Natural Language Processing (CS-472) Spring-2023

Muhammad Naseer Bajwa
Assistant Professor,
Department of Computing, SEECS
Co-Principal Investigator,
Deep Learning Lab, NCAI
NUST, Islamabad
naseer.bajwa@seecs.edu.pk

Overview of this week's lecture

Word Representations

- Localist Representation
- Distributional Semantics
- word2vec embeddings
- GloVe embeddings

Denotation

Home: A place to live in

Childish: Like a child

Plant: A manufacturing facility, Photosynthetic organisms, An action of putting into place

Mostly Used in Formal Communication

Every word in natural languages has a Denotation and a Connotation

Denotation

Home: A place to live in

Childish: Like a child

Plant: A manufacturing facility, Photosynthetic organisms, An action of putting into place

Mostly Used in Formal Communication

Connotation

Home: Security, Family, Shelter

Childish: Innocent, Stupid, Immature

Plant: Colonise, Conceal

Mostly Used in Poetry and Literature

How do computers understand the meaning of a word?

- By using **WordNet**: A thesaurus containing lists of synonyms and hypernyms.

https://wordnet.princeton.edu/

- By using **WordNet**: A thesaurus containing lists of synonyms and hypernyms.

Synonyms

Good:

noun,	Goodness
noun,	Commodity
adj,	Good
adj,	Honourable
adj,	Beneficial
adv,	Well
adv,	Thoroughly

Hypernyms

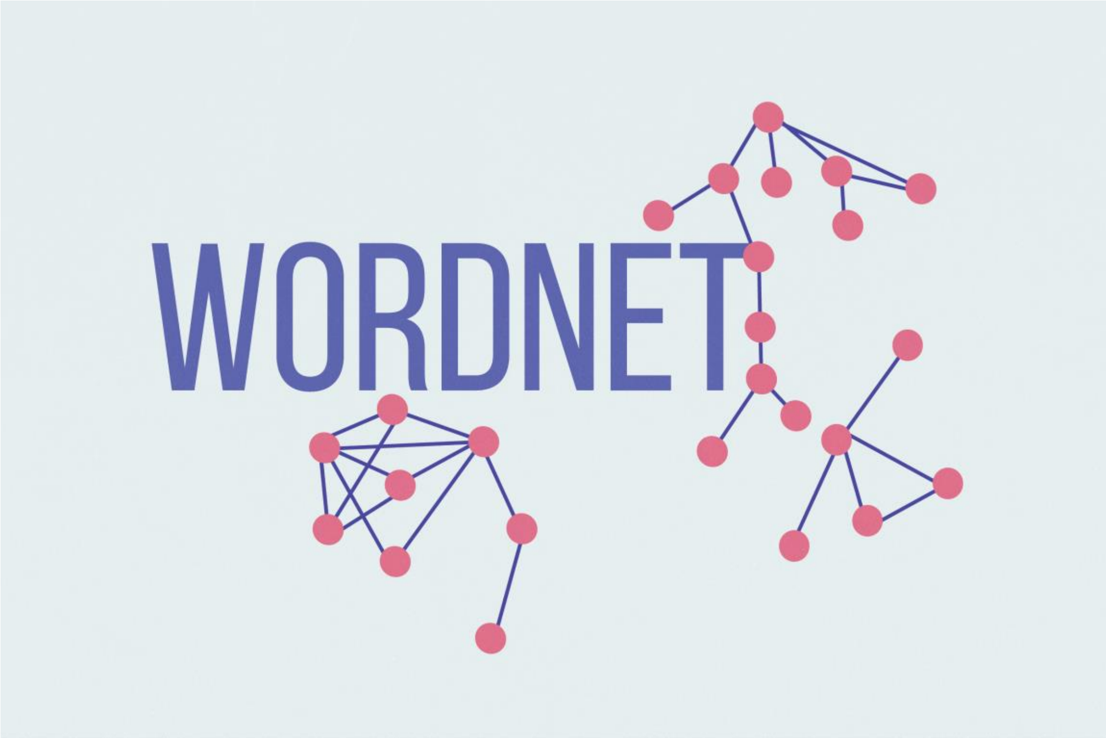
Panda:

- Animal
- Carnivores
- Mammal
- Physical Entity
- Living Thing
- Placental
- Vertebrate

https://wordnet.princeton.edu/

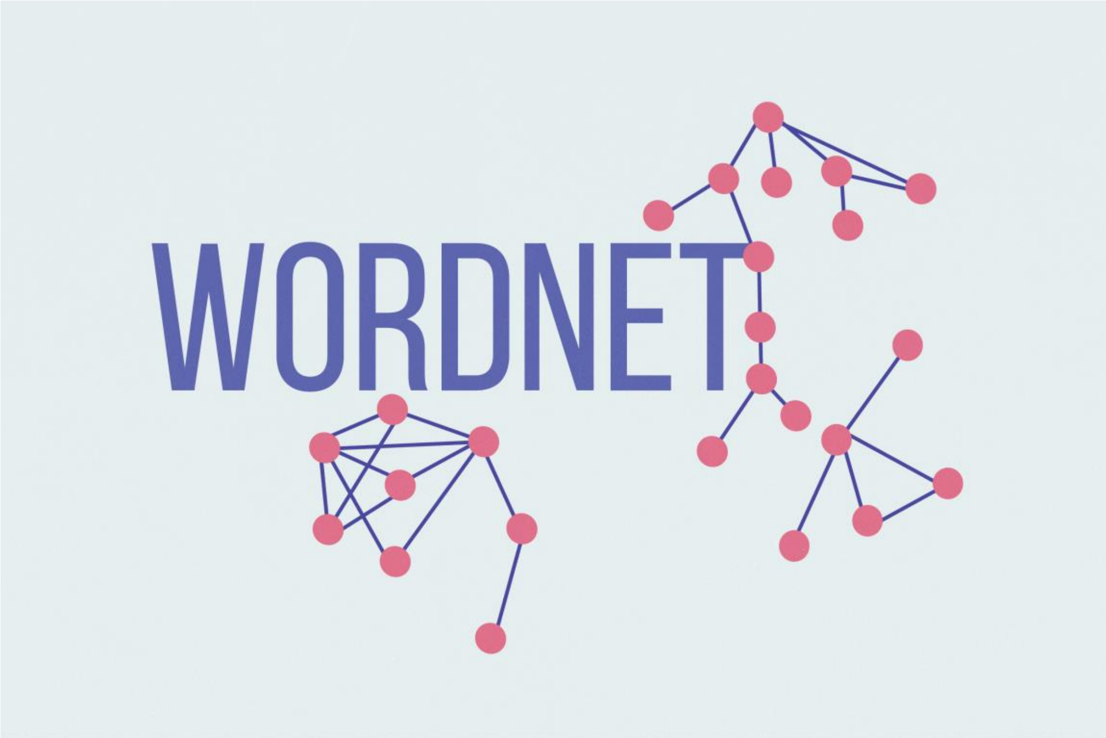
WordNet is like Swiss Army Knife of NLP.

- Missing new meaning of words.
    - Wicked, Badass, Wizard, Ninja
    - Very difficult to keep up-to-date



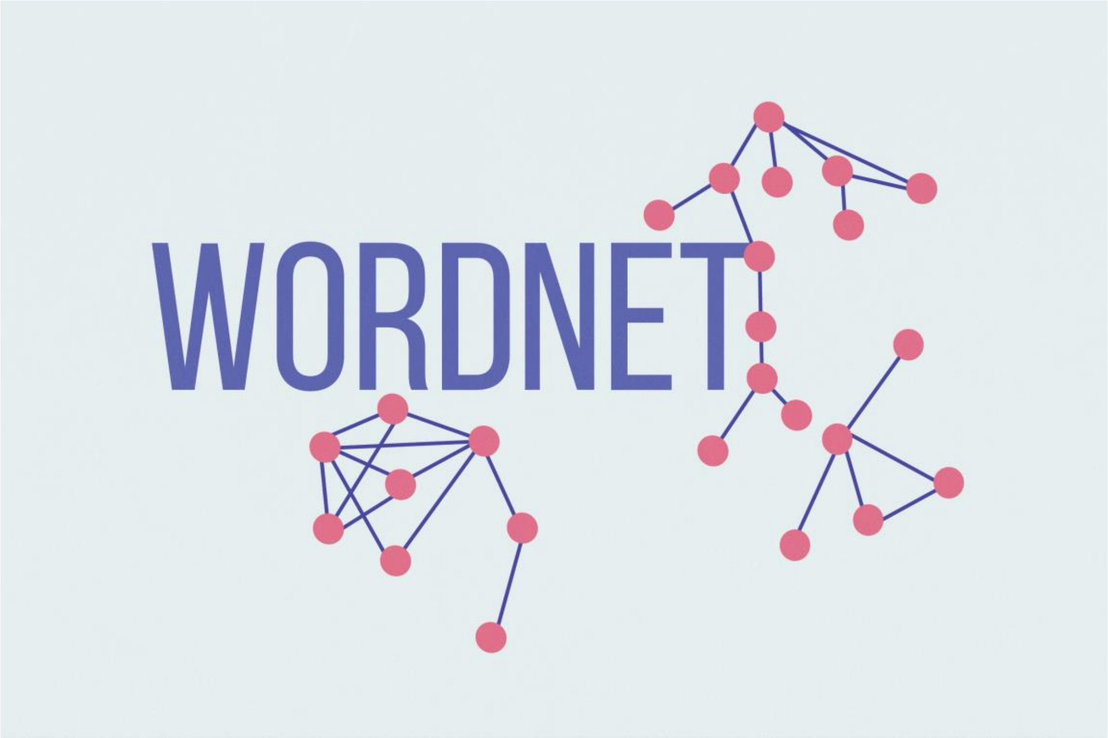
WordNet is like Swiss Army Knife of NLP.

- Missing new meaning of words.
    - Wicked, Badass, Wizard, Ninja
    - Very difficult to keep up-to-date
- Subjective.
    - Therefore, incomplete



WordNet is like Swiss Army Knife of NLP.

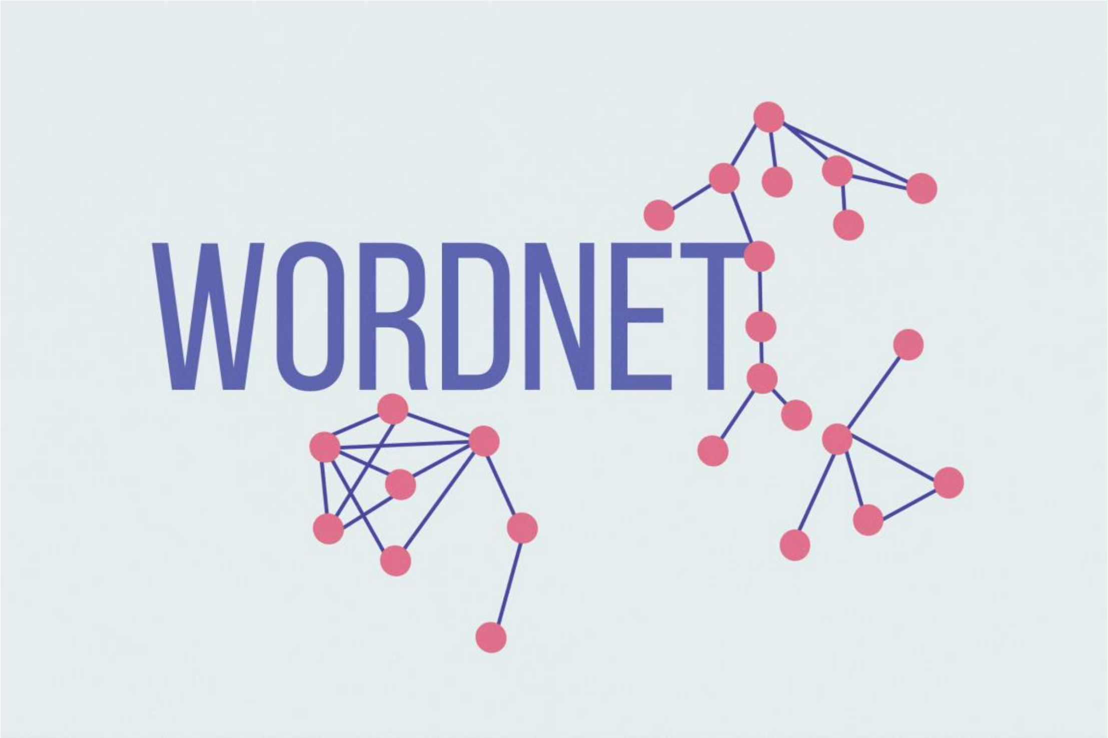
- Missing new meaning of words.
    - Wicked, Badass, Wizard, Ninja
    - Very difficult to keep up-to-date
- Subjective.
    - Therefore, incomplete
- Requires manual labour to curate.
    - What’s the purpose of AI, then?



NUST
Defining futures
School of Electrical Engineering
& Computer Science

WordNet is like Swiss Army Knife of NLP.

