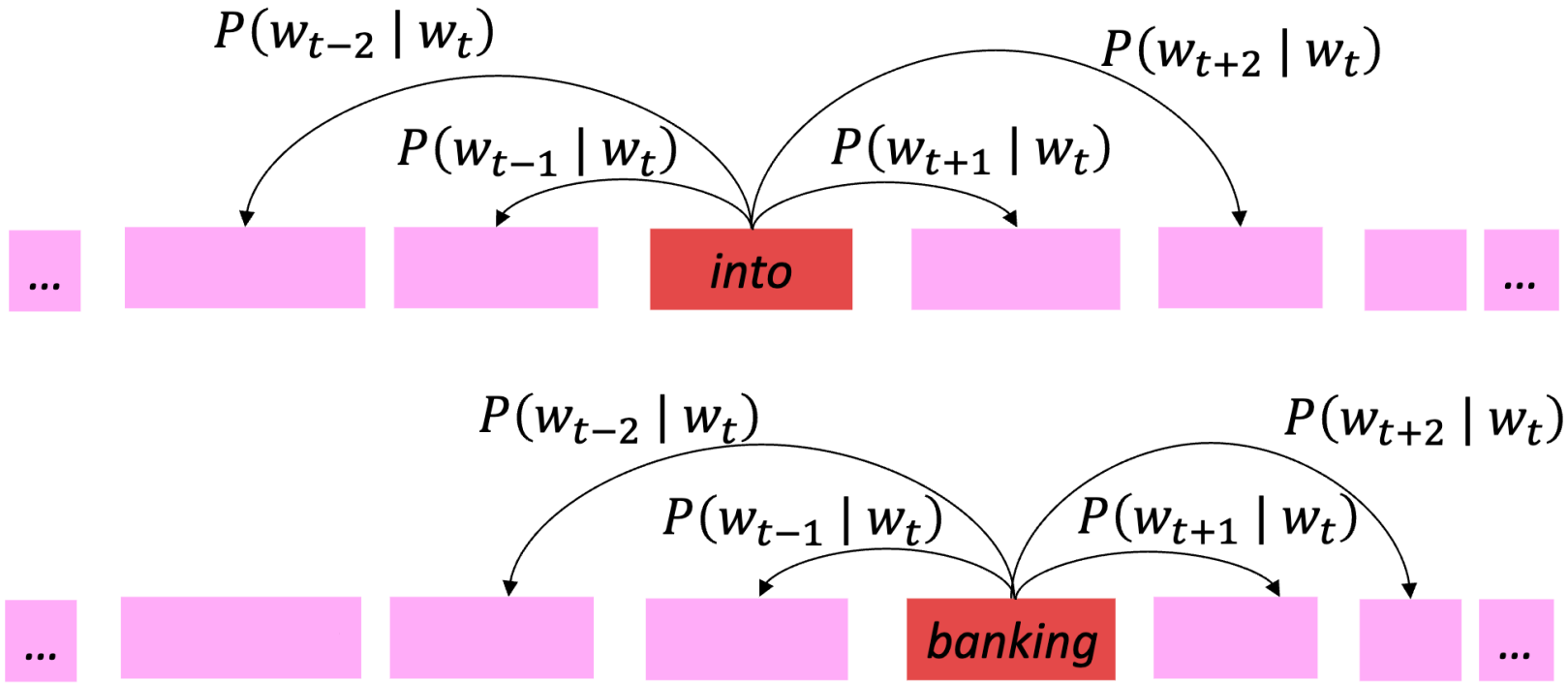
Given the center word, we can predict the most likely context words

Efficient Estimation of Word Representations in Vector Space, 2013

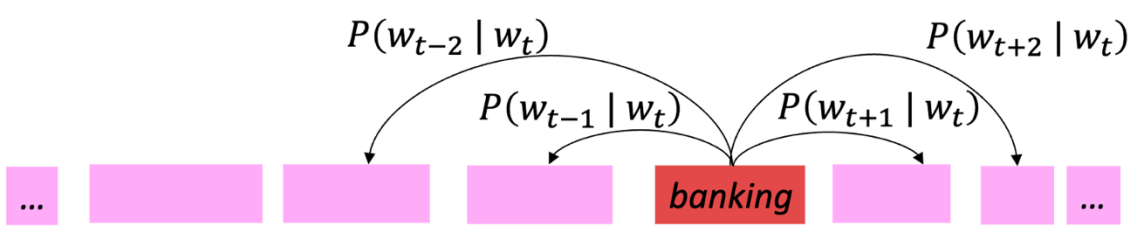
Word2Vec: Overview

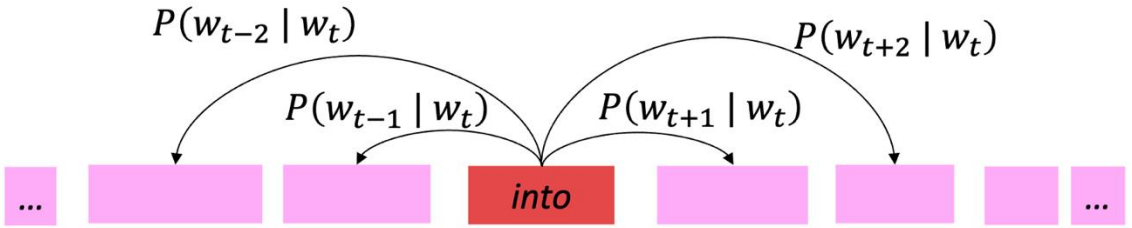
- **Main idea:** we want to use words to predict their context words
- Context: a fixed window of size m

Use center word w_t to predict context words w_{t-m} to w_{t+m}



Word2Vec: Overview





$P(\text{money} \mid \text{banking}) = 0.21$

$P(\text{crises} \mid \text{banking}) = 0.18$

$P(\text{deposit} \mid \text{banking}) = 0.15$

$P(\text{dog} \mid \text{banking}) = 0.01$

$P(\text{turning} \mid \text{banking}) = 0.06$

...

$P(\text{sunny} \mid \text{banking}) = 0.02$

$P(\text{money} \mid \text{into}) = 0.06$

$P(\text{crises} \mid \text{into}) = 0.19$

$P(\text{deposit} \mid \text{into}) = 0.03$

$P(\text{dog} \mid \text{into}) = 0.01$

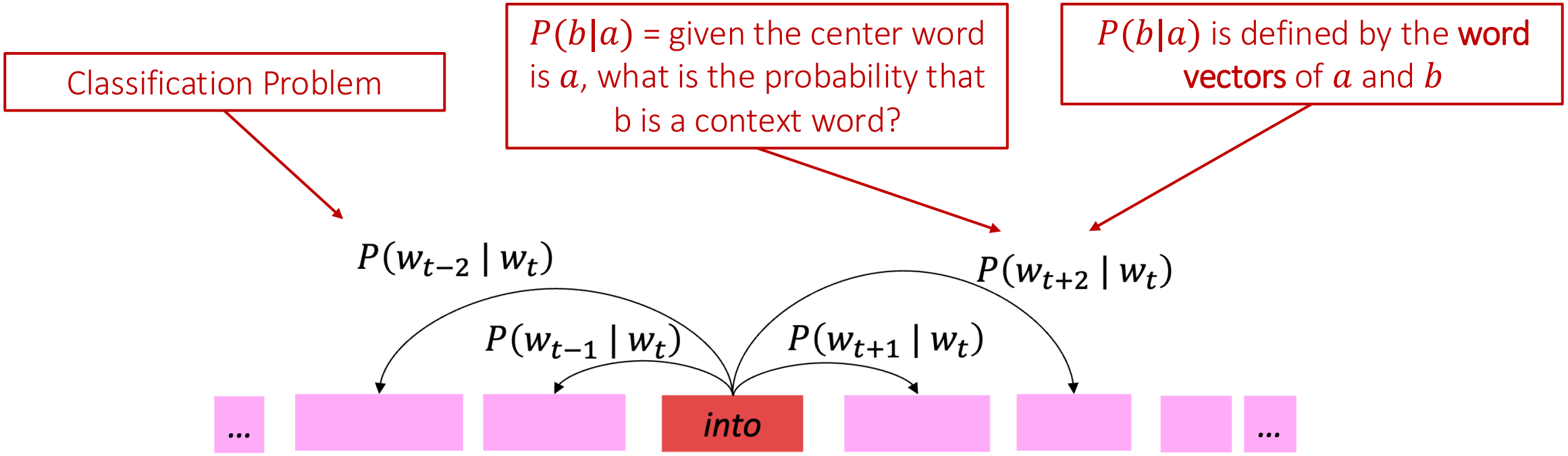
$P(\text{turning} \mid \text{into}) = 0.30$

...

