Pattern Recognition

Part 6: Object-to-vector Conversion

Gerhard Schmidt

Christian-Albrechts-Universität zu Kiel
Faculty of Engineering
Institute of Electrical and Information Engineering
Digital Signal Processing and System Theory

Contents

Motivation

Transformer-based Neural Networks

- Basis for a variety of powerful AI approaches
  - Language translation (deepl, …)
  - Text generation (chatGPT, …)
- Based on a paper by Ashish Vaswani et al.: "Attention is All You Need", appeared in 2017.

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

*Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head attention and the parameter-free position representation and became the other person involved in nearly every detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating our research.

†Work performed while at Google Brain.

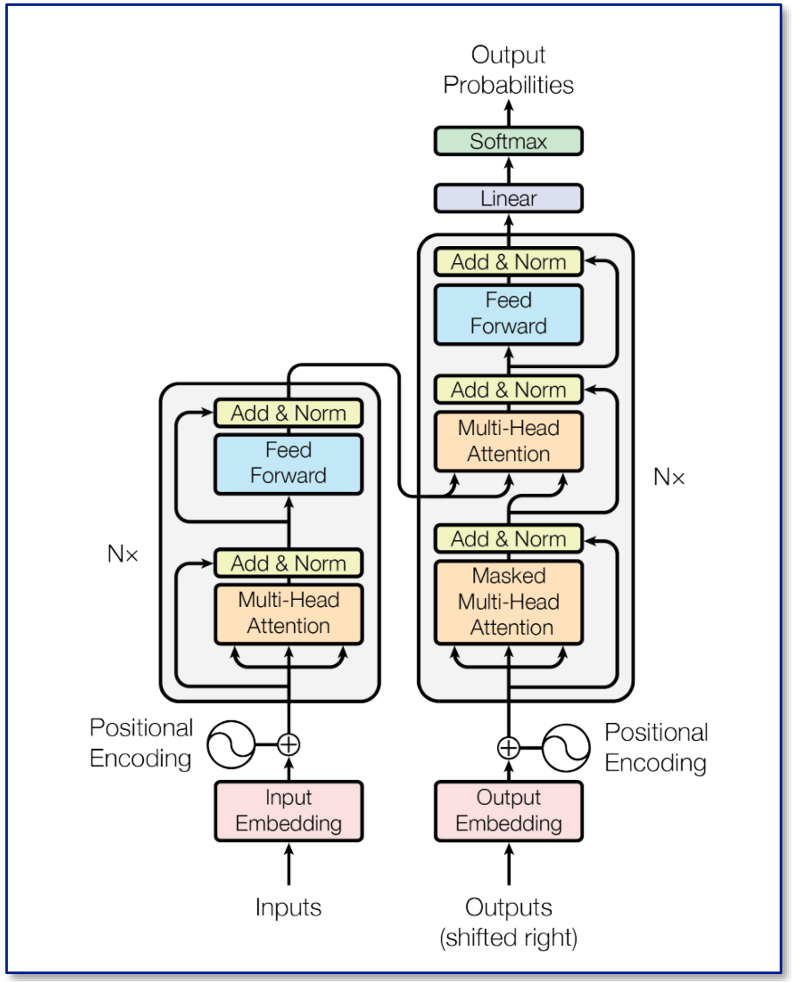
‡Work performed while at Google Research.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Motivation