- Missing new meaning of words.
    - Wicked, Badass, Wizard, Ninja
    - Very difficult to keep up-to-date
- Subjective.
    - Therefore, incomplete
- Requires manual labour to curate.
    - What’s the purpose of AI, then?
- Can’t compute accurate word similarity.
    - No partial resemblance
    - Has only fixed synonym set



Computers cannot understand words

- **Localist Representation:** Consider words as discrete symbols.
  - It was practiced in traditional NLP (up until 2012).
  - Example: One-Hot Encoding

Leopard	1	0	0	0	0	0
Sofa	0	1	0	0	0	0
Spider	0	0	1	0	0	0
Panther	0	0	0	1	0	0
Chair	0	0	0	0	1	0
Give	0	0	0	0	0	1

Computers cannot understand words

- **Localist Representation:** Consider words as discrete symbols.
  - It was practiced in traditional NLP (up until 2012).
  - Example: One-Hot Encoding
- What are the problems with Localist Representation?
  - Very large vectors (due to deviational morphology)
  - Each vector is orthogonal to all others (what does it mean?)
  - Sparse matrix

Leopard	1	0	0	0	0	0
Sofa	0	1	0	0	0	0
Spider	0	0	1	0	0	0
Panther	0	0	0	1	0	0
Chair	0	0	0	0	1	0
Give	0	0	0	0	0	1

Computers cannot understand words

- **Localist Representation:** Consider words as discrete symbols.
    - It was practiced in traditional NLP (up until 2012).
    - Example: One-Hot Encoding
- What are the problems with Localist Representation?
    - Very large vectors (due to deviational morphology)
    - Each vector is orthogonal to all others (what does it mean?)
    - Sparse matrix

Leopard	1	0	0	0	0	0	0	0	0
Sofa	0	1	0	0	0	0	0	0	0
Spider	0	0	1	0	0	0	0	0	0
Panther	0	0	0	1	0	0	0	0	0
Chair	0	0	0	0	1	0	0	0	0
Give	0	0	0	0	0	1	0	0	0
Gave	0	0	0	0	0	0	1	0	0
Gives	0	0	0	0	0	0	0	1	0
Given	0	0	0	0	0	0	0	0	1

How to find similarity between two words?

- Maybe rely on WordNet’s list of synonyms to get similarity?

How to find similarity between two words?

- Maybe rely on WordNet's list of synonyms to get similarity?
    - Has been tried already. Failed miserably because of incompleteness.

How to find similarity between two words?

- Maybe rely on WordNet's list of synonyms to get similarity?
  - Has been tried already. Failed miserably because of incompleteness.
- Pre-calculate similarity of each word with every other word and put it in a table?

How to find similarity between two words?

- Maybe rely on WordNet’s list of synonyms to get similarity?
    - Has been tried already. Failed miserably because of incompleteness.
- Pre-calculate similarity of each word with every other word and put it in a table?
    - Google did that. The table size grows exponentially.

How to find similarity between two words?

- Maybe rely on WordNet's list of synonyms to get similarity?
  - Has been tried already. Failed miserably because of incompleteness.
- Pre-calculate similarity of each word with every other word and put it in a table?
  - Google did that. The table size grows exponentially.
- Learn to encode similarity in the word representation.

How to find similarity between two words?

- Maybe rely on WordNet’s list of synonyms to get similarity?
  - Has been tried already. Failed miserably because of incompleteness.
- Pre-calculate similarity of each word with every other word and put it in a table?
  - Google did that. The table size grows exponentially.
- Learn to encode similarity in the word representation.
  - Smart and efficient

Representing words by their context is better than localist representation

- **Distributional Semantics:** A word’s intended meaning is determined by the context it appears in.

"You shall know a word by the company it keeps"
(J.R. Firth, 1957)

John R. Firth
(1890 - 1960)

Representing words by their context is better than localist representation

- **Distributional Semantics:** A word’s intended meaning is determined by the context it appears in.

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

  (J.R. Firth, 1957)

- What would be the size of this context?

John R. Firth
(1890 - 1960)

Representing words by their context is better than localist representation

- **Distributional Semantics:** A word's intended meaning is determined by the context it appears in.

  "You shall know a word by the company it keeps"

  (J.R. Firth, 1957)

- What would be the size of this context?

  - Fixed-sized window.

John R. Firth
(1890 - 1960)

Representing words by their context is better than localist representation

- **Distributional Semantics:** A word's intended meaning is determined by the context it appears in.

  "You shall know a word by the company it keeps"
  (J.R. Firth, 1957)

- What would be the size of this context?
  - Fixed-sized window.

- Use many contexts of a word w to make representation of w.
  - The prime minister urged the nation to plant a tree to preserve environment.
  - The prime minister inaugurated a plant to manufacture cars.
  - The prime minister emphasised that the research in plant biology is important.
  - The prime minister cautioned that the opposition may plant fake news about him.

John R. Firth
(1890 - 1960)

Represent each word by a vector of real numbers

- Makes a smaller and dense vector for each word, such that it's similar to the vectors of words that co-occur in similar context.

$$Plant = \begin{bmatrix} +0.285 \\ -0.188 \\ +0.892 \\ -0.109 \\ -0.349 \\ +0.543 \\ +0.271 \\ +0.018 \end{bmatrix}$$

Represent each word by a vector of real numbers

- Makes a smaller and dense vector for each word, such that it’s similar to the vectors of words that co-occur in similar context.

- The example word vector for the word ‘plant’ is an 8-dimentional vector.

    - In practice, the dimensions of this vector can range from hundreds to thousands.

$$Plant = \begin{bmatrix} +0.285 \\ -0.188 \\ +0.892 \\ -0.109 \\ -0.349 \\ +0.543 \\ +0.271 \\ +0.018 \end{bmatrix}$$

Represent each word by a vector of real numbers

- Makes a smaller and dense vector for each word, such that it’s similar to the vectors of words that co-occur in similar context.

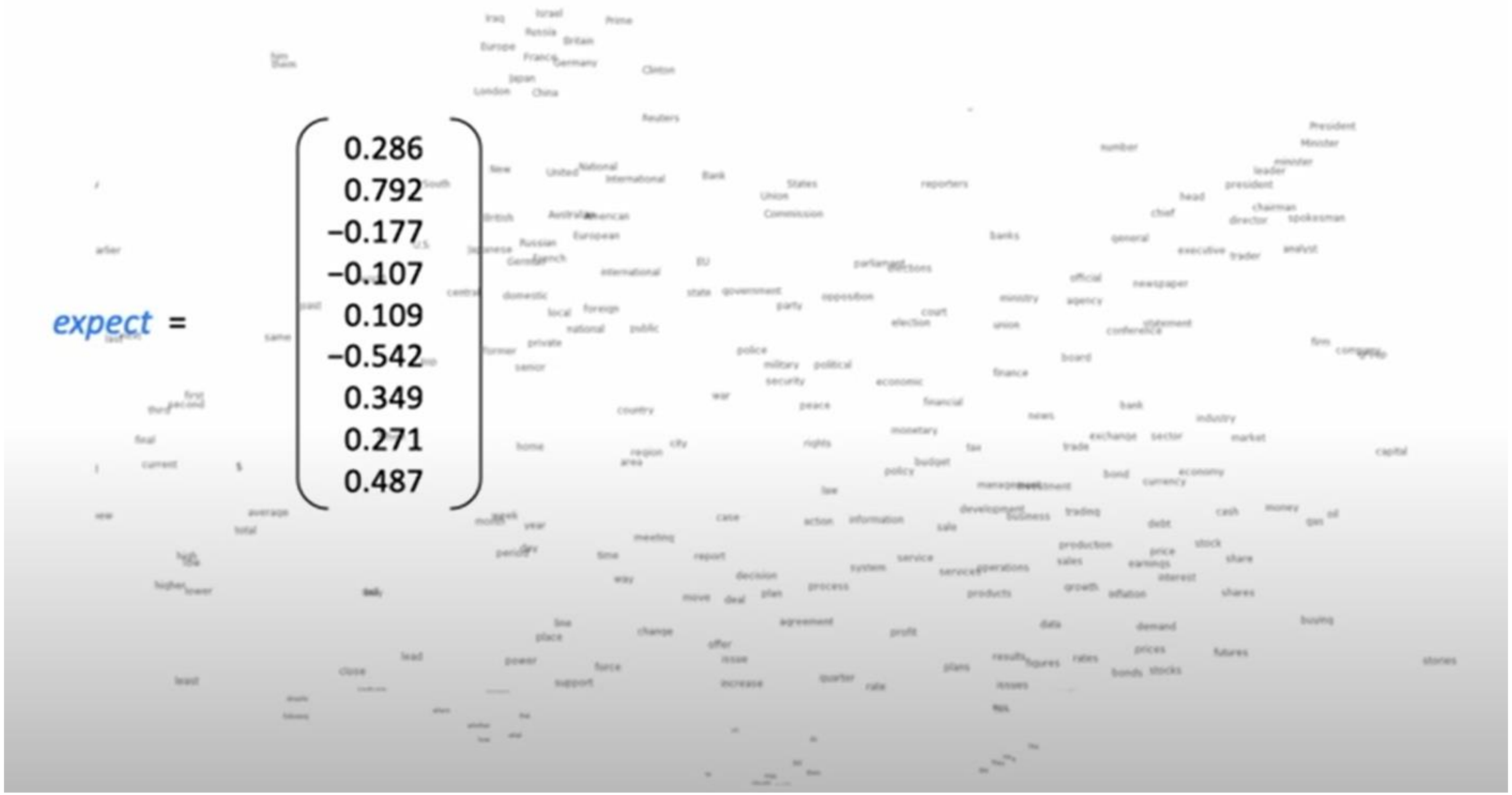
- The example word vector for the word ‘plant’ is an 8-dimentional vector.

    - In practice, the dimensions of this vector can range from hundreds to thousands.

- These word vectors are also called word embeddings or word representations.

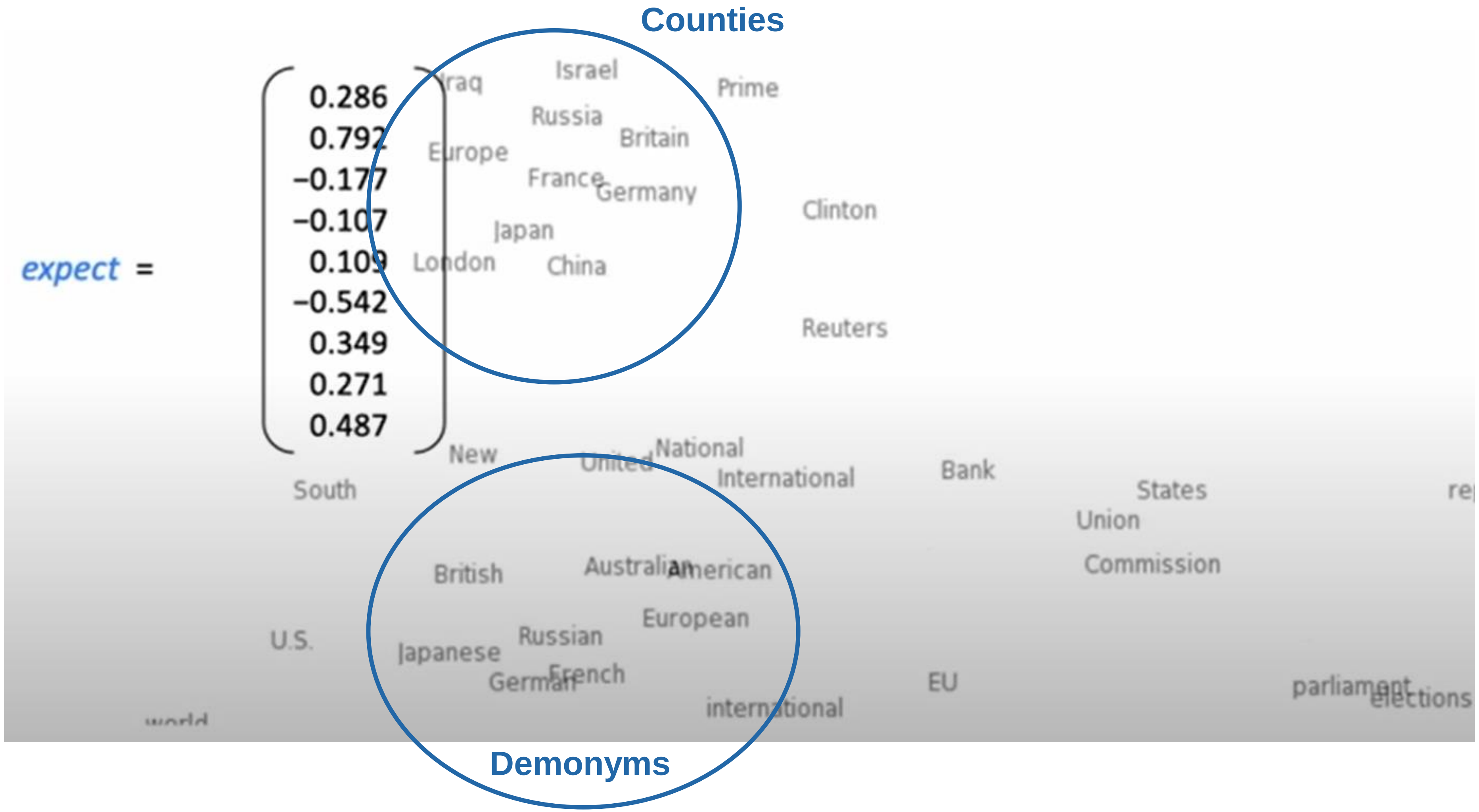
$$Plant = \begin{bmatrix} +0.285 \\ -0.188 \\ +0.892 \\ -0.109 \\ -0.349 \\ +0.543 \\ +0.271 \\ +0.018 \end{bmatrix}$$