$P(\text{sunny} \mid \text{into}) = 0.01$

Word2Vec: Overview

- **Main idea:** use center word to predict its context words



Word2Vec: Defining Probabilities (Simplified Version)

How to calculate $P(w_{context} | w_{center})$?

$$\frac{\mathbf{v}_{word1} \cdot \mathbf{v}_{word2}}{\|\mathbf{v}_{word1}\| \times \|\mathbf{v}_{word2}\|}$$

We consider Inner product $\mathbf{v}_{w_{center}} \cdot \mathbf{v}_{w_{context}}$ as the score

If $\mathbf{v}_{w_{center}} \cdot \mathbf{v}_{w_{context}}$ is higher, $P(w_{context} | w_{center})$ is higher

$$P(w_{context} | w_{center}) = \frac{\exp(\mathbf{v}_{w_{center}} \cdot \mathbf{v}_{w_{context}})}{\sum_{k \in V} \exp(\mathbf{v}_{w_{center}} \cdot \mathbf{v}_k)}$$

Scores can be asymmetric! It is less likely that a word appears in its own context

Normalize scores to probabilities

Word2Vec: Defining Probabilities (Full Version)

How to calculate $P(w_{context} | w_{center})$?

$$\frac{\mathbf{v}_{word1} \cdot \mathbf{v}_{word2}}{\|\mathbf{v}_{word1}\| \times \|\mathbf{v}_{word2}\|}$$

We consider Inner product $\mathbf{u}_{w_{center}} \cdot \mathbf{v}_{w_{context}}$ as the score

If $\mathbf{u}_{w_{center}} \cdot \mathbf{v}_{w_{context}}$ is higher, $P(w_{context} | w_{center})$ is higher

$$P(w_{context} | w_{center}) = \frac{\exp(\mathbf{u}_{w_{center}} \cdot \mathbf{v}_{w_{context}})}{\sum_{k \in V} \exp(\mathbf{u}_{w_{center}} \cdot \mathbf{v}_k)}$$

Normalize scores to probabilities

Word2Vec: Defining Probabilities (Full Version)

We have two sets of vectors for each word in the vocabulary

$\mathbf{u}_w \in \mathbb{R}^d$: word vector when w is a center word

$\mathbf{v}_w \in \mathbb{R}^d$: word vector when w is a context word

$$P(w_{t+j} | w_t ; \theta) = \frac{\exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_{w_{t+j}})}{\sum_{k \in V} \exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_k)}$$

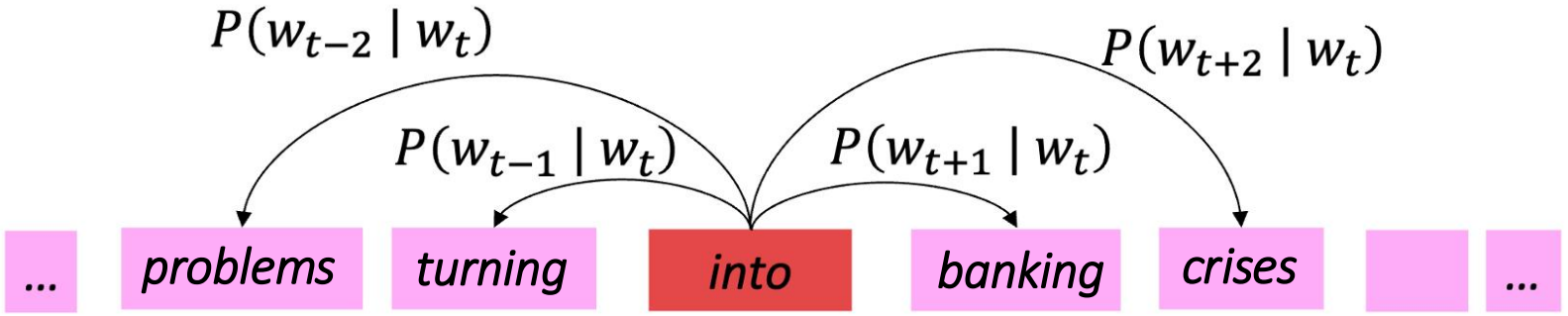
Normalize over entire vocabulary to give probability distribution

The score to indicate how likely the context word w_{t+j} appears with the center word w_t