Transformer-based Neural Networks

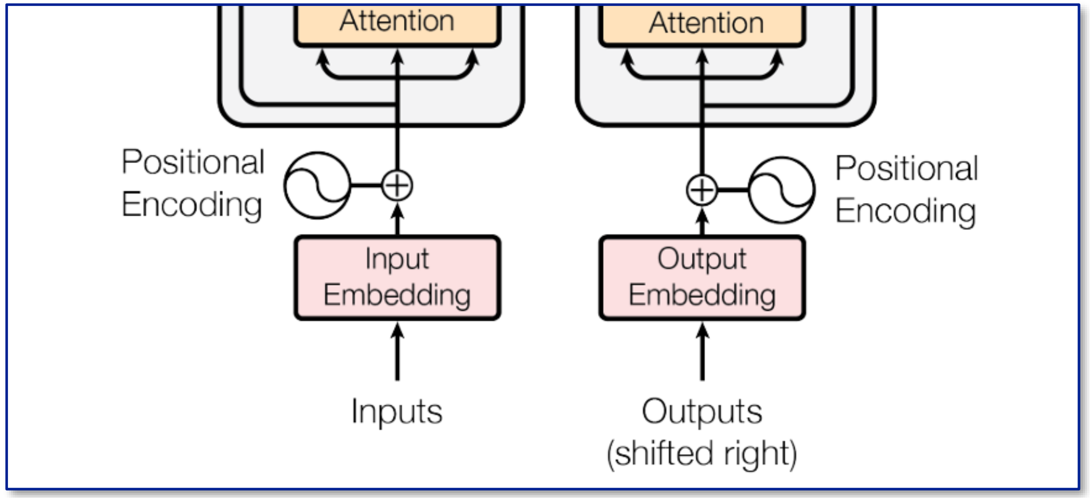
- Basis for a variety of powerful AI approaches
  - Language translation (deepl, …)
  - Text generation (chatGPT, …)
- Based on a paper by Ashish Vaswani et al.: “Attention is All You Need”, appeared in 2017.
- Here a so-called transformer approach was used and extended by so-called attention networks.
- Those networks will be discussed later.



Motivation

Transformer-based Neural Networks

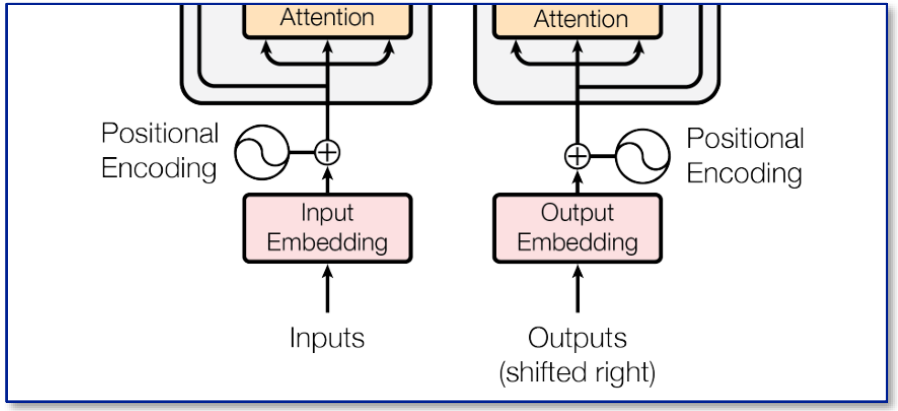
- Basis for a variety of powerful AI approaches
  - Language translation (deepl, …)
  - Text generation (chatGPT, …)
- Based on a paper by Ashish Vaswani et al.: “Attention is All You Need”, appeared in 2017.
- Here a so-called transformer approach was used and extended by so-called attention networks.
- Those networks will be discussed later.
- But prior to applying the attention networks a conversion from words to high-dimensional vectors is required.
- The generation of these vector models will be the topic of this lecture part.



Motivation

Transformer-based Neural Networks

- Basis for a variety of powerful AI approaches
    - Language translation (deepl, …)
    - Text generation (chatGPT, …)
- Based on a paper by Ashish Vaswani et al.: "Attention is All You Need", appeared in 2017.
- Here a so-called transformer approach was used and extended by so-called attention networks.
- Those networks will be discussed later.
- But prior to applying the attention networks a conversion from words to high-dimensional vectors is required.
- The generation of these vector models will be the topic of this lecture part.