Word embeddings can be visualised in vector space

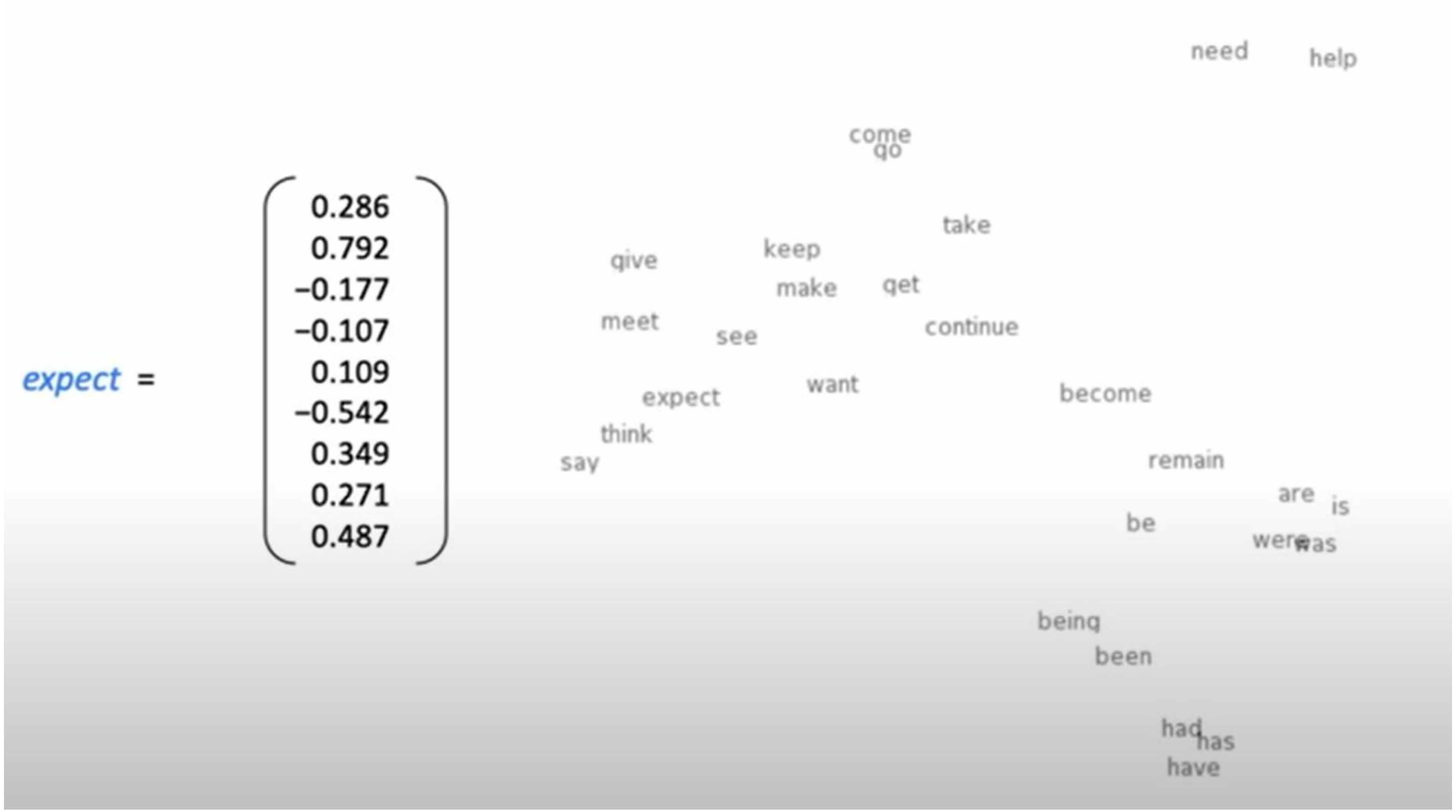


NUST
Defining futures
School of Electrical Engineering
& Computer Science

Word embeddings can be visualised in vector space

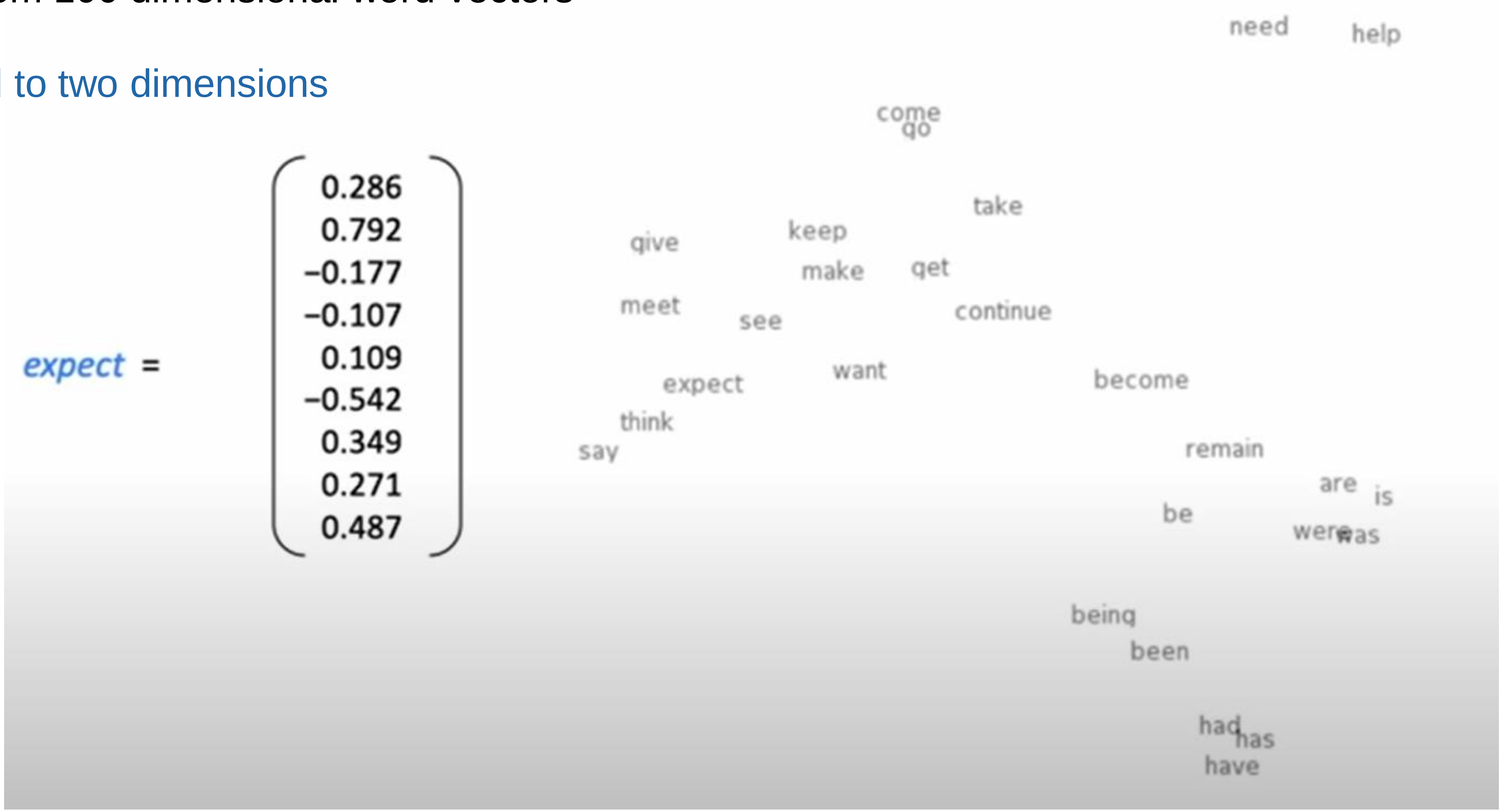


Word embeddings can be visualised in vector space



Word embeddings can be visualised in vector space

- Generated from 100 dimensional word vectors
  - Reduced to two dimensions



NUST
Defining futures
School of Electrical Engineering
& Computer Science

word2vec is a famous algorithm used to create word vectors

- Have a large corpus of text.
- Represent each word in a fixed vocabulary by a vector (already?).
- Go through each position t in the text that has a centre word c and a context word o.

https://arxiv.org/pdf/1301.3781.pdf

word2vec is a famous algorithm used to create word vectors

- Have a large corpus of text.
- Represent each word in a fixed vocabulary by a vector (already?).
- Go through each position t in the text that has a centre word c and a context word o.
- Use the similarity of the word vectors c and o to calculate the probability $P(o|c)$, or vice versa.
- Keep updating the word vectors to maximise this probability.

https://arxiv.org/pdf/1301.3781.pdf

How to compute the probability of a context word given a centre word?

- Process of computing $P(w_{t+j}|w_t)$, where j is the size of context window:

$\boldsymbol{For\ j = 1}$

The prime **minister** inaugurated a pant to manufacture cars.

Centre word w at position t, (w_t)

NUST Defining futures
School of Electrical Engineering & Computer Science

How to compute the probability of a context word given a centre word?

- Process of computing $P(w_{t+j}|w_t)$, where j is the size of context window:

$\boldsymbol{For\ j = 1}$

$P(w_{t-1}|w_t)$ $P(w_{t+1}|w_t)$

The prime **minister** inaugurated a pant to manufacture cars.

Centre word w at
position t, (w_t)

How to compute the probability of a context word given a centre word?

- Process of computing $P(w_{t+j}|w_t)$, where j is the size of context window:

$For\ j = 2$

$P(w_{t-2}|w_t)$ $P(w_{t+2}|w_t)$

$P(w_{t-1}|w_t)$ $P(w_{t+1}|w_t)$

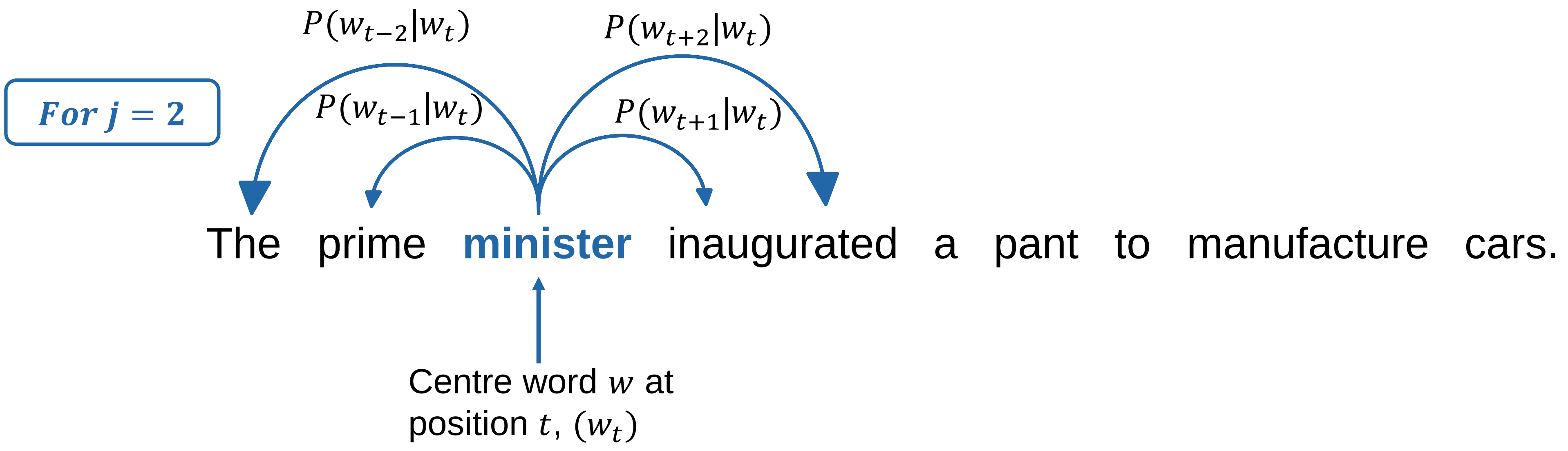
The prime **minister** inaugurated a pant to manufacture cars.

Centre word w at position t, (w_t)

- To know the meaning of the word minister, we predict what words come in context of the word minister.

How to compute the probability of a context word given a centre word?

- Process of computing $P(w_{t+j}|w_t)$, where j is the size of context window:



- To know the meaning of the word minister, we predict what words come in context of the word minister.

- Evaluate the predictions to update the representations of the word and try again.

How to compute the probability of a context word given a centre word?

- Process of computing $P(w_{t+j}|w_t)$, where j is the size of context window:

$\boxed{\boldsymbol{For\ j = 2}}$

$P(w_{t-2}|w_t)$ $P(w_{t+2}|w_t)$

$P(w_{t-1}|w_t)$ $P(w_{t+1}|w_t)$

The prime minister **inaugurated** a pant to manufacture cars.

Centre word w at position t, (w_t)

Once the representation of minister is done, move on to the next word.

- To know the meaning of the word minister, we predict what words come in context of the word minister.

- Evaluate the predictions to update the representations of the word and try again.

The updates in word vectors in made using Objective Function

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

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

  - θ represents all optimisable parameters.

The updates in word vectors in made using Objective Function

- For each position $t = 1, \dots, T$, predict context words within a window of fixed size m, given centre word w_t.

$$\text{Likielihood} = L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \ne 0}} P(w_{t+j} | w_t; \theta)$$

  - θ represents all optimisable parameters.

- The objective function is the **negative log likelihood**.

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

  - Minimising this objective function results in maximising predictive accuracy.