Softmax function: mapping arbitrary values to a probability distribution

$$\text{softmax}(t) = \frac{e^t}{\sum_c e^c}$$

Word2Vec: Training Intuition



$P(\text{problems} \mid \text{into})$ ↑

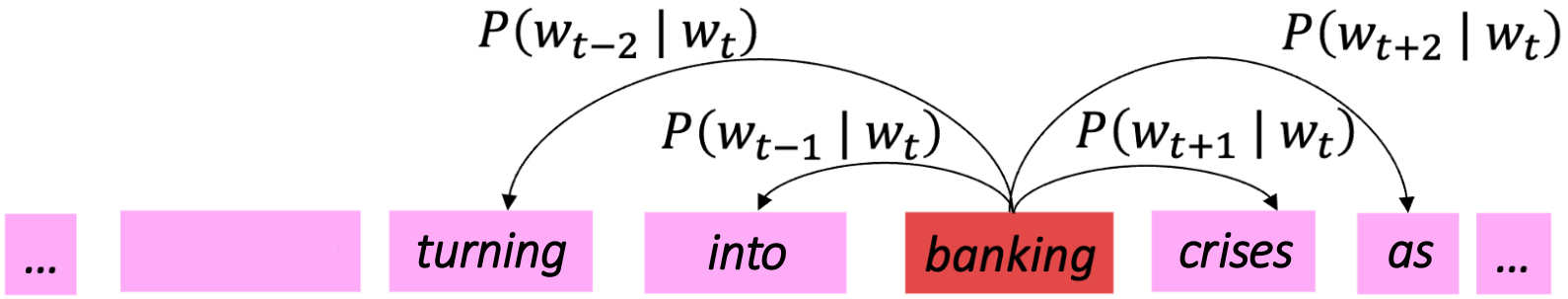
$P(\text{turning} \mid \text{into})$ ↑

$P(\text{banking} \mid \text{into})$ ↑

$P(\text{crises} \mid \text{into})$ ↑

$P(\text{other words} \mid \text{into})$ ↓

Word2Vec: Training Intuition



$P(\text{turning} \mid \text{banking})$ ↑

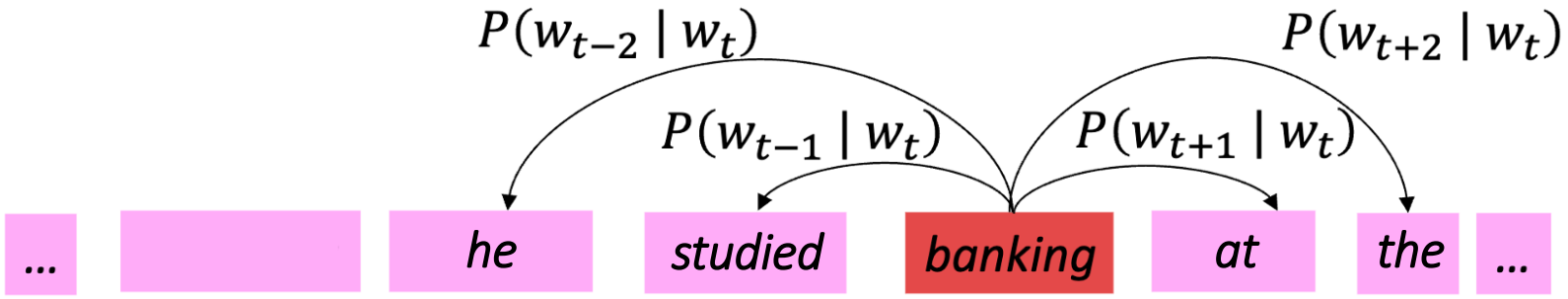
$P(\text{into} \mid \text{banking})$ ↑

$P(\text{crises} \mid \text{banking})$ ↑

$P(\text{as} \mid \text{banking})$ ↑

$P(\text{other words} \mid \text{banking})$ ↓

Word2Vec: Training Intuition



$P(\text{he} \mid \text{banking})$ ↑

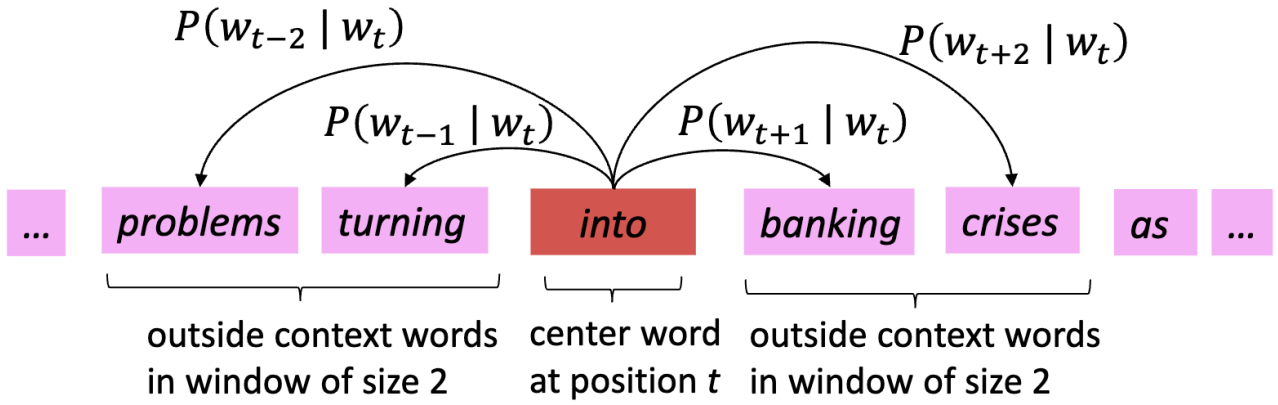
$P(\text{studied} \mid \text{banking})$ ↑

$P(\text{at} \mid \text{banking})$ ↑

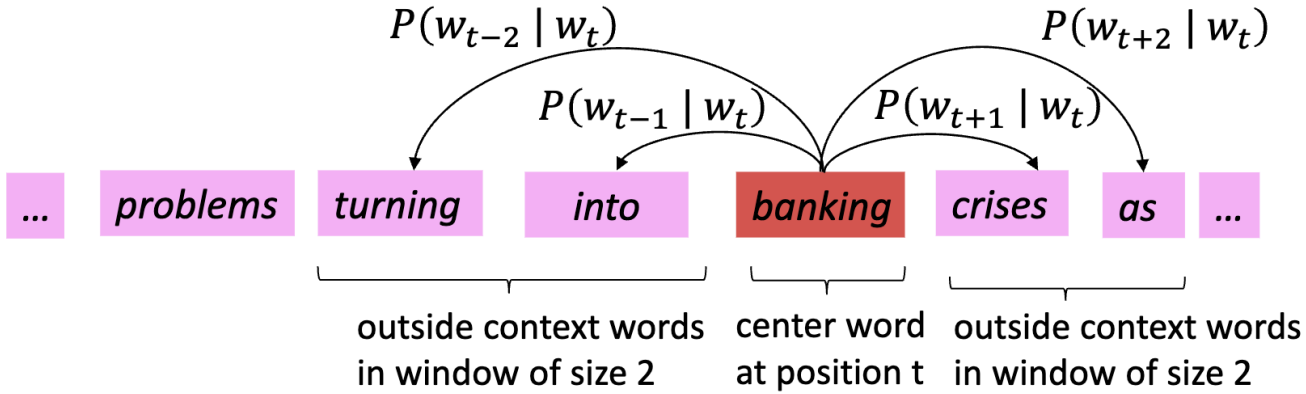
$P(\text{the} \mid \text{banking})$ ↑

$P(\text{other words} \mid \text{banking})$ ↓

Word2Vec: Training Data



Collect into training data
(into, problems)
(into, turning)
(into, banking)
(into, crises)



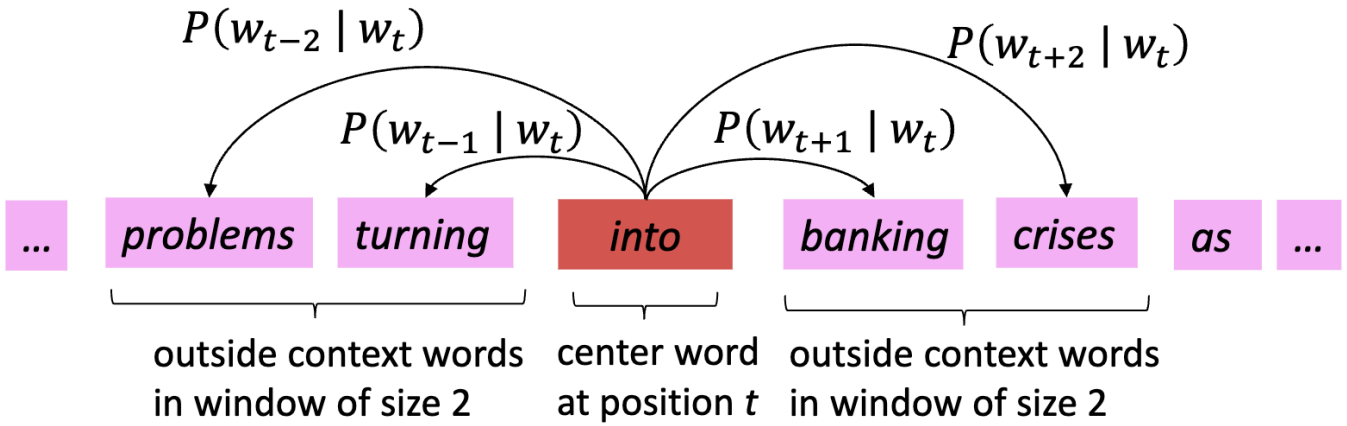
Collect into training data
(banking, turning)
(banking, into)
(banking, crises)
(banking, as)

Maximize the likelihood

$P(\text{problems}|\text{into}) \times P(\text{turning}|\text{into}) \times P(\text{banking}|\text{into}) \times P(\text{crises}|\text{into})$

$\times P(\text{turning}|\text{banking}) \times P(\text{into}|\text{banking}) \times P(\text{crises}|\text{banking}) \times P(\text{as}|\text{banking})$

Word2Vec: Likelihood



For each position $t = 1, \ldots, T$, predict context words within a window of fixed size m, given center word w_t