SoS – What – will – be – the – next –

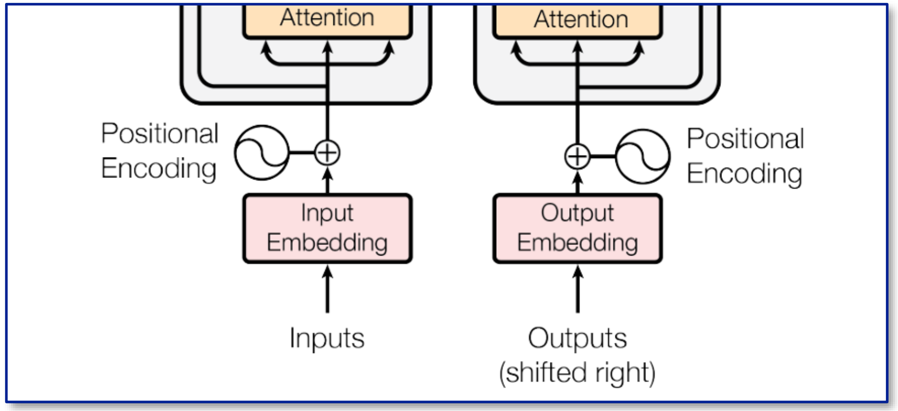
SoS – Was – wird – das –

Transforming this into a computer-processable (and "processable friendly") manner will be the task of this lecture part.

Motivation

Transformer-based Neural Networks

- Basis for a variety of powerful AI approaches
  - Language translation (deepl, …)
  - Text generation (chatGPT, …)
- Based on a paper by Ashish Vaswani et al.: “Attention is All You Need”, appeared in 2017.
- Here a so-called transformer approach was used and extended by so-called attention networks.
- Those networks will be discussed later.
- But prior to applying the attention networks a conversion from words to high-dimensional vectors is required.
- The generation of these vector models will be the topic of this lecture part.



SoS – What – will – be – the – next –

$$\{w_{\text{eng},n-5},\,\ldots,\,w_{\text{eng},n-1},\,w_{\text{eng},n}\}$$

SoS – Was – wird – das –

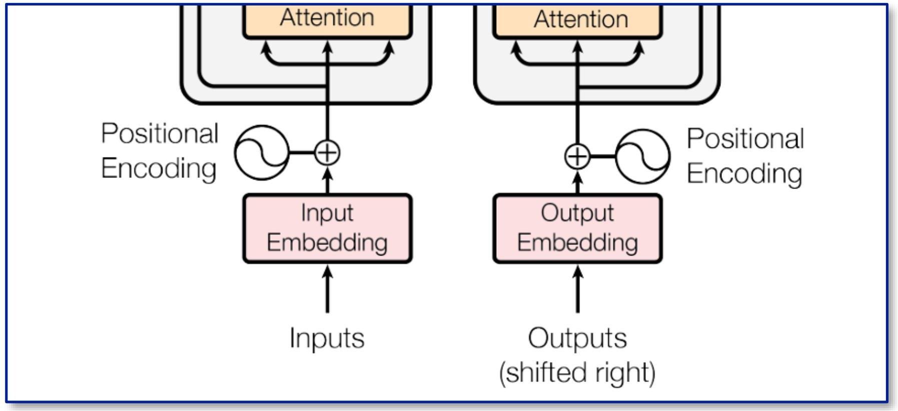
$$\{w_{\text{ger},n-5},\,\ldots,\,w_{\text{ger},n-1}\}$$

Transforming this into a computer-processable (and “processable friendly”) manner will be the task of this lecture part.

Motivation

Transformer-based Neural Networks

- Basis for a variety of powerful AI approaches
  - Language translation (deepl, …)
  - Text generation (chatGPT, …)
- Based on a paper by Ashish Vaswani et al.: "Attention is All You Need", appeared in 2017.
- Here a so-called transformer approach was used and extended by so-called attention networks.
- Those networks will be discussed later.
- But prior to applying the attention networks a conversion from words to high-dimensional vectors is required.
- The generation of these vector models will be the topic of this lecture part.