The updates in word vectors in made using Objective Function

- For each position $t = 1, \dots, T$, predict context words within a window of fixed size m, given centre word w_t.

$$\text{Likielihood} = L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \neq 0}} P(w_{t+j} | w_t; \theta)$$

  - θ represents all optimisable parameters.

- The objective function is the **negative log likelihood**.

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

  - Minimising this objective function results in maximising predictive accuracy.

- How to calculate $P(w_{t+j}|w_t)$?

  - Go to the next slide.

How to calculate $P(w_{t+j}|w_t)$?

- Every word w has two representations,
  - v_w as a centre word
  - u_w as context word.

How to calculate $P(w_{t+j}|w_t)$?

- Every word w has two representations,

  - v_w as a centre word
  - u_w as context word.

- For each centre word c and context word o, calculate

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

$\exp$ makes everything positive.
$u_o^T v_c$ is just dot product of the two vectors. (what does dot products do?)
Normalise over entire vocabulary. (why?)

How to calculate $P(w_{t+j}|w_t)$?

- Every word w has two representations,

  - v_w as a centre word
  - u_w as context word.

- For each centre word c and context word o, calculate

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

  $\exp$ makes everything positive.
  $u_o^T v_c$ is just dot product of the two vectors. (what does dot products do?)
  Normalise over entire vocabulary. (why?)

- Does this function ring any bells?

How to calculate $P(w_{t+j}|w_t)$?

- Every word w has two representations,

    - v_w as a centre word
    - u_w as context word.

- For each centre word c and context word o, calculate

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

$\exp$ makes everything positive.
$u_o^T v_c$ is just dot product of the two vectors. (what does dot products do?)
Normalise over entire vocabulary. (why?)

- Does this function ring any bells?

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

    - $softmax$ amplifies probability of the largest x_i, while still assigning some probability to other classes.

Train the model using any optimisation algorithm

- Training a model **simply** means finding (optimising) suitable parameters of the model to minimize the cost function.

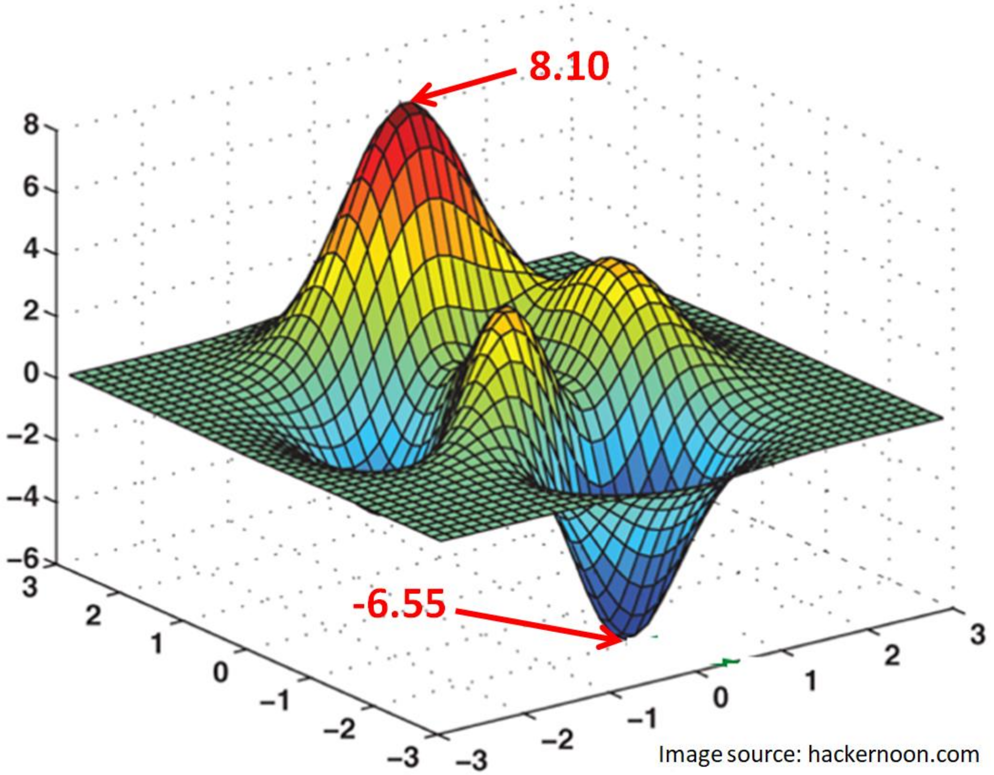


Image source: hackernoon.com

Train the model using any optimisation algorithm

- Training a model **simply** means finding (optimising) suitable parameters of the model to minimize the cost function.

- In practice,

    - Evaluate the model with any given weights θ (initially random)
    - Find the difference between evaluated output and desired output using objective function
    - Calculate gradient with respect to each weight
    - Follow the slop (up or down?) to find optimal parameters.

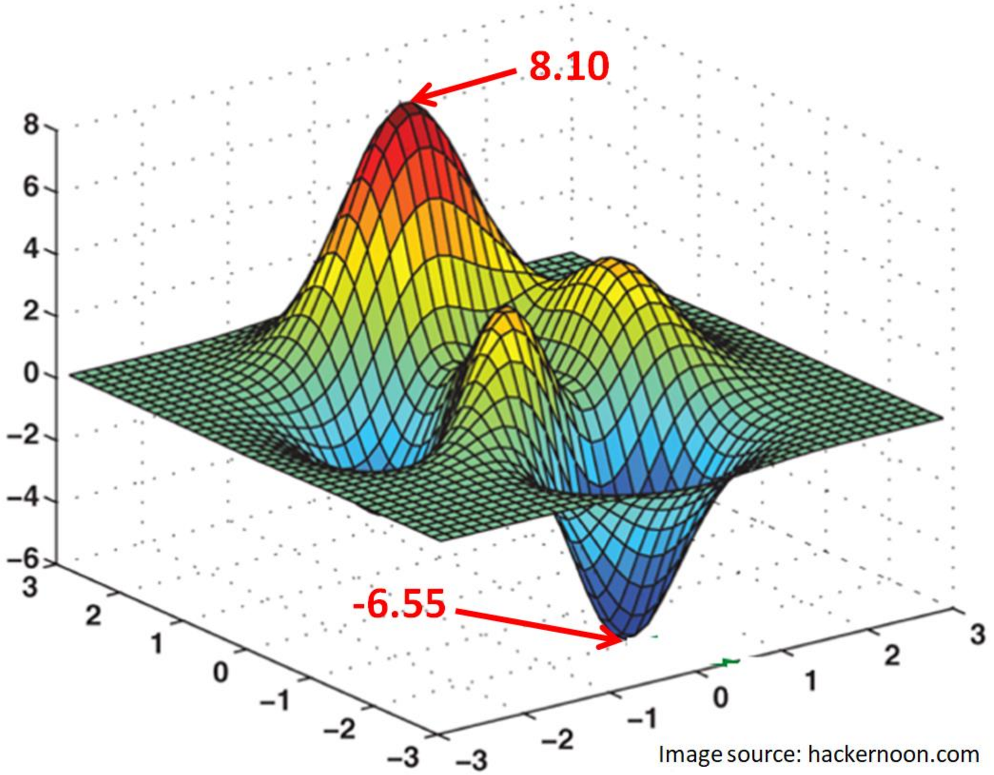


Image source: hackernoon.com

Train the model using any optimisation algorithm

- Training a model **simply** means finding (optimising) suitable parameters of the model to minimize the cost function.

- In practice,

    - Evaluate the model with any given weights θ (initially random)
    - Find the difference between evaluated output and desired output using objective function
    - Calculate gradient with respect to each weight
    - Follow the slop (up or down?) to find optimal parameters.

- With d-dimensional vector for each word and V-dimensional vocabulary, θ represents a vector of the dimensions $\mathbb{R}^{2dV}$.

    - θ consists of the contents of word vectors.

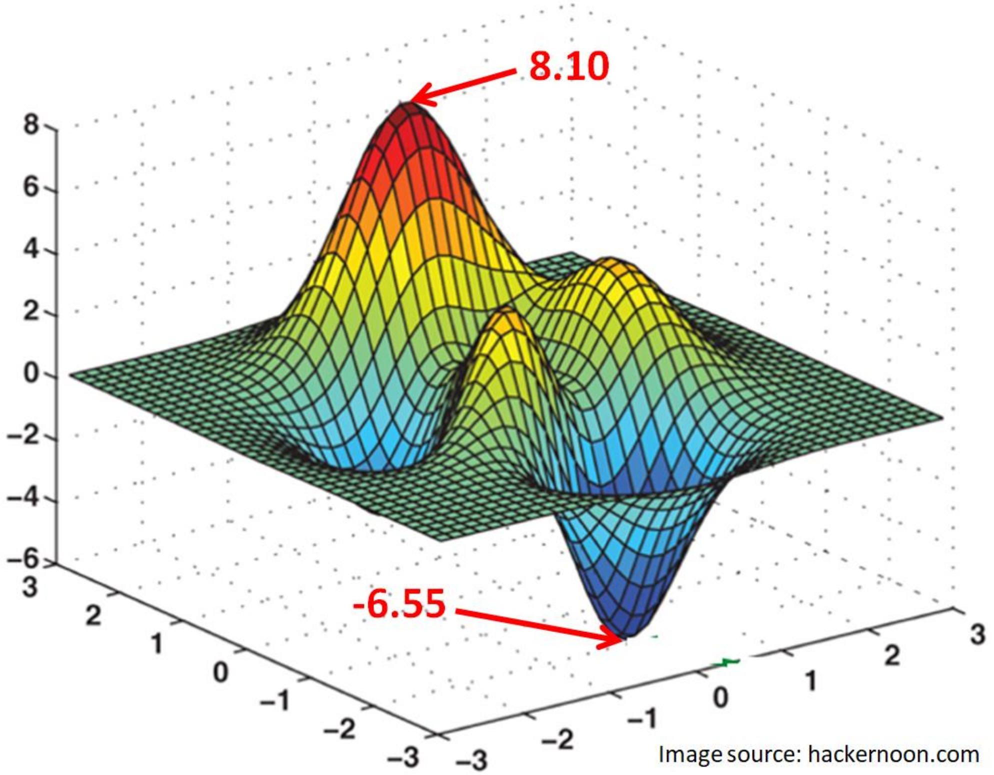


Image source: hackernoon.com

Additional details on word2vec

- Why we have two vectors u_w and v_w for each word?
  - Easy to optimise. But what to do with the second vector after optimisation?
  - One vector for each word from the outset can also work.
- There are two main implementations of word2vec algorithm.
  - Skip-Gram (SG): Predicts context words given centre word.
  - Bag of Words (BOW): Predicts centre word from (a bag of) context words
  - Which model we discussed?

Vectorise the data to speed up computations

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

$$\begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \end{bmatrix} \quad \begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \end{bmatrix}$$

U

Outside Words

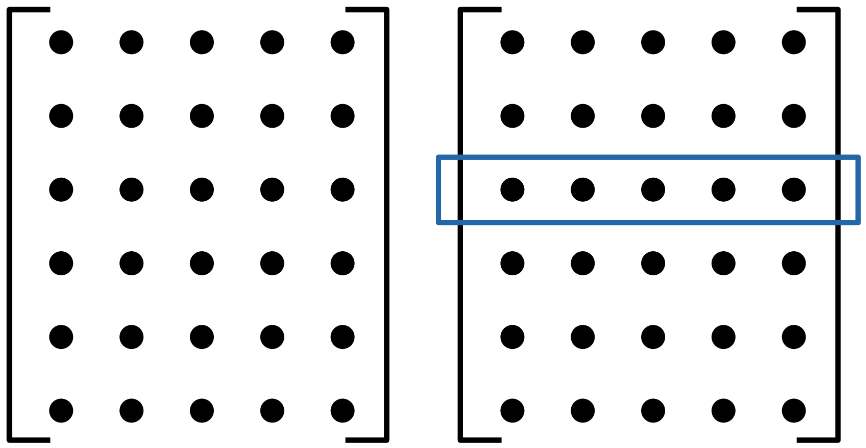
V

Centre Words

Vectorise the data to speed up computations

- Each row in U and V corresponds to a word.

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



U Outside Words

V Centre Words

Vectorise the data to speed up computations

- Each row in U and V corresponds to a word.