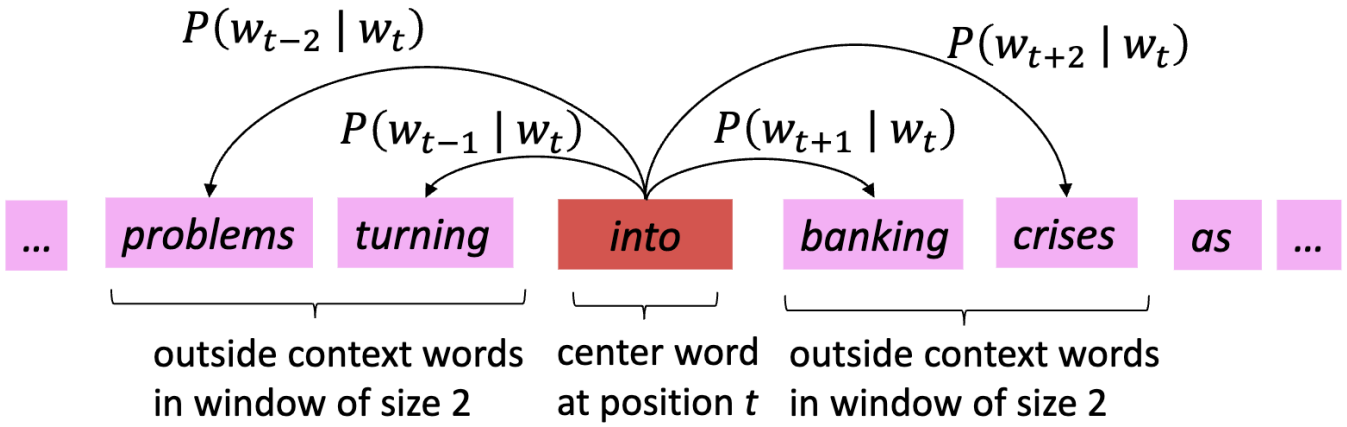
θ all parameters to be optimized

$$\text{Likelihood} = \mathcal{L}(\theta) = \prod_{t=1}^{T} \prod_{-m \le j \le m, j \neq 0} P(w_{t+j} \mid w_t ; \theta)$$

Probability given the center word w_t

For each position $t = 1, \ldots, T$ Likelihood for all context words given center word w_t

Word2Vec: Objective Function



The objective function $J(\theta)$ is the (average) negative log likelihood

$$J(\theta) = -\frac{1}{T}\log \mathcal{L}(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-m\leq j\leq m, j\neq 0} \log P\left(w_{t+j} \middle| w_t ; \theta\right)$$

We minimize the objective function (also called cost or loss function)

Word2Vec: How to Train Word Vectors?

Parameters: $\theta = \{\{\mathbf{u}_k\}, \{\boldsymbol{v}_k\}\}$

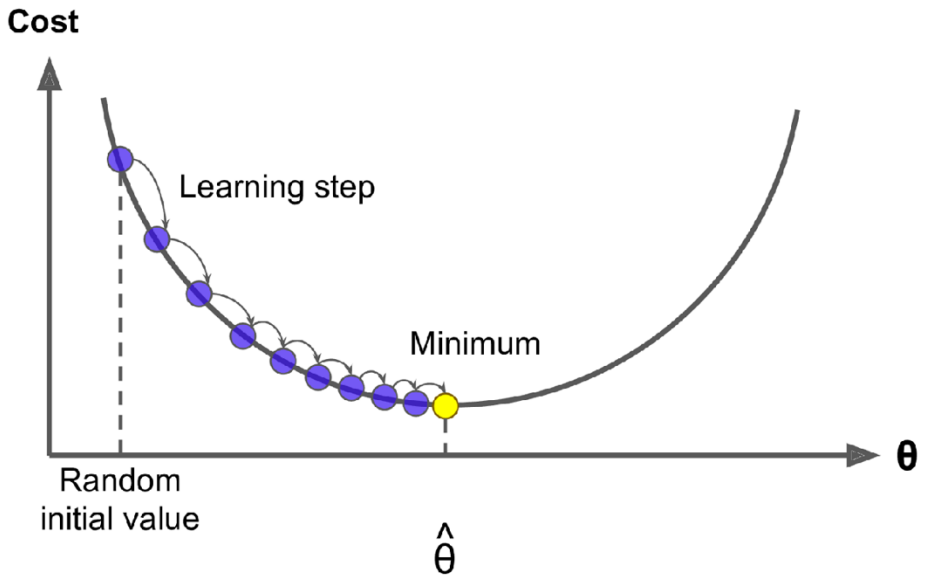
Objective function: $J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-m\leq j\leq m, j\neq 0} \log P\left(w_{t+j} \middle| w_t ; \theta\right)$

Our goal: find parameters θ that minimize the objective function $J(\theta)$

Solution: stochastic gradient descent (SGD)

- Randomly initialize parameters θ
- For each iteration $\theta \leftarrow \theta - \eta \nabla_\theta J(\theta)$

Learning step Gradient



https://insightfultscript.com/collections/programming/machine-learning/sgd/

Word2Vec: Computing the Gradients

Objective function

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

$$= \frac{1}{T}\sum_{t=1}^{T}\sum_{-m\le j\le m, j\neq 0} \boxed{-\log P\left(w_{t+j}\middle|\, w_t\,;\theta\right)}$$

The gradients can be calculated separately!

For simplicity, we consider one pair of center/context words (o, c)

$$y = -\log P(c|o\,;\theta) = -\log\left(\frac{\exp(\mathbf{u}_o \cdot \mathbf{v}_c)}{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)}\right)$$

$$\boxed{\frac{\partial y}{\partial \mathbf{u}_o} \quad \frac{\partial y}{\partial \boldsymbol{v}_c}}$$

We need to compute this!

Word2Vec: Computing the Gradients

$$y = -\log P(c|o) = -\log\left(\frac{\exp(\mathbf{u}_o \cdot \mathbf{v}_c)}{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)}\right) = \boxed{-\log(\exp(\mathbf{u}_o \cdot \mathbf{v}_c))} + \log\left(\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)\right)$$

$$= -\mathbf{u}_o \cdot \mathbf{v}_c$$

$\frac{\partial \log(x)}{\partial x} = \frac{1}{x}$ $\quad \frac{\partial \exp(x)}{\partial x} = \exp(x)$

$$\frac{\partial y}{\partial \mathbf{u}_o} = \frac{\partial(-\mathbf{u}_o \cdot \mathbf{v}_c + \log(\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)))}{\partial \mathbf{u}_o} = -\mathbf{v}_c + \frac{\sum_{k\in V}\frac{\partial \exp(\mathbf{u}_o \cdot \mathbf{v}_k)}{\partial \mathbf{u}_o}}{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)}$$

$$= -\mathbf{v}_c + \frac{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)\,\mathbf{v}_k}{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)} = -\mathbf{v}_c + \sum_{k\in V}\frac{\exp(\mathbf{u}_o \cdot \mathbf{v}_k)\,\mathbf{v}_k}{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)}$$

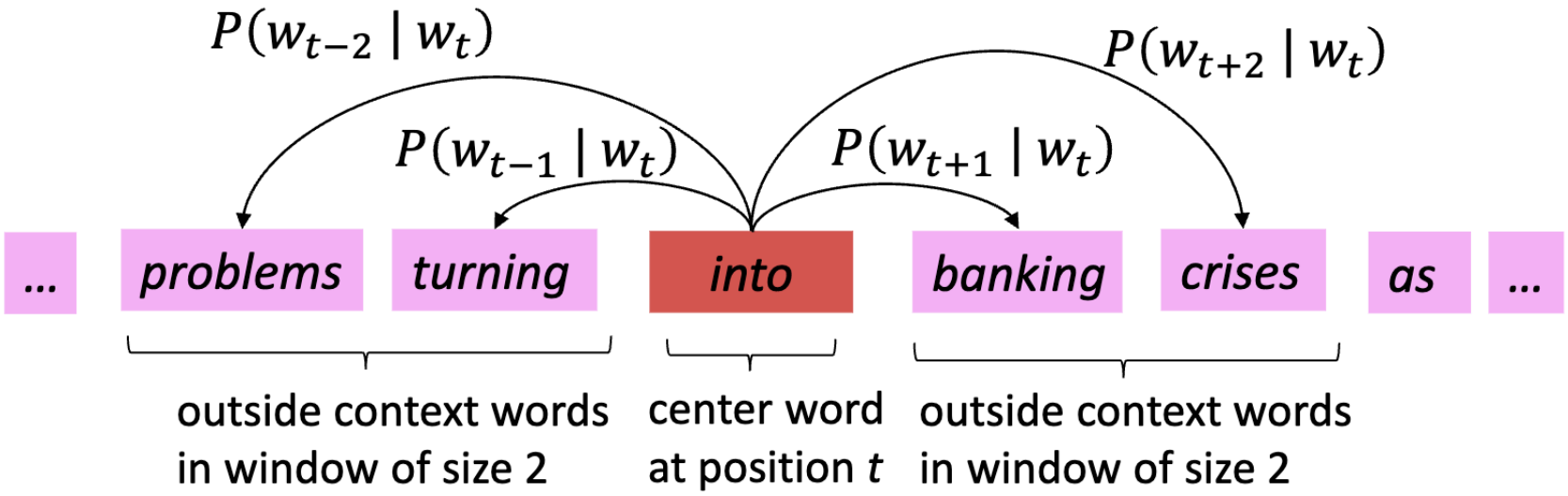
$$= -\mathbf{v}_c + \sum_{k\in V} P(k|o)\,\mathbf{v}_k$$

$$\boxed{\frac{\partial y}{\partial \mathbf{v}_k} = -1(k = c)\mathbf{u}_o + P(k|o)\mathbf{u}_o}$$