SoS – What – will – be – the – next –

$$\{w_{\mathrm{eng},n-5},\ \ldots,\ w_{\mathrm{eng},n-1},\ w_{\mathrm{eng},n}\}$$

$$\{\boldsymbol{x}_{\mathrm{eng},18},\ \ldots,\ \boldsymbol{x}_{\mathrm{eng},144},\ \boldsymbol{x}_{\mathrm{eng},258}\}$$

SoS – Was – wird – das –

$$\{w_{\mathrm{ger},n-5},\ \ldots,\ w_{\mathrm{ger},n-1}\}$$

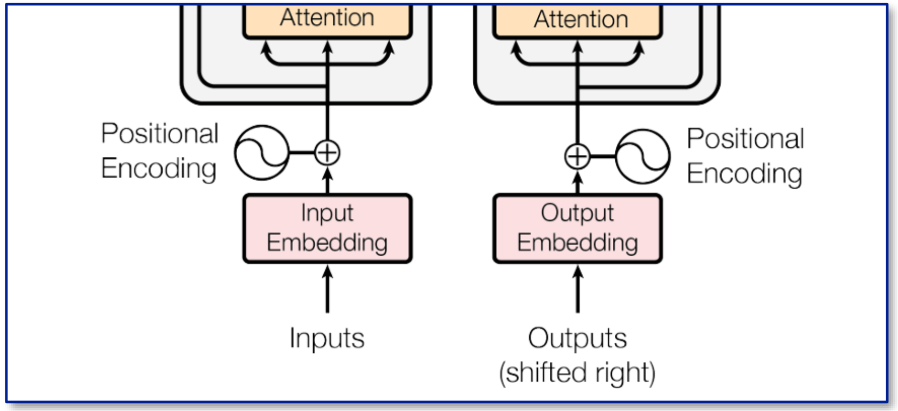
$$\{\boldsymbol{x}_{\mathrm{ger},255},\ \ldots,\ \boldsymbol{x}_{\mathrm{ger},158}\}$$

Transforming this into a computer-processable (and "processable friendly") manner will be the task of this lecture part.

Motivation

Transformer-based Neural Networks

- Basis for a variety of powerful AI approaches
  - Language translation (deepl, …)
  - Text generation (chatGPT, …)
- Based on a paper by Ashish Vaswani et al.: "Attention is All You Need", appeared in 2017.
- Here a so-called transformer approach was used and extended by so-called attention networks.
- Those networks will be discussed later.
- But prior to applying the attention networks a conversion from words to high-dimensional vectors is required
- The generation of these vector models will be the topic of this lecture part.



Start of sentence

Entries of word lists

High-dimensional (100 … 1000) word vectors

SoS – What – will – be – the – next –

$$\{w_{\text{eng},n-5},\ \ldots,\ w_{\text{eng},n-1},\ w_{\text{eng},n}\}$$

$$\{\boldsymbol{x}_{\text{eng},18},\ \ldots,\ \boldsymbol{x}_{\text{eng},144},\ \boldsymbol{x}_{\text{eng},258}\}$$

SoS – Was – wird – das –

$$\{w_{\text{ger},n-5},\ \ldots,\ w_{\text{ger},n-1}\}$$

$$\{\boldsymbol{x}_{\text{ger},255},\ \ldots,\ \boldsymbol{x}_{\text{ger},158}\}$$

Transforming this into a computer-processable (and "processable friendly") manner will be the task of this lecture part.

Focus on text

Objects versus words:

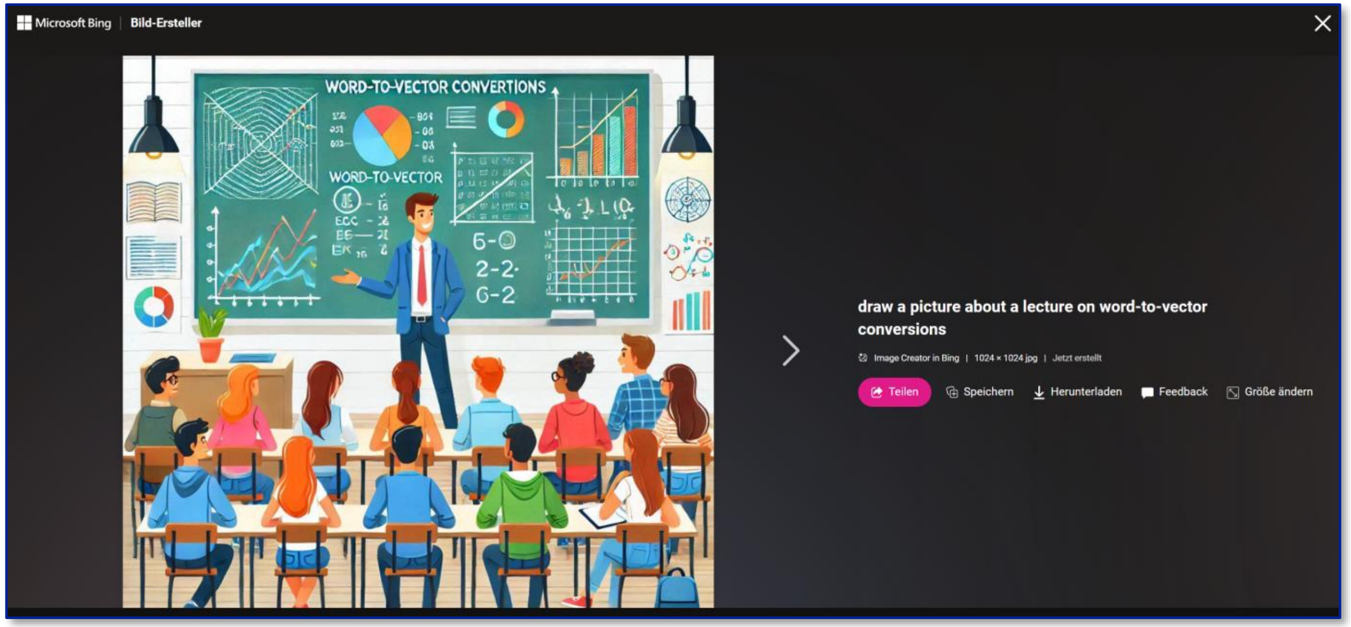
- In the following we will mainly focus on words and sentences.
- Words in our meaning here are conventional words such as “I”, “can”, “do”, “it”, but also the start and the end of a sentence (coded as “SOS” and “EOS”) and all punctuation marks.

Christian-Albrechts-Universität zu Kiel

Focus on text

Objects versus words:

- In the following we will mainly focus on words and sentences.
- Words in our meaning here are conventional words such as “I”, “can”, “do”, “it”, but also the start and the end of a sentence (coded as “SOS” and “EOS”) and all punctuation marks.
- Beside text also pictures or sequences of pictures can be treated in a similar way – however, we will focus here only on words.



Basics

Vector distances and similarity measures:

- The most obvious distance between to vectors is their Euclidean distance:

$$\left\|\boldsymbol{x}_1 - \boldsymbol{x}_2\right\|^2 = \sum_{i=0}^{D-1} \left(x_{1,i} - x_{2,i}\right)^2.$$

Small, if vectors are close to each other.

Basics

Vector distances and similarity measures:

- The most obvious distance between to vectors is their Euclidean distance:

$$\left\|\boldsymbol{x}_1 - \boldsymbol{x}_2\right\|^2 = \sum_{i=0}^{D-1} \left(x_{1,i} - x_{2,i}\right)^2.$$

Small, if vectors are close to each other.

- Another one – something that we will use in the following – is the vector (dot) product:

$$\boldsymbol{x}_1^{\mathrm{T}}\, \boldsymbol{x}_2 = \sum_{i=0}^{D-1} x_{1,i}\, x_{2,i}.$$

Large, if vectors are close to each other.