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

$$\begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \end{bmatrix} \quad \begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \boxed{\bullet \;\; \bullet \;\; \bullet \;\; \bullet \;\; \bullet} \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \end{bmatrix} \quad \begin{bmatrix} \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \end{bmatrix}$$

U
Outside Words

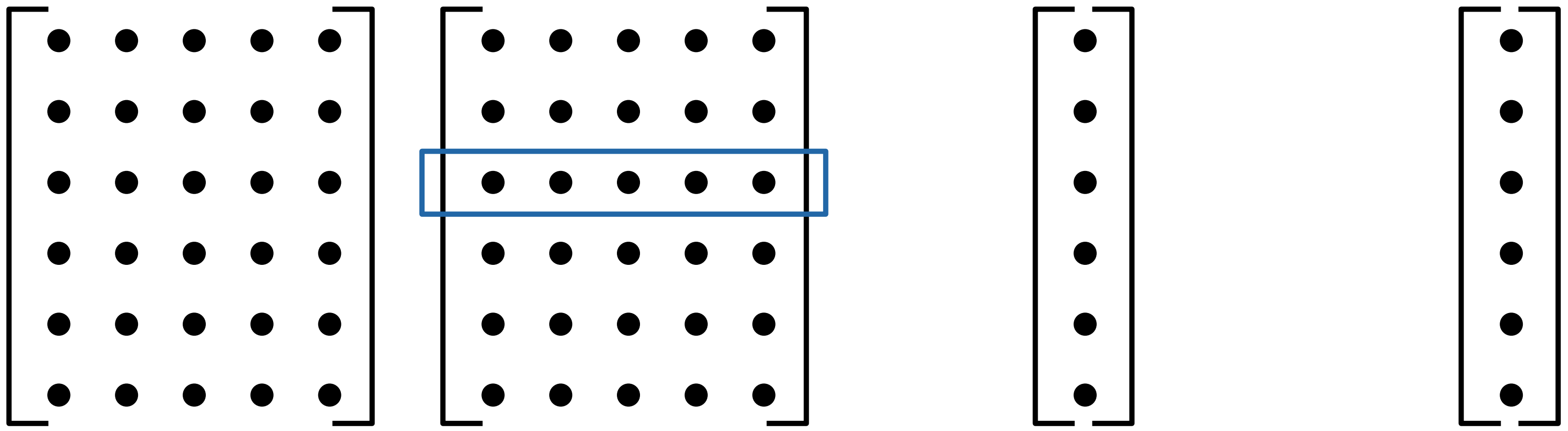
V
Centre Words

$U . v_i^T$
Vector of Dot Products

Vectorise the data to speed up computations

- Each row in U and V corresponds to a word.

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



U
Outside Words

V
Centre Words

$U.v_i^T$
Vector of Dot Products

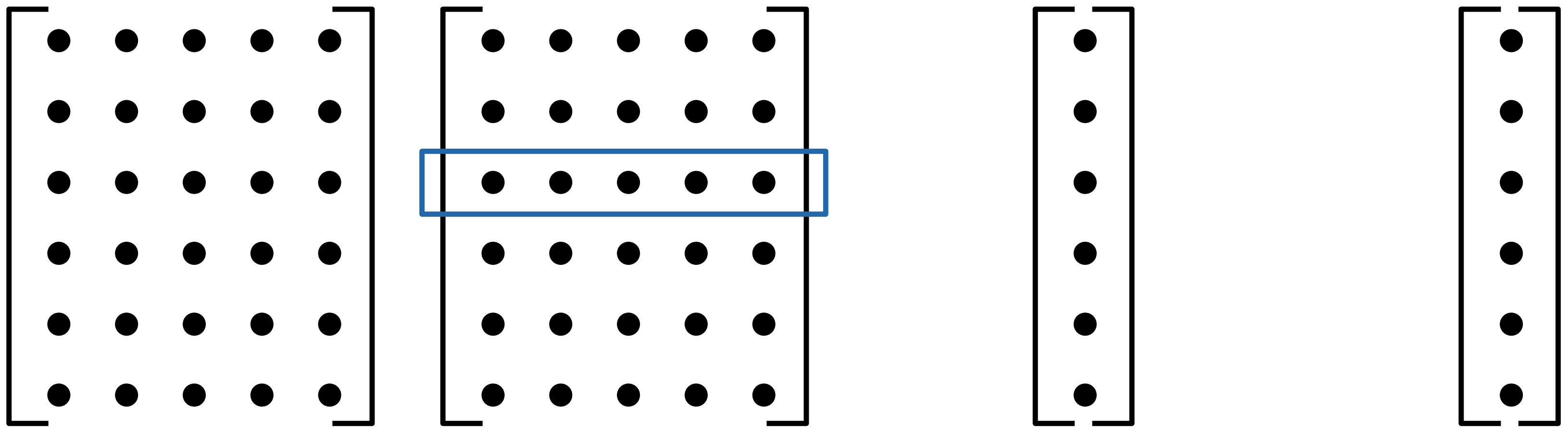
$Softmax(U.v_i^T)$
Probability Distribution

Vectorise the data to speed up computations

- Each row in U and V corresponds to a word.

- The $softmax$ distribution may fall victim to high frequency words. How to fix it?

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



U
Outside Words

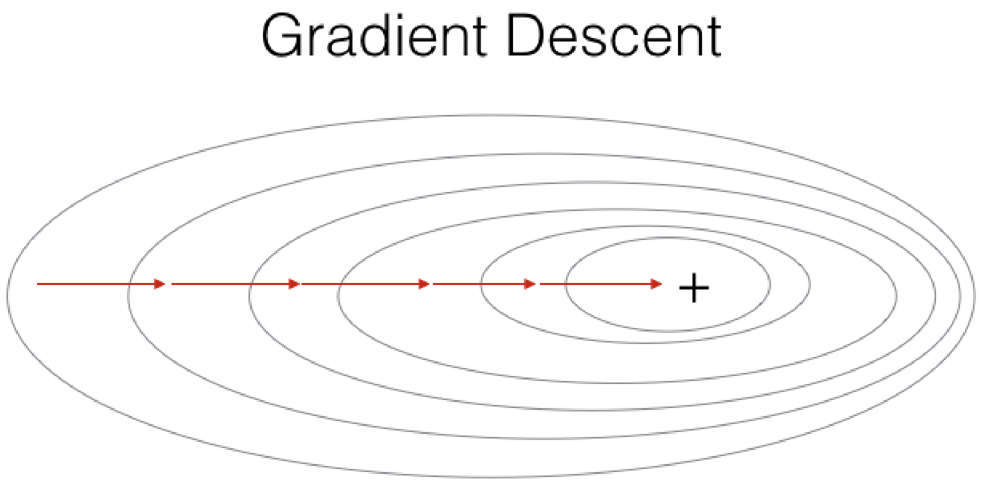
V
Centre Words

$U.v_i^T$
Vector of Dot Products

$Softmax(U.v_i^T)$
Probability Distribution

Gradient Descent is an accurate but slow optimization algorithm

- Training a model **simply** means finding (optimising) suitable parameters of the model to minimize the cost function.



Gradient Descent is an accurate but slow optimization algorithm

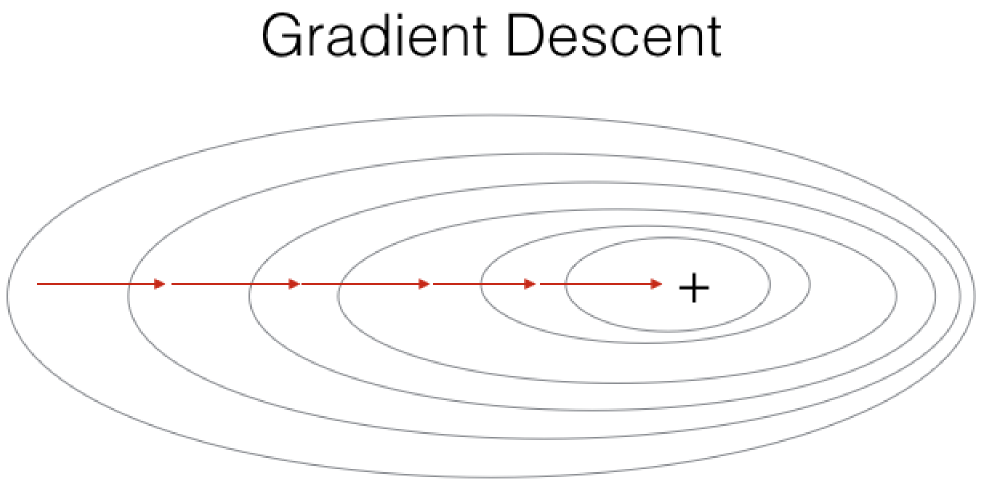
- Training a model **simply** means finding (optimising) suitable parameters of the model to minimize the cost function.

- Update equation for single parameter is,

$$\theta_i^{new} = \theta_i^{old} - \alpha \frac{\partial J(\theta)}{\partial \theta_i^{old}}$$

- Update equation for all parameters is,

$$\theta^{new} = \theta^{old} - \alpha \nabla_\theta J(\theta)$$



Gradient Descent is an accurate but slow optimization algorithm

- Training a model **simply** means finding (optimising) suitable parameters of the model to minimize the cost function.

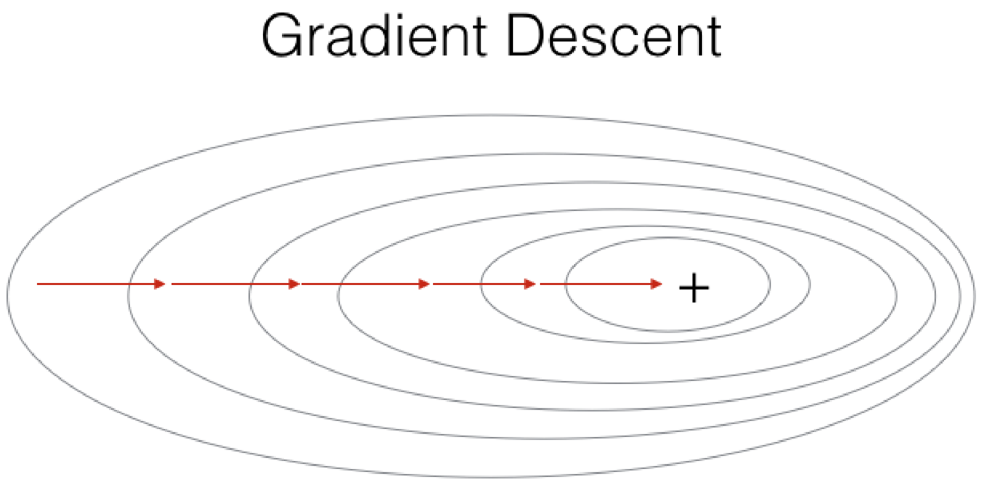
- Update equation for single parameter is,

$$\theta_i^{new} = \theta_i^{old} - \alpha \frac{\partial J(\theta)}{\partial \theta_i^{old}}$$

- Update equation for all parameters is,

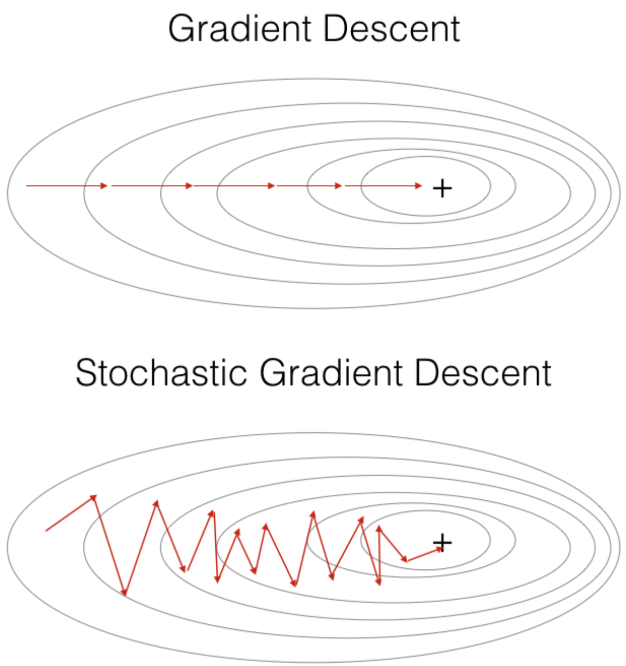
$$\theta^{new} = \theta^{old} - \alpha \nabla_{\theta} J(\theta)$$

- Gradient descent is an old, time-tested but very slow optimisation algorithm.
    - Can we make it faster?



Stochastic Gradient Descent is sloppy but fast

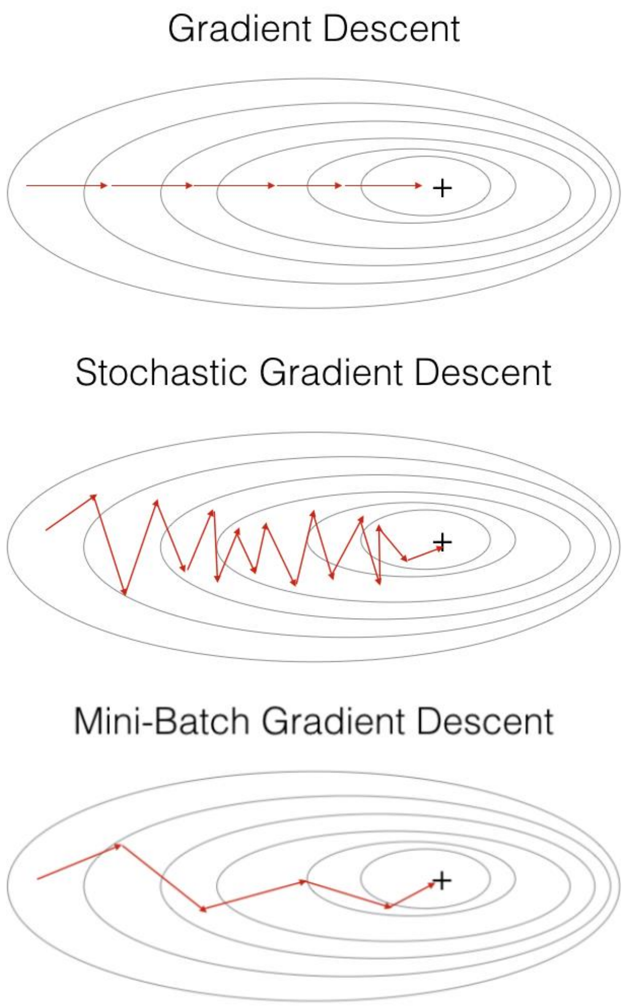
- Less accurate but more time and memory efficient.
    - Can do more with the same resources.
- Updates are made based on a randomly selected window of word.



https://www.youtube.com/watch?v=UmathvAKj80

Stochastic Gradient Descent is sloppy but fast

- Less accurate but more time and memory efficient.
  - Can do more with the same resources.
- Updates are made based on a randomly selected window of word.
- Minibatches can further help smooth the optimisation path of SGD and take advantage of parallelisation on GPUs.