Similar calculation step

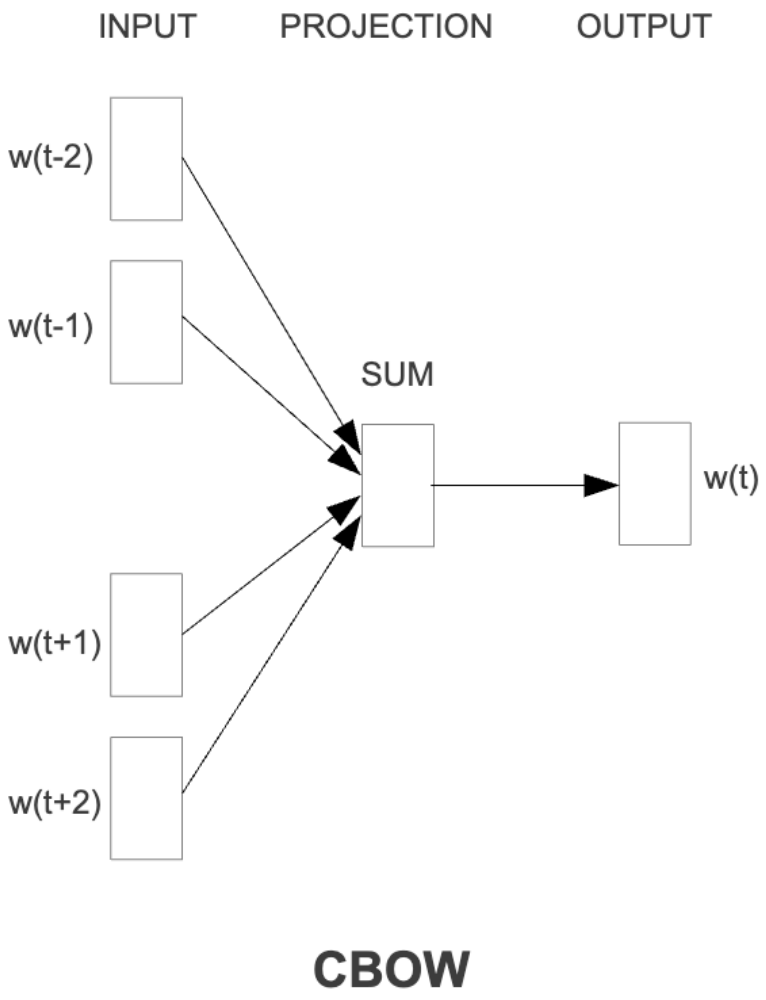
Word2Vec: Training Process

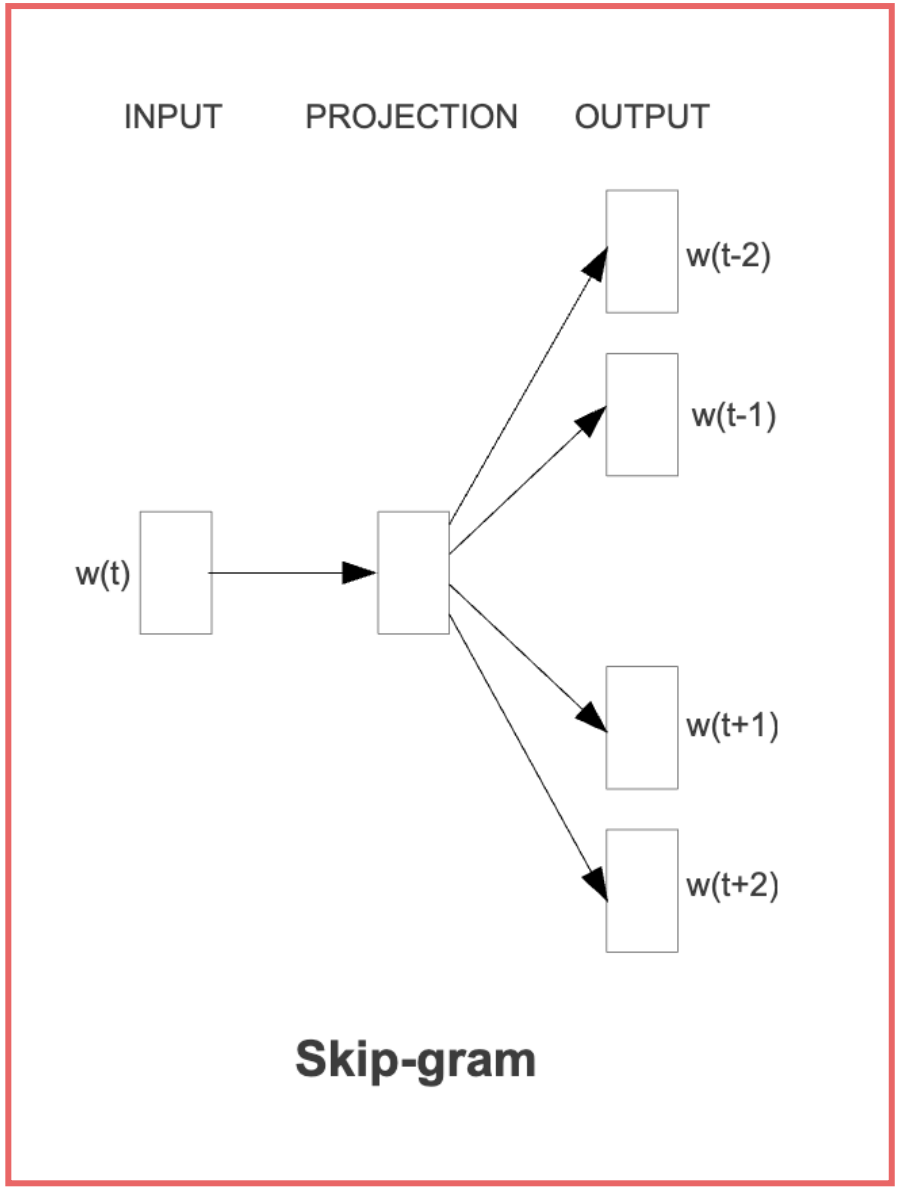
- Randomly initialize parameters $\mathbf{u}_i, \mathbf{v}_i$
- Walk through the training corpus and collect training data (o, c)