Basics

Vector distances and similarity measures:

- The most obvious distance between to vectors is their Euclidean distance:

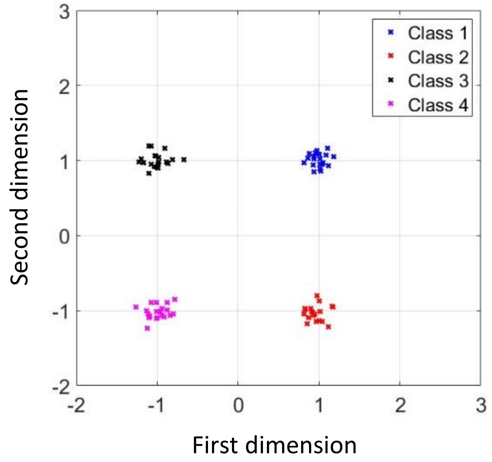
$$\Big\|\boldsymbol{x}_1 - \boldsymbol{x}_2\Big\|^2 = \sum_{i=0}^{D-1} \big(x_{1,i} - x_{2,i}\big)^2.$$

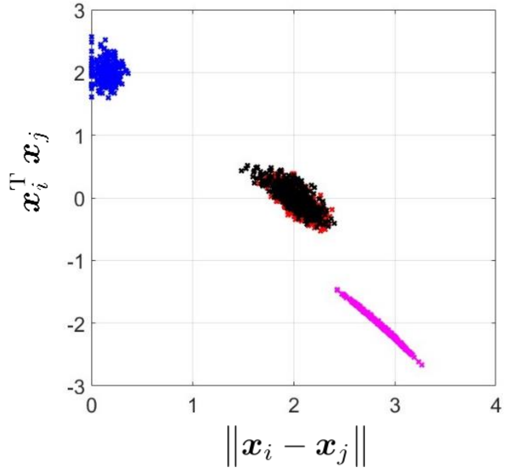
Small, if vectors are close to each other.

- Another one – something that we will use in the following – is the vector (dot) product:

$$\boldsymbol{x}_1^{\mathrm{T}}\,\boldsymbol{x}_2 = \sum_{i=0}^{D-1} x_{1,i}\,x_{2,i}.$$

Large, if vectors are close to each other.





Context-word probabilities

Conditional word probabilities:

(example part of a sentence)

… like to be in the gym for …

Context-word probabilities

Conditional word probabilities:

… like to be in the gym for …

Context words
at position n - 1, n - 2, …

Center word
at position n

Context words
at position $n + 1$, $n + 2$, …

Context-word probabilities

Conditional word probabilities:

Probability for the previous word …

$$p\big(w_{n-1}\big|w_n\big)$$

… like to be in the gym for …

Center word
at position n

Context words
at position n - 1, n - 2, …

Context words
at position $n + 1,\ n + 2,\ …$

Context-word probabilities

Conditional word probabilities:

… two words before … $p\big(w_{n-2}\big|w_n\big)$

… like to be in the gym for …

Center word
at position n

Context words
at position n - 1, n - 2, …

Context words
at position $n + 1$, $n + 2$, …

Christian-Albrechts-Universität zu Kiel

Context-word probabilities

Conditional word probabilities:

… three words before … $p\left(w_{n-3} \middle| w_n\right)$

… like to be in the gym for …

$\underbrace{\text{like to be}}\ \underbrace{\text{in}}\ \underbrace{\text{the gym for}}$

Context words
at position n - 1, n - 2, …

Center word
at position n

Context words
at position $n + 1$, $n + 2$, …

Context-word probabilities

Conditional word probabilities:

… also possible for future words

$p\left(w_{n+3} \middle| w_n\right)$

… like to be in the gym for …

Center word
at position n

Context words
at position n - 1, n - 2, …

Context words
at position $n + 1$, $n + 2$, …

Possibilities to (mathematically) describe words

Word list / object list:

and

amateur

animal

…

zebra

zumba

All words (without duplicates) – in total N words.

Possibilities to (mathematically) describe words

Word list / object list:

Option 1 would be to give each word an integer number.

and	$w(0) = 0$
amateur	$w(1) = 1$
animal	$w(2) = 2$
…	
zebra	$w(N-2) = N-2$
zumba	$w(N-1) = N-1$

This allows to efficiently "process" words by means of a computer / a neural network. However, this is not the best option ….