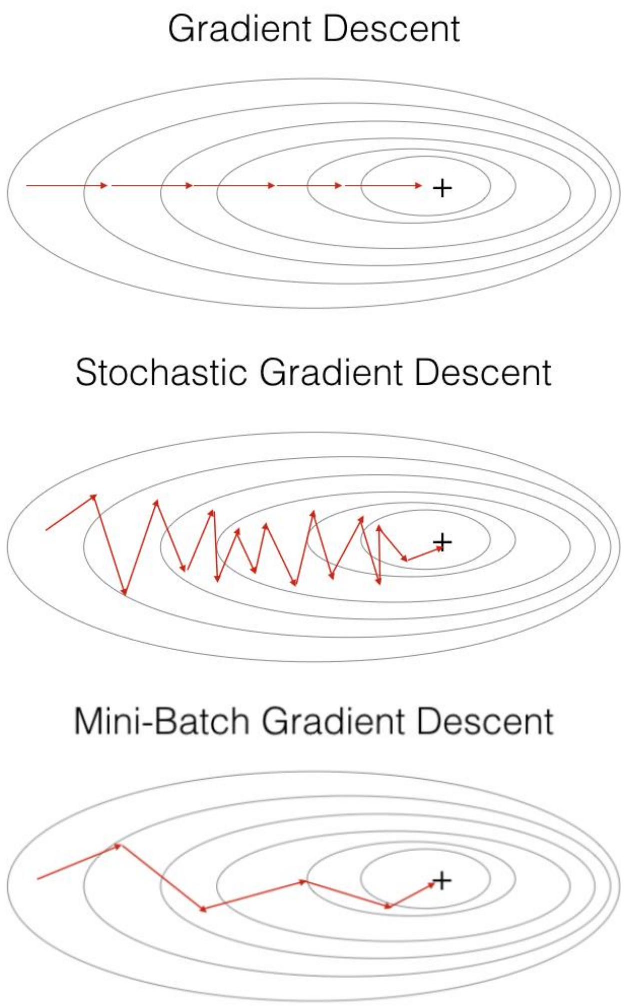


https://www.youtube.com/watch?v=UmathvAKj80

Stochastic Gradient Descent is sloppy but fast

- Less accurate but more time and memory efficient.
  - Can do more with the same resources.
- Updates are made based on a randomly selected window of word.
- Minibatches can further help smooth the optimisation path of SGD and take advantage of parallelisation on GPUs.
- The SGD has the problem of sparsity though.
  - **Solution?**

$$\nabla_\theta J(\theta) = \begin{bmatrix} 0 \\ . \\ . \\ . \\ \nabla v_i \\ 0 \\ . \\ . \\ \nabla u_i \\ 0 \\ . \\ . \end{bmatrix} \in \mathbb{R}^{2dV}$$

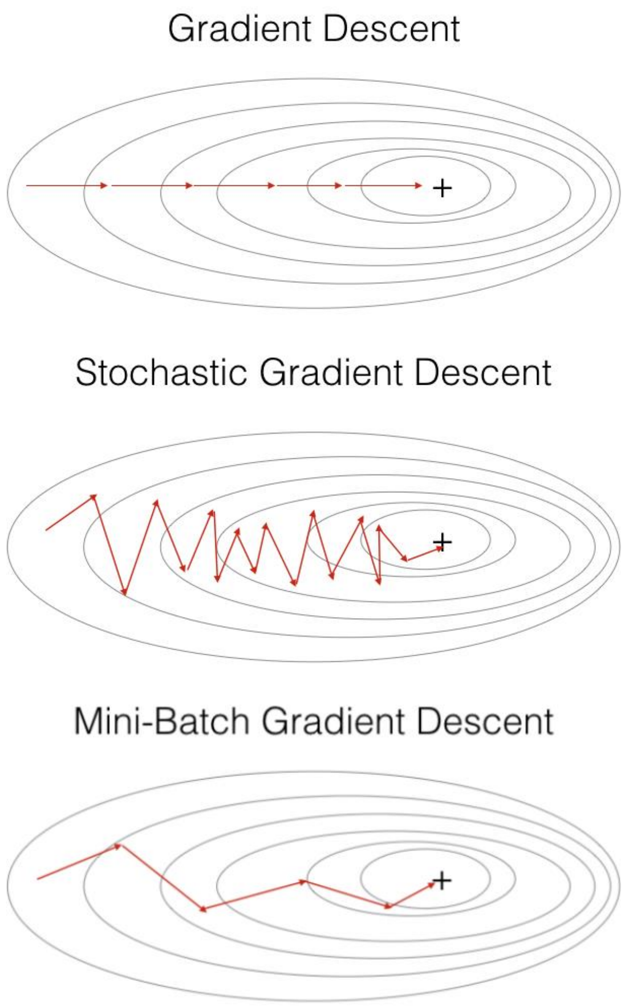


https://www.youtube.com/watch?v=UmathvAKj80

Stochastic Gradient Descent is sloppy but fast

- Less accurate but more time and memory efficient.
    - Can do more with the same resources.
- Updates are made based on a randomly selected window of word.
- Minibatches can further help smooth the optimisation path of SGD and take advantage of parallelisation on GPUs.
- The SGD has the problem of sparsity though.
    - **Solution?** Update only those word vectors that appear in the minibatch.

$$\nabla_\theta J(\theta) = \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ \nabla v_i \\ 0 \\ \cdot \\ \cdot \\ \nabla u_i \\ 0 \\ \cdot \\ \cdot \end{bmatrix} \in \mathbb{R}^{2dV}$$



https://www.youtube.com/watch?v=UmathvAKj80

There is something wrong with the existing word2vec model

- Naïve $softmax$ is simpler but expensive.

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

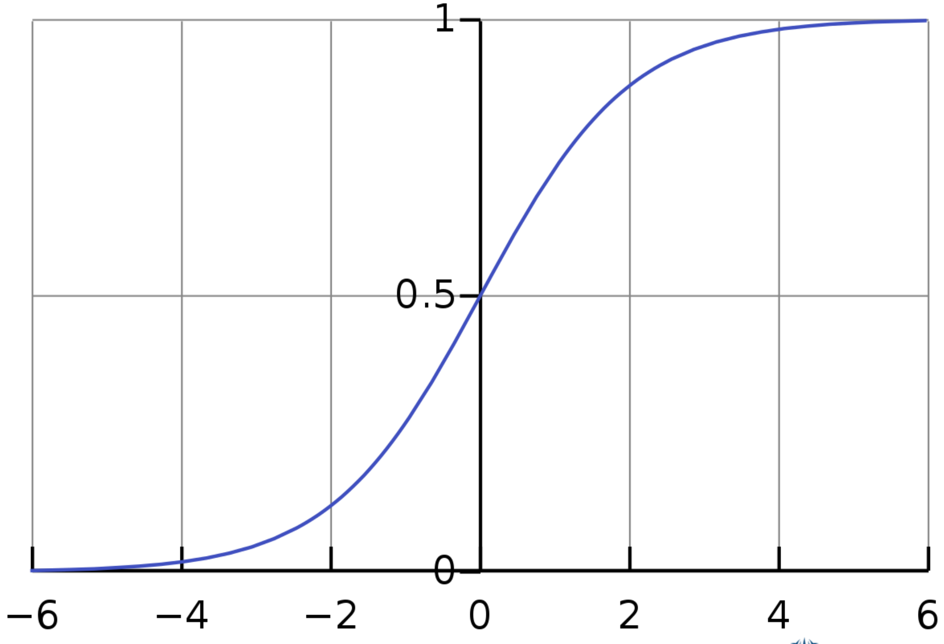
There is something wrong with the existing word2vec model

- Naïve $softmax$ is simpler but expensive.

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

- We may get similar performance using sigmoid function with significantly reduced computations.

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



Negative Sampling can improve training efficiency of Skip-Gram

- What's negative sampling?

  - Train binary logistic regression for each word in the numerator for a true pair vs several noise pairs (negative samples).

  - The goal is to maximise probability for true pair (an actual neighbour) and minimise probabilities for negative labels. (How to pick negative labels?)

https://arxiv.org/pdf/1310.4546.pdf

Skip-Gram model with negative sampling

- The objective function changes to

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T} J_t(\theta)$$

$$J_t(\theta) = \underbrace{\log \sigma(u_o^T v_c)}_{\text{Positive Sample}} + \underbrace{\sum_{j=1}^{k} \mathbb{E}_{j\sim P(w)}\left[\log \sigma(-u_j^T v_c)\right]}_{\text{Negative Samples}}$$

https://arxiv.org/pdf/1310.4546.pdf

Skip-Gram model with negative sampling

- The objective function changes to

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T} J_t(\theta)$$

$$J_t(\theta) = \underbrace{\log \sigma(u_o^T v_c)}_{\text{Positive Sample}} + \underbrace{\sum_{j=1}^{k} \mathbb{E}_{j\sim P(w)}\left[\log \sigma(-u_j^T v_c)\right]}_{\text{Negative Samples}}$$

- Sampling of negative pairs is not completely random.

$$P(w) = \frac{U(w)^{3/4}}{Z}$$

  - Here $P(w)$ is the distribution of noise and $U(w)$ is unigram distribution (just a histogram if each word in a corpus). Raising unigram to ¾ helps compensate sampling of rarer and frequent words.

https://arxiv.org/pdf/1310.4546.pdf

The content discussed so far is taken from original word2vec papers

Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

Distributed Representations of Words and Phrases and their Compositionality

Tomas Mikolov
Google Inc.
Mountain View
mikolov@google.com

Ilya Sutskever
Google Inc.
Mountain View
ilyasu@google.com

Kai Chen
Google Inc.
Mountain View
kai@google.com

Greg Corrado
Google Inc.
Mountain View
gcorrado@google.com

Jeffrey Dean
Google Inc.
Mountain View
jeff@google.com

Co-occurrence is another criterion to estimate word similarity

- With huge amount of data (which we usually have), simply counting co-occurrences can also provide statistically significant information.

Co-occurrence is another criterion to estimate word similarity

- With huge amount of data (which we usually have), simply counting co-occurrences can also provide statistically significant information.

- It can be done in two ways.

  - **Using a window:** Similar to word2vec, captures **syntactic** and **semantic** information using a window around each word.

Co-occurrence is another criterion to estimate word similarity

- With huge amount of data (which we usually have), simply counting co-occurrences can also provide statistically significant information.

- It can be done in two ways.

  - **Using a window:** Similar to word2vec, captures **syntactic** and **semantic** information using a window around each word.

  - **Over the whole document:** Provides **topics** leading to Latent Semantic Analysis.

    - LSA deals with representation of text data in terms of latent features and reducing the dimensionality of original data.

Let’s make a window based co-occurrence matrix

- Let the window size is 1 (unrealistic but simple).

Let’s make a window based co-occurrence matrix

- Let the window size is 1 (unrealistic but simple).

- Window is symmetrical in context direction.

Let’s make a window based co-occurrence matrix

- Let the window size is 1 (unrealistic but simple).
- Window is symmetrical in context direction.
- Example corpus:
    - I like deep learning. I like NLP. I enjoy driving.

Let’s make a window based co-occurrence matrix

- Let the window size is 1 (unrealistic but simple).
- Window is symmetrical in context direction.
- Example corpus: I like artificial intelligence. I like pizza. I enjoy driving.

	I	Like	Artificial	Intelligence	Pizza	Enjoy	Driving	.
I	0	2	0	0	0	1	0	2
Like	2	0	1	0	1	0	0	0
Artificial	0	1	0	1	0	0	0	0
Intelligence	0	0	1	0	0	0	0	1
Pizza	0	1	0	0	0	0	0	1
Enjoy	1	0	0	0	0	0	1	0
Driving	0	0	0	0	0	1	0	1
.	2	0	0	1	1	0	1	0

Co-occurrence matrix suffers from a few problems

- **Curse of dimensionality:** Matrix size increases drastically with vocabulary.
- **Sparsity:** Model are less robust because of sparsity.
  - How to fix it?

Co-occurrence matrix suffers from a few problems

- **Curse of dimensionality:** Matrix size increases drastically with vocabulary.
- **Sparsity:** Model are less robust because of sparsity.
  - How to fix it?
    - Store only the most important information in a small yet dense vector (25-1000).
  - How?

Co-occurrence matrix suffers from a few problems

- **Curse of dimensionality:** Matrix size increases drastically with vocabulary.
- **Sparsity:** Model are less robust because of sparsity.
    - How to fix it?
        - Store only the most important information in a small yet dense vector (25-1000).
    - How? **Singular Value Decomposition**.
        - Data-driven dimensionality reduction technique tailored for specific problem.
        - SDV is the basis for PCA.
        - Google uses it for page ranking.
        - Facebook uses it for facial recognition.

https://www.youtube.com/watch?v=gXbThCXjZFM
https://www.youtube.com/watch?v=vSczTbgc8Rc

How well co-occurrence matrix works?

- The technique was used around 2000s for LSA and Information Retrieval.

How well co-occurrence matrix works?

- The technique was used around 2000s for LSA and Information Retrieval.