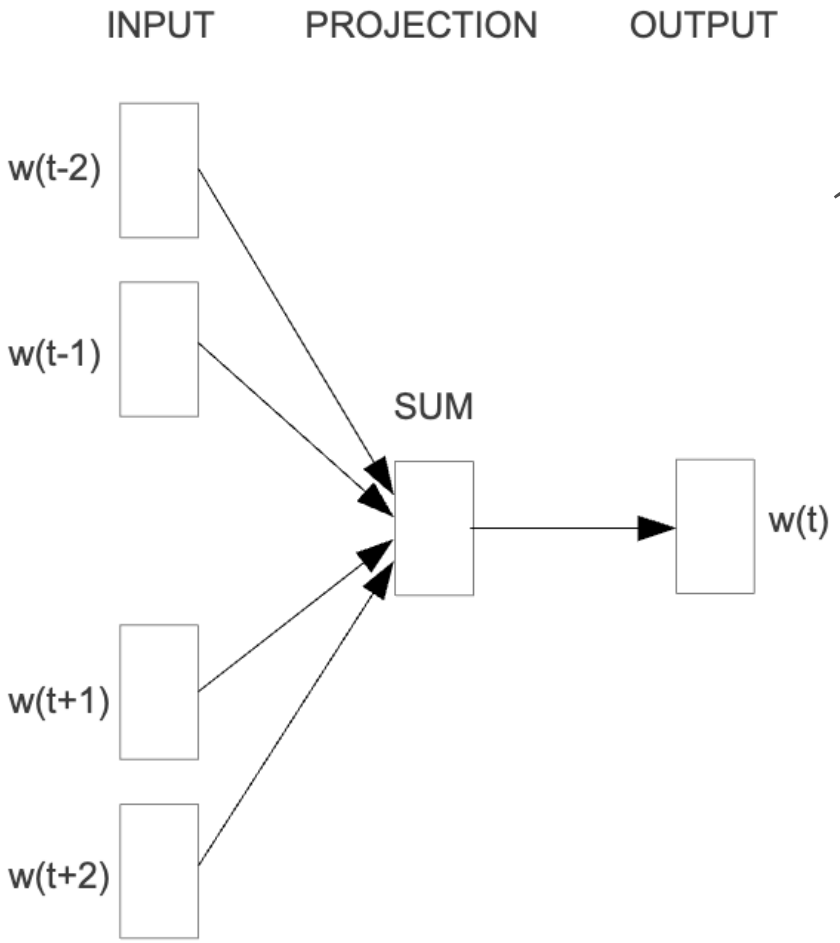
$$\mathbf{u}_o \leftarrow \mathbf{u}_o - \eta \frac{\partial y}{\partial \mathbf{u}_o} \qquad \mathbf{v}_k \leftarrow \mathbf{v}_k - \eta \frac{\partial y}{\partial \mathbf{v}_k} \qquad \forall k \in V$$

Continuous Bag of Words (CBOW) vs Skip-Grams





Continuous Bag of Words (CBOW)



$$\mathcal{L}(\theta) = \prod_{t=1}^{T} P(w_t | \{w_{t+j}\}), -m \leq j \leq m, j \neq 0$$

$$P(w_t | \{w_{t+j}\}) = \frac{\exp(\mathbf{u}_{w_t} \cdot \bar{\mathbf{v}}_t)}{\sum_{k \in V} \exp(\mathbf{u}_k \cdot \bar{\mathbf{v}}_t)}$$

$$\bar{\mathbf{v}}_t = \frac{1}{2m} \sum_{-m \leq j \leq m, j \neq 0} \mathbf{v}_{t+j}$$

GloVe: Global Vectors

GloVe: Global Vectors for Word Representation (Pennington et al. 2014)

Idea: capture ratios of co-occurrence probabilities as linear meaning components in a word vector space

Log-bilinear model $w_i \cdot w_j = \log P(i|j)$

Vector difference $w_i \cdot (w_a - w_b) = \frac{\log P(x|a)}{\log P(x|b)}$

$$J = \sum_{i,j=1}^{V} f(X_{ij})\left(w_i^\top \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$

Global co-occurrence statistics

Training faster and scalable to very large corpora!

FastText: Sub-Word Embeddings

Enriching Word Vectors with Subword Information (Bojanowski et al. 2017)

Similar as Skip-gram, but break words into n-grams with n = 3 to 6

where

3-grams: <wh, whe, her, ere, re>

4-grams: <whe, wher, here, ere>

5-grams: <wher, where, here>

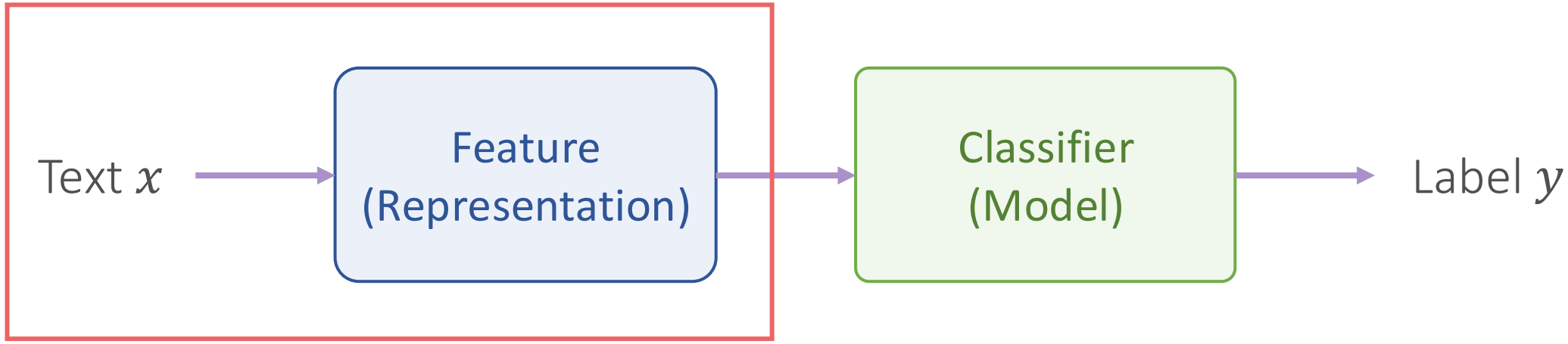
6-grams: <where, where>

Replace $\mathbf{u}_i \cdot \mathbf{v}_j$ with $\sum_{g \in n-grams(w_i)} \mathbf{u}_g \cdot \mathbf{v}_j$

Trained Word Vectors Are Available

- Word2Vec: https://code.google.com/archive/p/word2vec/
- GloVe: https://nlp.stanford.edu/projects/glove/
- FastText: https://fasttext.cc/

Learning-Based Word Vectors



- Learn word vectors directly from text
  - Word2Vec (Skip-Gram and CBOW)
  - GloVe
  - FastText

Word Analogy

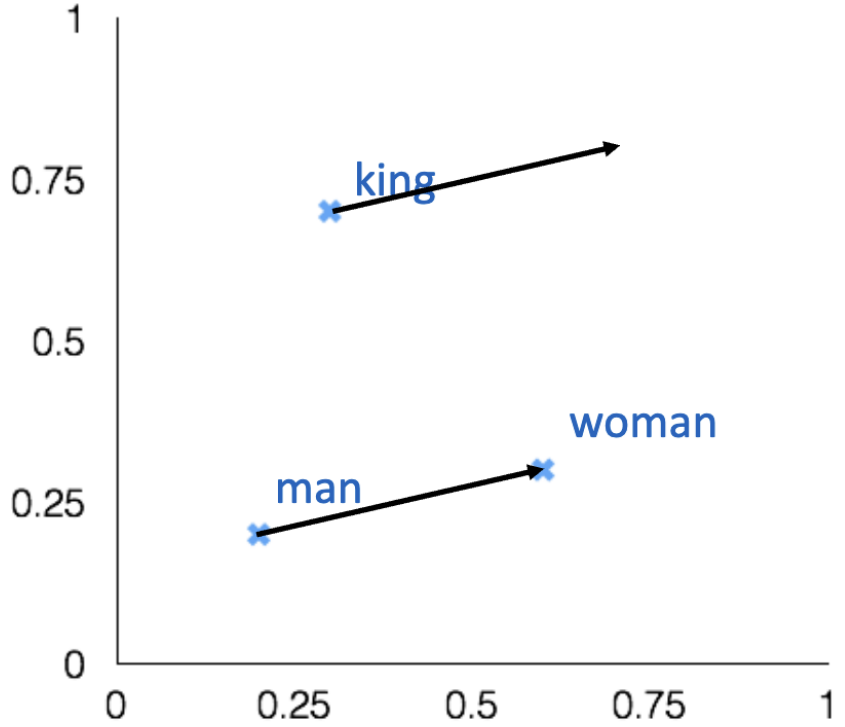
Word analogy

man: woman ≈ king: ?