All words (without duplicates) – in total N words.

Possibilities to (mathematically) describe words

Word list / object list:

Option 1

Option 2 would so-called one-hot encoded vectors

	Option 1	Option 2
and	$w(0) = 0$	$w(0) \rightarrow [1, 0, 0, ..., 0, 0]^{\mathrm{T}}$
amateur	$w(1) = 1$	$w(1) \rightarrow [0, 1, 0, ..., 0, 0]^{\mathrm{T}}$
animal	$w(2) = 2$	$w(2) \rightarrow [0, 0, 1, ..., 0, 0]^{\mathrm{T}}$
…		
zebra	$w(N-2) = N-2$	$w(N-2) \rightarrow [0, 0, 0, ..., 1, 0]^{\mathrm{T}}$
zumba	$w(N-1) = N-1$	$w(N-1) \rightarrow [0, 0, 0, ..., 0, 1]^{\mathrm{T}}$

This allows to perform vector operations (such as additions and subtractions), but this is not of great value at this stage.

However, the individual entries can be interpreted as probabilities – but this is not used here.

All words (without duplicates) – in total N words.

Possibilities to (mathematically) describe words

Word list / object list:

	Option 1	*Option 2*	*Option 3 would be vectors of dimension M (with M << N)*
and	$w(0) = 0$	$w(0) \rightarrow [1, 0, 0, ..., 0, 0]^{\mathrm{T}}$	$w(0) \rightarrow \boldsymbol{x}_0$
amateur	$w(1) = 1$	$w(1) \rightarrow [0, 1, 0, ..., 0, 0]^{\mathrm{T}}$	$w(1) \rightarrow \boldsymbol{x}_1$
animal	$w(2) = 2$	$w(2) \rightarrow [0, 0, 1, ..., 0, 0]^{\mathrm{T}}$	$w(2) \rightarrow \boldsymbol{x}_2$
…			
zebra	$w(N-2) = N-2$	$w(N-2) \rightarrow [0, 0, 0, ..., 1, 0]^{\mathrm{T}}$	$w(N-2) \rightarrow \boldsymbol{x}_{N-2}$
zumba	$w(N-1) = N-1$	$w(N-1) \rightarrow [0, 0, 0, ..., 0, 1]^{\mathrm{T}}$	$w(N-1) \rightarrow \boldsymbol{x}_{N-1}$

All words (without duplicates) – in total N words.

Cost functions

Text from Wikipedia about "Machine Learning":

Machine learning is a field of study in artificial intelligence concerned with the development and study of statistical algorithms that can learn from data and generalize to unseen data, and thus perform tasks without explicit instructions. Quick progress in the field of deep learning, beginning in 2010s, allowed neural networks to surpass many previous approaches in performance.

Cost functions

Text from Wikipedia about "Machine Learning":

Machine learning is a field of study in artificial intelligence concerned with the development and study of statistical algorithms that can learn from data ***and*** generalize to unseen data, and thus perform tasks without explicit instructions. Quick progress in the field of deep learning, beginning in 2010s, allowed neural networks to surpass many previous approaches in performance.

Select current center word

Cost functions

Text from Wikipedia about "Machine Learning":

Machine learning is a field of study in artificial intelligence concerned with the development and study of statistical ***algorithms that can learn from data and generalize to unseen data, and*** thus perform tasks without explicit instructions. Quick progress in the field of deep learning, beginning in 2010s, allowed neural networks to surpass many previous approaches in performance.

Selecting a "word window" – consisting of previous and follow-up words

Cost functions

Text from Wikipedia about "Machine Learning":

Machine learning is a field of study in artificial intelligence concerned with the development and study of statistical ***algorithms that can learn from data and generalize to unseen data, and*** thus perform tasks without explicit instructions. Quick progress in the field of deep learning, beginning in 2010s, allowed neural networks to surpass many previous approaches in performance.