- Co-occurrence matrix was explored to discover meaningful semantic directions in the low-dimensional projections of embeddings.

How well co-occurrence matrix works?

- The technique was used around 2000s for LSA and Information Retrieval.

- Co-occurrence matrix was explored to discover meaningful semantic directions in the low-dimensional projections of embeddings.

- The performance was acceptable but not extraordinary.

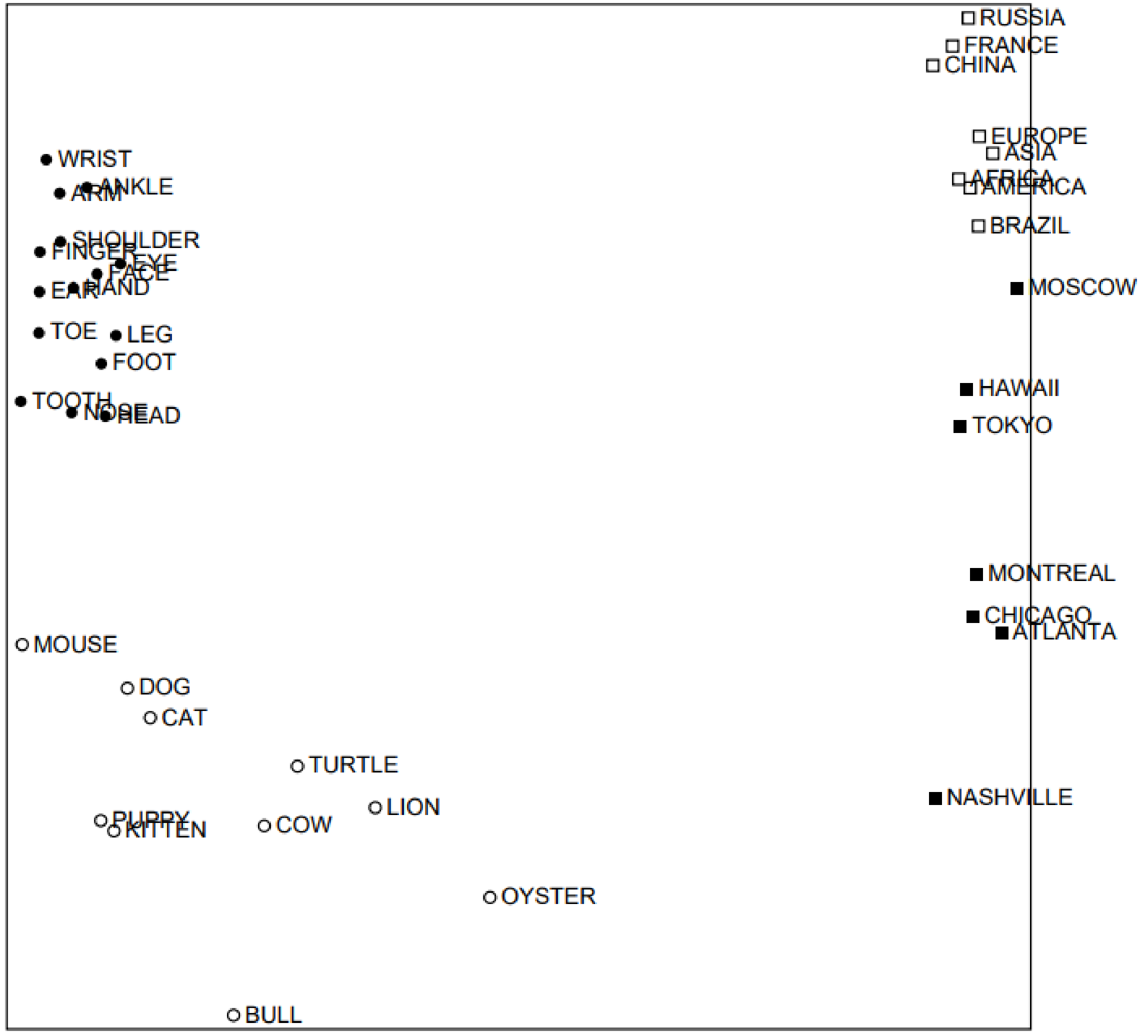
Many hacks are used to improved the performance of co-occurrence matrix

- To fix the problem of frequently occurring words,
  - Scaling (log scaling) the counts in the cells of co-occurrence matrix can greatly help.
  - Applying ceiling function, $\min(X, t)$, where $t \approx 100$.
  - Using ramped window to count the closer words more.

Many hacks are used to improved the performance of co-occurrence matrix

- To fix the problem of frequently occurring words,
    - Scaling (log scaling) the counts in the cells of co-occurrence matrix can greatly help.
    - Applying ceiling function, $\min(X, t)$, where $t \approx 100$.
    - Using ramped window to count the closer words more.
- Instead of counts, use Pearson correlation.
    - Tells about correlation plus strength of correlation.

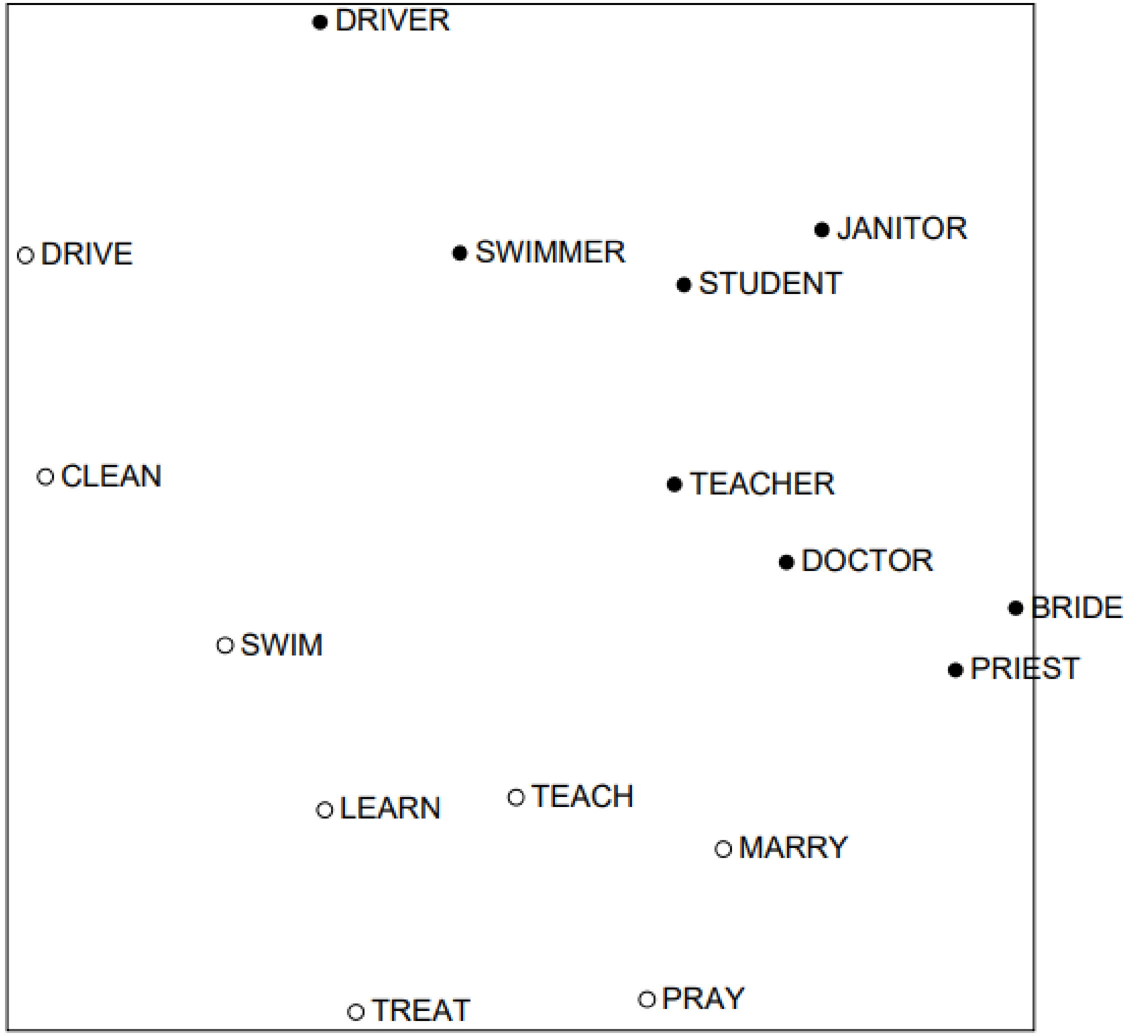
Interesting patterns emerge from word vectors using these transformations



Rohde, D. L., Gonnerman, L. M., & Plaut, D. C. (2006). An improved model of semantic similarity based on lexical co-occurrence. Communications of the ACM, 8(627-633), 116.

Interesting patterns emerge from word vectors using these transformations

- Semantic vectors are encoded as linear components of word representation.



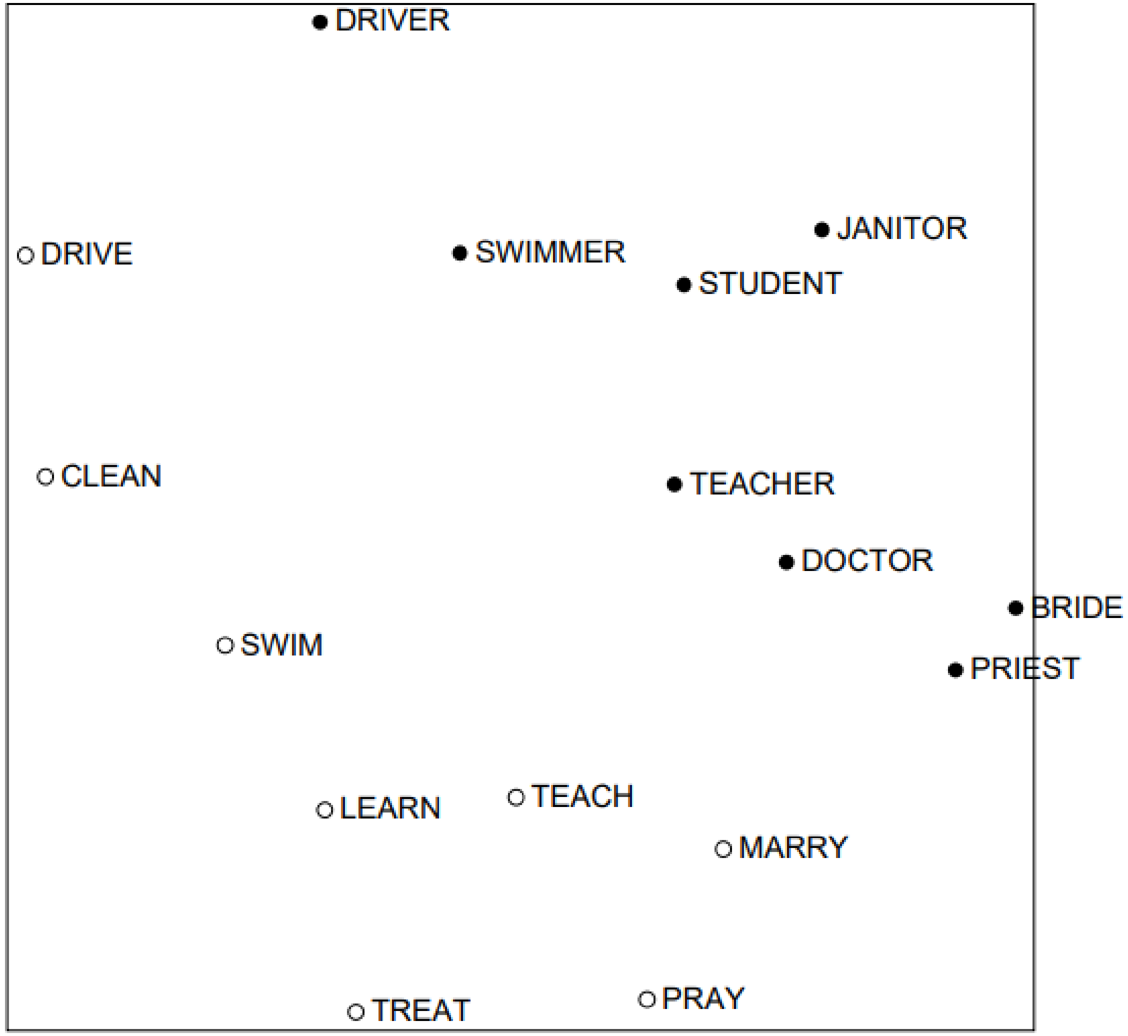
Rohde, D. L., Gonnerman, L. M., & Plaut, D. C. (2006). An improved model of semantic similarity based on lexical co-occurrence. Communications of the ACM, 8(627-633), 116.

Interesting patterns emerge from word vectors using these transformations

- Semantic vectors are encoded as linear components of word representation.

- A vector space that has this linearity property can do well with word analogy.

    - Man is to king as women is to … ?



Rohde, D. L., Gonnerman, L. M., & Plaut, D. C. (2006). An improved model of semantic similarity based on lexical co-occurrence. Communications of the ACM, 8(627-633), 116.