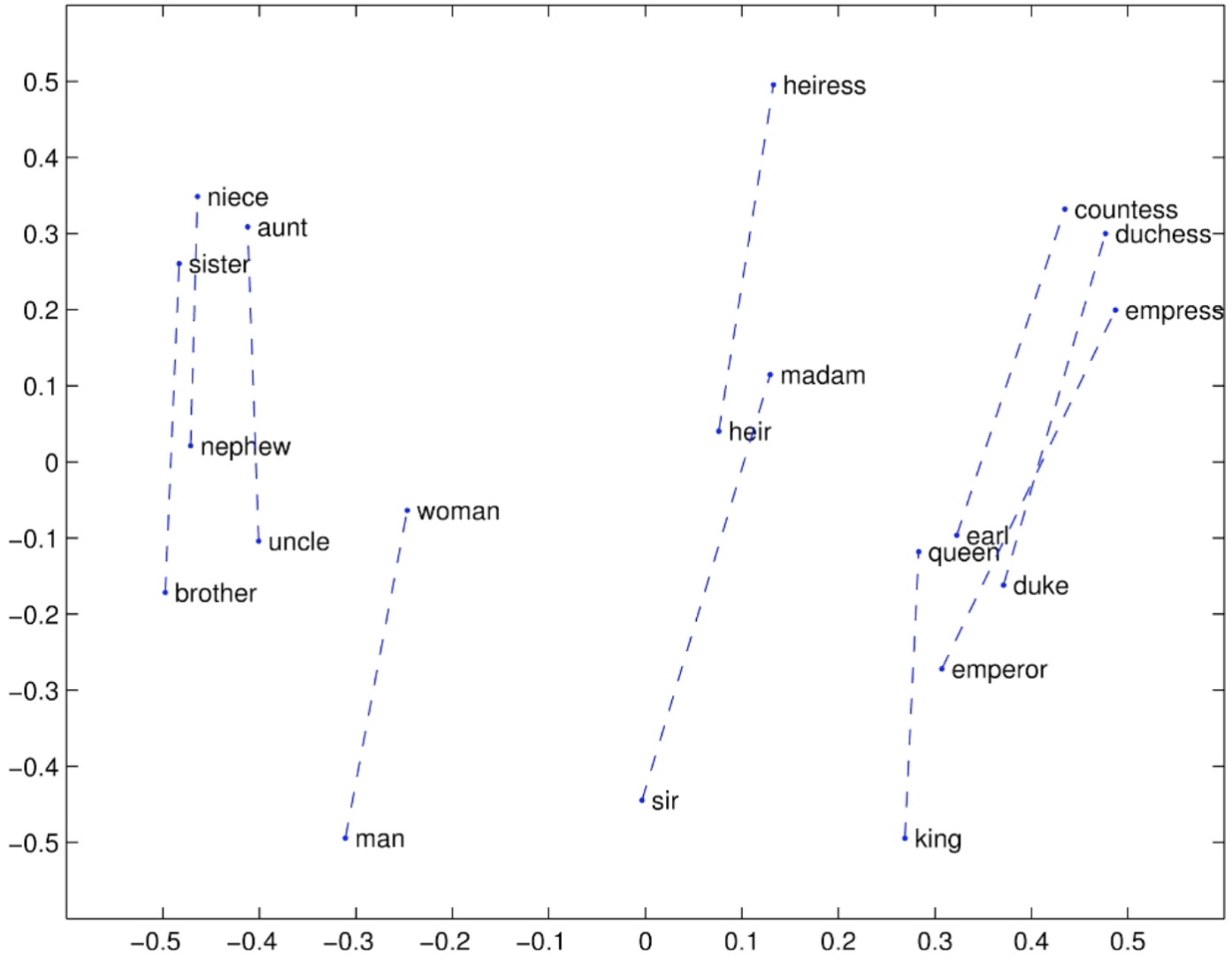
Paris: France ≈ London: ?

bad: worst ≈ cool: ?

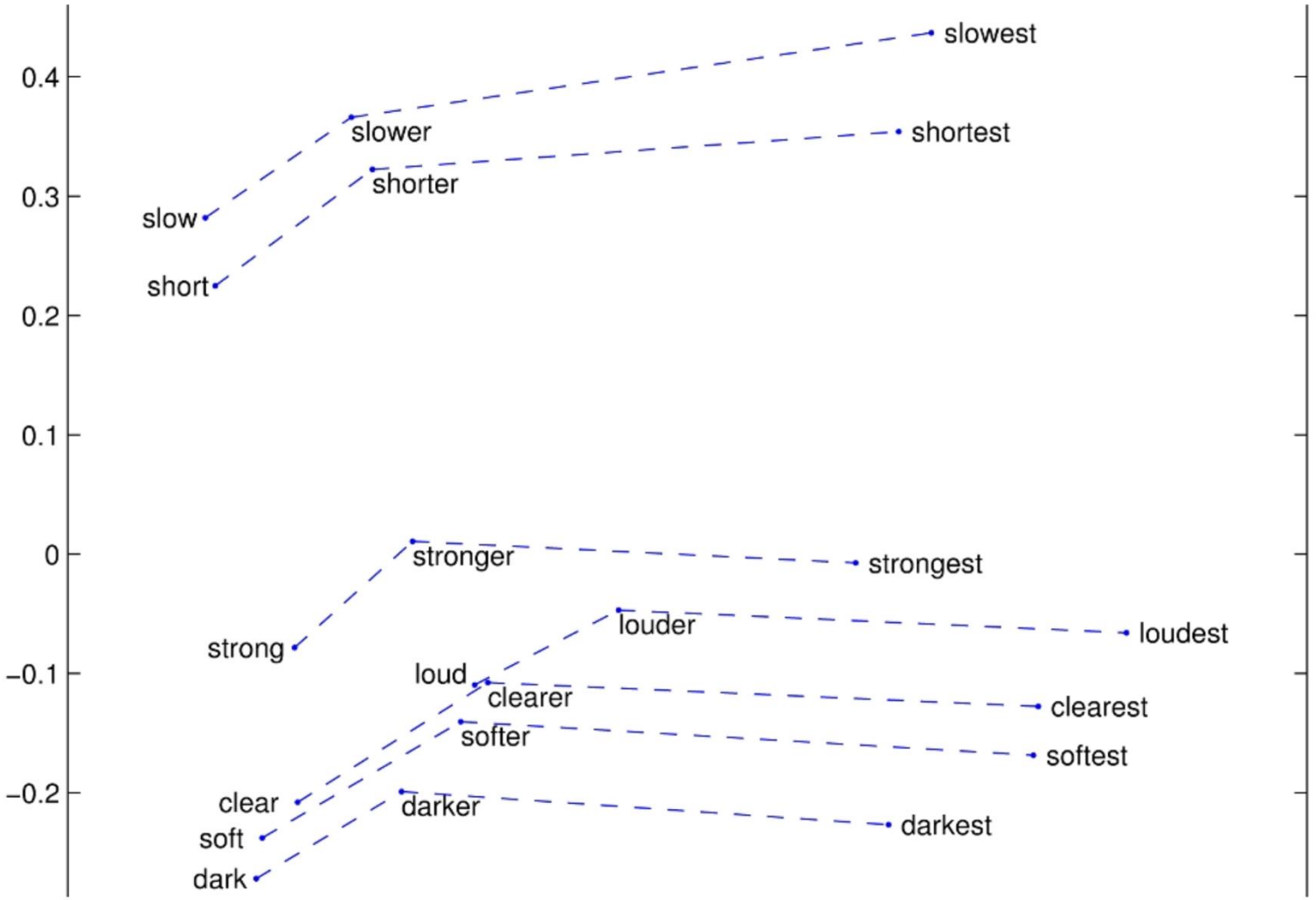
$$\arg\max_{w}\left(\cos(\mathbf{u}_w, \mathbf{u}_{woman} - \mathbf{u}_{man} + \mathbf{u}_{king})\right)$$