Interesting patterns emerge from word vectors using these transformations

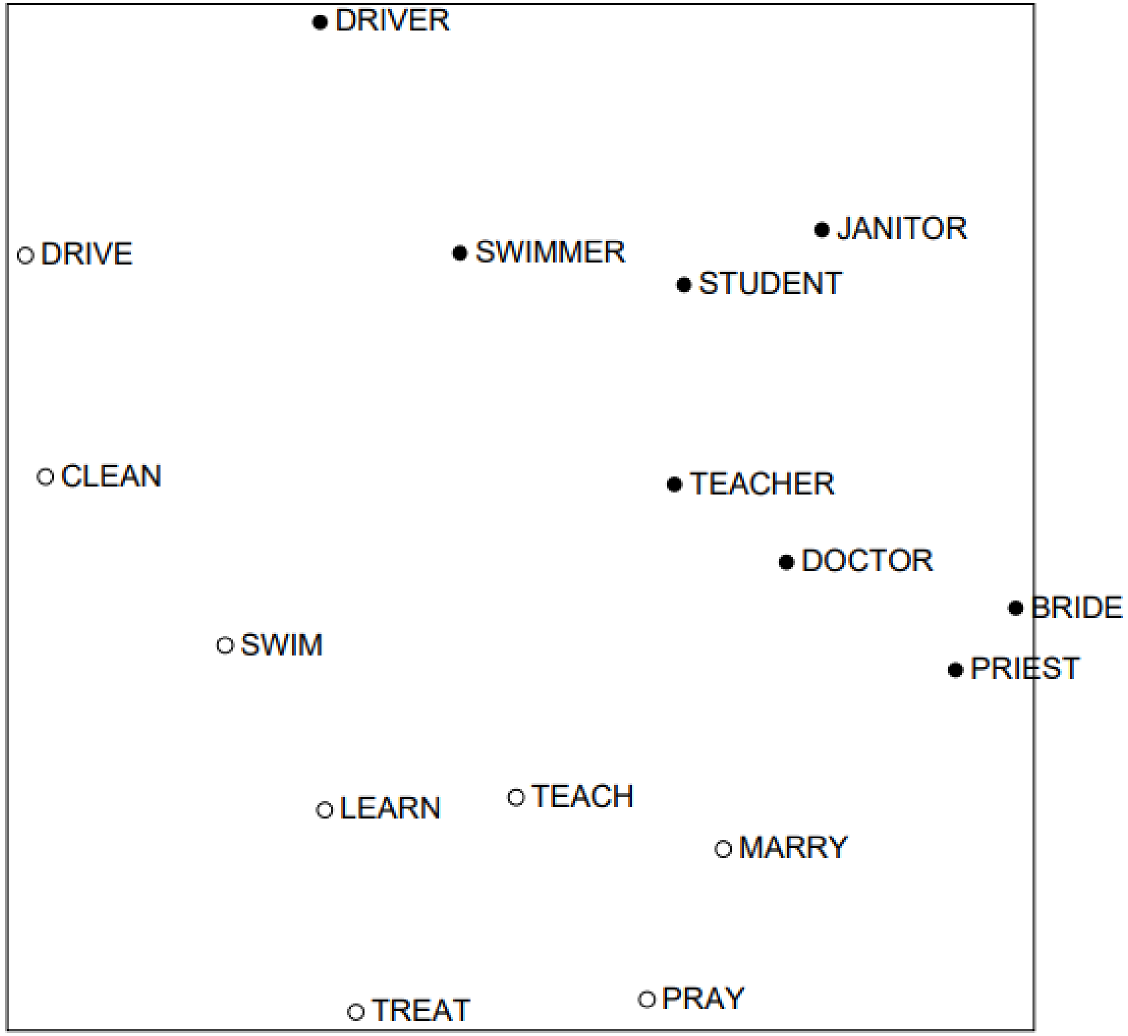
- Semantic vectors are encoded as linear components of word representation.

- A vector space that has this linearity property can do well with word analogy.

  - Man is to king as women is to … ?

- Lesson?

  - Even with simply counting word occurrences (with some hacks, of course), we can still make reasonably good word representations.



Rohde, D. L., Gonnerman, L. M., & Plaut, D. C. (2006). An improved model of semantic similarity based on lexical co-occurrence. Communications of the ACM, 8(627-633), 116.

Count-based and neural-based models have their merits and demerits

Count-Based

- Fast training.
- Efficient usage of statistics.
- Mostly good for capturing word similarities.
- Disproportionate importance given to large counts.
- Requires large memory to construct matrix.

Count-based and neural-based models have their merits and demerits

Count-Based

- Fast training.
- Efficient usage of statistics.
- Mostly good for capturing word similarities.
- Disproportionate importance given to large counts.
- Requires large memory to construct matrix.

Neural-Based

- Scales well with corpus size.
- Can capture complex patterns other than word similarities.
- Performs well on other tasks also.
- Due to sampling words, memory is not an issue.
- Inefficient usage of statistics.

Combining counts and neural methods can produce better embeddings

Combining counts and neural methods can produce better embeddings

- **Objective:** Obtain components of meaning as linear operations in vector space without (many) hacks.

- **Contribution:** Ratios of co-occurrence probabilities can encode meaning components.

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)

Combining counts and neural methods can produce better embeddings

- **Objective:** Obtain components of meaning as linear operations in vector space without (many) hacks.

- **Contribution:** Ratios of co-occurrence probabilities can encode meaning components.

	$x =$ Solid	$x =$ Gas	$x =$ Water	$x =$ Car
$P(x \mid \text{ice})$	$Large$	$Small$	$Large$	$Small$

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)

Combining counts and neural methods can produce better embeddings

- **Objective:** Obtain components of meaning as linear operations in vector space without (many) hacks.

- **Contribution:** Ratios of co-occurrence probabilities can encode meaning components.

	$x =$ Solid	$x =$ Gas	$x =$ Water	$x =$ Car
$\boldsymbol{P(x \mid \mathrm{ice})}$	$Large$	$Small$	$Large$	$Small$
$\boldsymbol{P(x \mid steam)}$	$Small$	$Large$	$Large$	$Small$

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)

Combining counts and neural methods can produce better embeddings

- **Objective:** Obtain components of meaning as linear operations in vector space without (many) hacks.

- **Contribution:** Ratios of co-occurrence probabilities can encode meaning components.

	$x =$ Solid	$x =$ Gas	$x =$ Water	$x =$ Car
$P(x \mid \text{ice})$	$Large$	$Small$	$Large$	$Small$
$P(x \mid steam)$	$Small$	$Large$	$Large$	$Small$
$\frac{P(x \mid \text{ice})}{P(x \mid steam)}$	$Large$	$Small$	~ 1	~ 1

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)

Combining counts and neural methods can produce better embeddings

- **Objective:** Obtain components of meaning as linear operations in vector space without (many) hacks.
- **Contribution:** Ratios of co-occurrence probabilities can encode meaning components.

	x = Solid	x = Gas	x = Water	x = Car
$P(x \mid \text{ice})$	1.9^{-4}	6.6^{-5}	3.0^{-3}	1.7^{-5}
$P(x \mid steam)$	2.2^{-5}	7.8^{-4}	2.2^{-3}	1.8^{-5}
$\frac{P(x \mid \text{ice})}{P(x \mid steam)}$	8.9	8.5^{-2}	1.36	0.96

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)

Combining counts and neural methods can produce better embeddings

- **Objective:** Obtain components of meaning as linear operations in vector space without (many) hacks.

- **Contribution:** Ratios of co-occurrence probabilities can encode meaning components.

	x = Solid	x = Gas	x = Water	x = Car
$P(x \mid \text{ice})$	1.9^{-4}	6.6^{-5}	3.0^{-3}	1.7^{-5}
$P(x \mid \text{steam})$	2.2^{-5}	7.8^{-4}	2.2^{-3}	1.8^{-5}
$\frac{P(x \mid \text{ice})}{P(x \mid \text{steam})}$	8.9	8.5^{-2}	1.36	0.96

Dimension of meaning

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)

Combining counts and neural methods can produce better embeddings

- **Objective:** Obtain components of meaning as linear operations in vector space without (many) hacks.

- **Contribution:** Ratios of co-occurrence probabilities can encode meaning components.

	x = Solid	x = Gas	x = Water	x = Car
$P(x \mid ice)$	1.9^{-4}	6.6^{-5}	3.0^{-3}	1.7^{-5}
$P(x \mid steam)$	2.2^{-5}	7.8^{-4}	2.2^{-3}	1.8^{-5}
$\frac{P(x \mid ice)}{P(x \mid steam)}$	8.9	8.5^{-2}	1.36	0.96

Dimension of meaning

Ratio of co-occurrence probabilities should be linear in this vector space

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)

NUST
Defining futures
School of Electrical Engineering
& Computer Science

GloVe embeddings encode meaning in vector differences

- How to ensure ratio of co-occurrence probabilities are linear in vector space?
  - **Using log bilinear model:** Make the dot product equal to the log of co-occurrence probabilities.

$$w_i . w_j = \log P(i|j)$$

GloVe embeddings encode meaning in vector differences

- How to ensure ratio of co-occurrence probabilities are linear in vector space?
  - **Using log bilinear model:** Make the dot product equal to the log of co-occurrence probabilities.

$$w_i . w_j = \log P(i|j)$$

- With vector difference, the dot product becomes the log of ratio of co-occurrence probabilities

$$w_x . (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$$

GloVe embeddings encode meaning in vector differences

- How to ensure ratio of co-occurrence probabilities are linear in vector space?
  - **Using log bilinear model:** Make the dot product equal to the log of co-occurrence probabilities.

$$w_i . w_j = \log P(i|j)$$

- With vector difference, the dot product becomes the log of ratio of co-occurrence probabilities

$$w_x . (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$$

- The objective function, is then

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

NUST
Defining futures
School of Electrical Engineering
& Computer Science

GloVe embeddings encode meaning in vector differences

- How to ensure ratio of co-occurrence probabilities are linear in vector space?
  - **Using log bilinear model:** Make the dot product equal to the log of co-occurrence probabilities.

$$w_i . w_j = \log P(i|j)$$

- With vector difference, the dot product becomes the log of ratio of co-occurrence probabilities

$$w_x . (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$$

- The objective function, is then

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

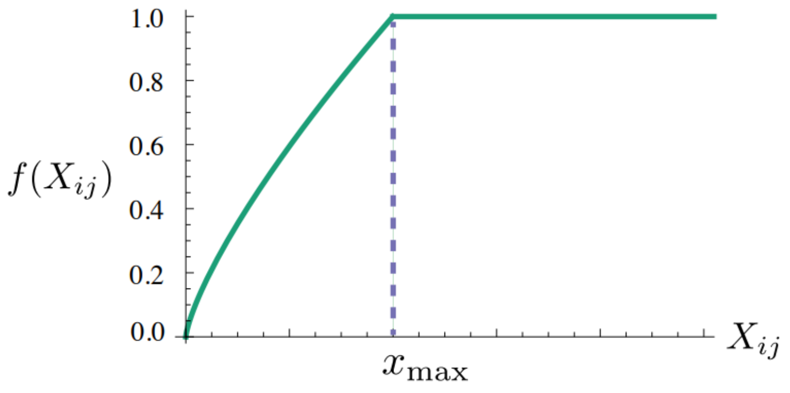


Figure 1: Weighting function f with $\alpha = 3/4$.

NUST
Defining futures
School of Electrical Engineering
& Computer Science

GloVe embeddings encode meaning in vector differences

- How to ensure ratio of co-occurrence probabilities are linear in vector space?

  - **Using log bilinear model:** Make the dot product equal to the log of co-occurrence probabilities.

$$w_i . w_j = \log P(i|j)$$

- With vector difference, the dot product becomes the log of ratio of co-occurrence probabilities

$$w_x . (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$$

- The objective function, is then

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

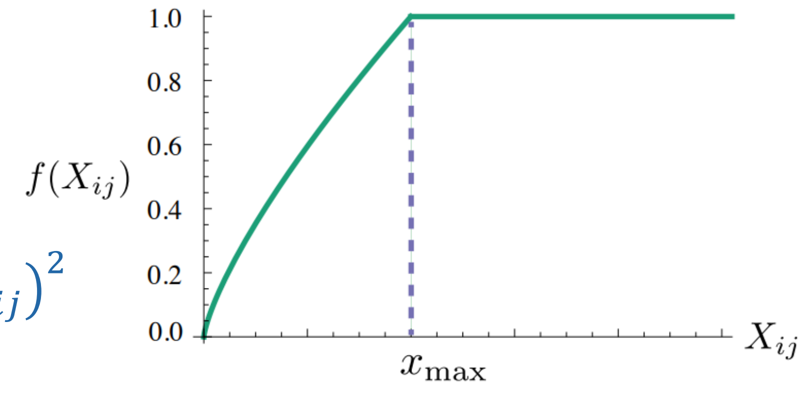


Figure 1: Weighting function f with $\alpha = 3/4$.

- GloVe trains faster, is scalable, and has better performance with small corpora and small vectors compared to word2vec

GloVe Results

- Nearest words to frog:
  - Frogs
  - Toad
  - Litoria
  - Leptodactylidae
  - Rana
  - Lizard
  - Eleutherodactylus

GloVe: Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, Christopher D. Manning
Computer Science Department, Stanford University, Stanford, CA 94305
jpennin@stanford.edu, richard@socher.org, manning@stanford.edu

Litoria

Leptodactylidae

Rana

Eleutherodactylus

Do you have any problem?

Some material (images, tables, text etc.) in this presentation has been borrowed from different books, lecture notes, and the web. The original contents solely belong to their owners, and are used in this presentation only for clarifying various educational concepts. Any copyright infringement is ***not at all*** intended.