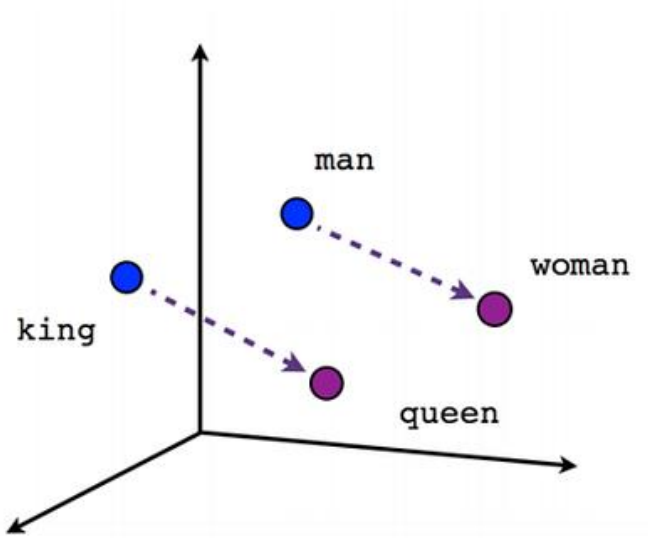
Word Analogy



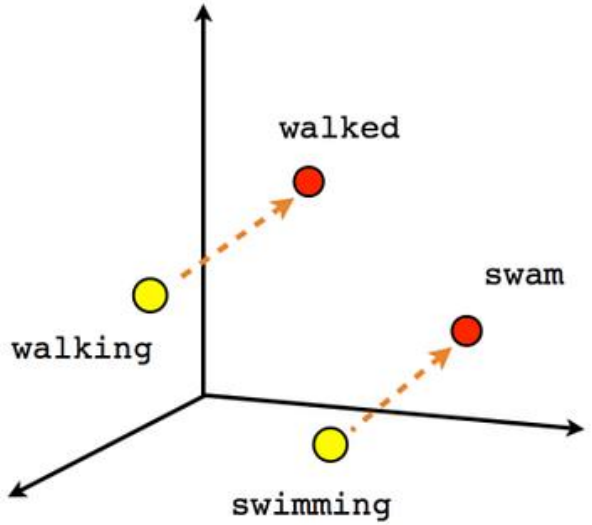
Word Analogy



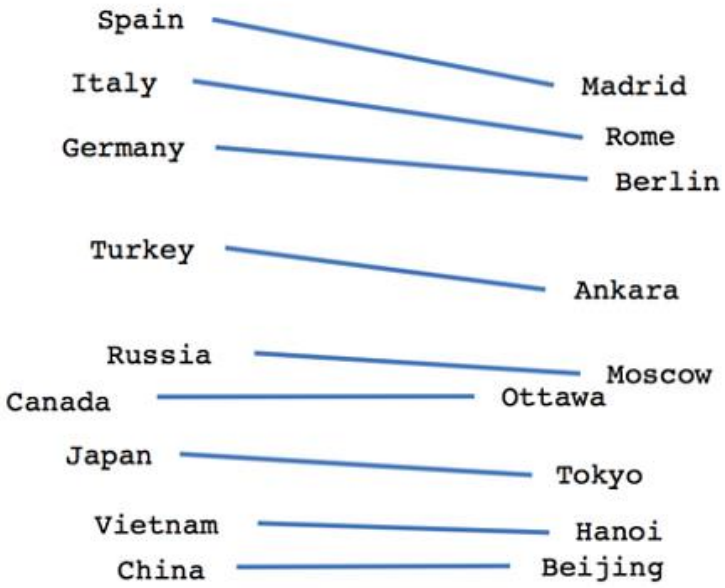
Word Analogy



Male-Female



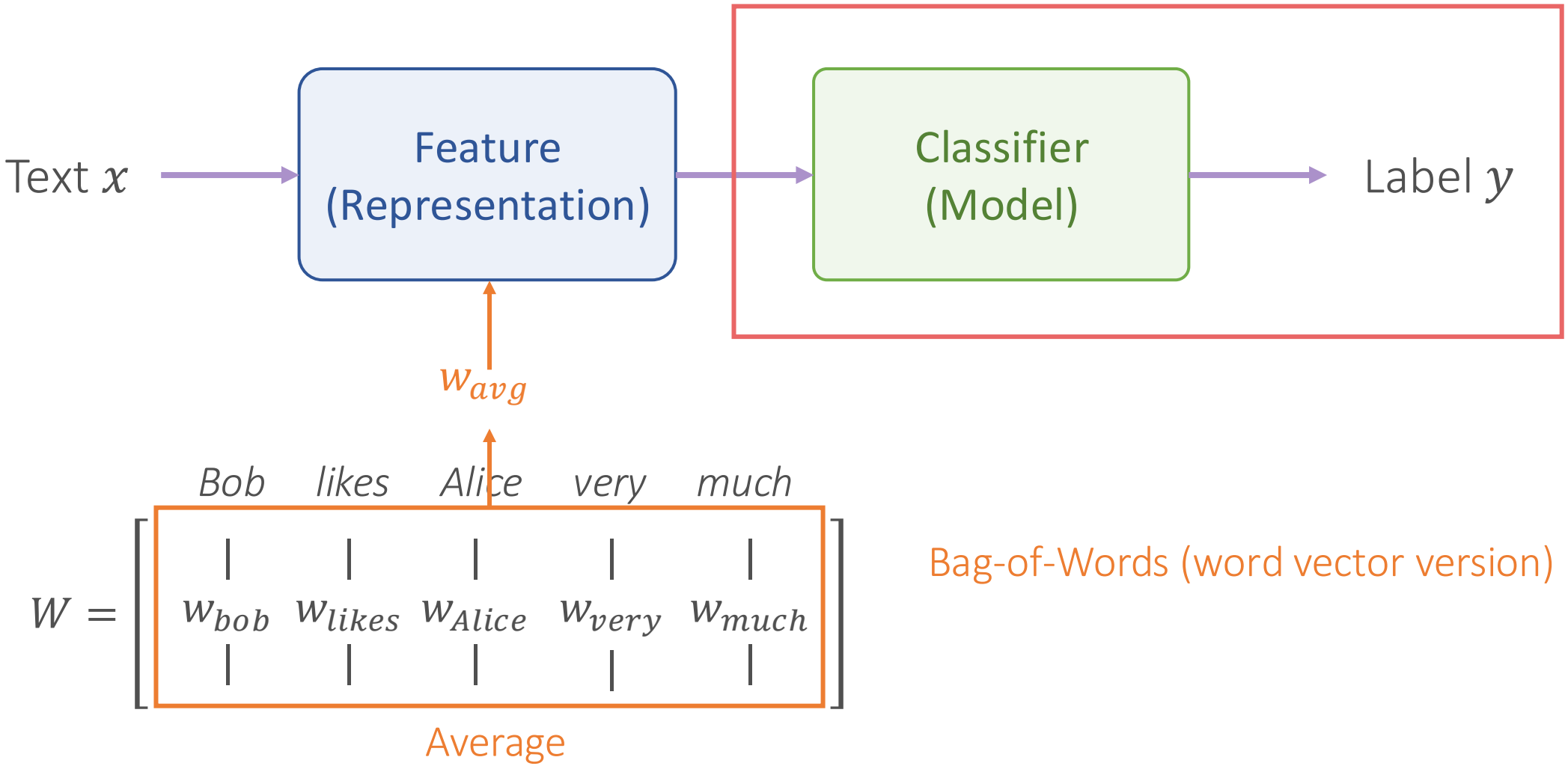
Verb tense



Country-Capital

https://sanjayc.medium.com/word2vec-analogical-reasoning-d47d3a66b9fb

Bag-of-Words (Word Vector Version)



Question?