Compute the following probability:

$$\prod_{k=-K,\, k\neq 0}^{k=K} p\left(w_{n+k} \middle| w_n\right)$$

Cost functions

Text from Wikipedia about "Machine Learning":

Machine learning is a field of study in artificial intelligence concerned with the development and study of statistical algorithms ***that can learn from data and generalize to unseen data, and thus*** perform tasks without explicit instructions. Quick progress in the field of deep learning, beginning in 2010s, allowed neural networks to surpass many previous approaches in performance.

Shift the center word and the window by one word

Cost functions

Text from Wikipedia about "Machine Learning":

Machine learning is a field of study in artificial intelligence concerned with the development and study of statistical algorithms ***that can learn from data and generalize to unseen data, and thus*** perform tasks without explicit instructions. Quick progress in the field of deep learning, beginning in 2010s, allowed neural networks to surpass many previous approaches in performance.

Compute the probability of the overall paragraph / text:

$$\prod_{n=0}^{N-1} \prod_{k=-K,\, k\neq 0}^{k=K} p\big(w_{n+k} \big| w_n\big)$$

Please note, that N is indicating here all words of the text (in contrast to the slides before, where N has been the number of different words)!

Training – Basic Ideas

Gradient-based optimization

- The word probabilities will be dependent on the word vectors:

$$p\big(w_i \big| w_j\big) \;=\; p\big(\boldsymbol{x}_i \big| \boldsymbol{x}_j\big).$$

Training – Basic Ideas

Gradient-based optimization

- The word probabilities will be dependent on the word vectors:

$$p\big(w_i \big| w_j\big) \;=\; p\big(\boldsymbol{x}_i \big| \boldsymbol{x}_j\big).$$

- Thus, the global probability can be written like this:

$$\prod_{n=0}^{N-1} \prod_{k=-K,\, k\neq 0}^{k=K} p\big(w_{n+k} \big| w_n\big) \;=\; \prod_{n=0}^{N-1} \prod_{k=-K,\, k\neq 0}^{k=K} p\big(\boldsymbol{x}_{i(n,k)} \big| \boldsymbol{x}_{i(n)}\big).$$

Training – Basic Ideas

Gradient-based optimization

- The word probabilities will be dependent on the word vectors:

$$p\big(w_i \big| w_j\big) \;=\; p\big(\boldsymbol{x}_i \big| \boldsymbol{x}_j\big).$$

- Thus, the global probability can be written like this:

$$\prod_{n=0}^{N-1} \prod_{k=-K,\, k\neq 0}^{k=K} p\big(w_{n+k} \big| w_n\big) \;=\; \prod_{n=0}^{N-1} \prod_{k=-K,\, k\neq 0}^{k=K} p\big(\boldsymbol{x}_{i(n,k)} \big| \boldsymbol{x}_{i(n)}\big).$$

- This could be used as a cost function with all parameters described in the set of parameters $\boldsymbol{\theta}$:

$$\tilde{J}(\boldsymbol{\theta}) \;=\; \prod_{n=0}^{N-1} \prod_{k=-K,\, k\neq 0}^{k=K} p\big(\boldsymbol{x}_{i(n,k)} \big| \boldsymbol{x}_{i(n)}, \boldsymbol{\theta}\big).$$

Training – Basic Ideas

Gradient-based optimization

- This could be used as a cost function with all parameters described in the set of parameters $\boldsymbol{\theta}$:

$$\tilde{J}(\boldsymbol{\theta}) = \prod_{n=0}^{N-1} \prod_{k=-K,\, k\neq 0}^{k=K} p\big(\boldsymbol{x}_{i(n,k)} \big| \boldsymbol{x}_{i(n)}, \boldsymbol{\theta}\big).$$

- Since probabilities are non-negative, it turns out that a normalized and logarithmic cost function leads to the same optimum (optima), but has benefits, when computing gradients:

$$J(\boldsymbol{\theta}) = \frac{1}{N} \frac{1}{2K+1} \sum_{n=0}^{N-1} \sum_{k=-K,\, k\neq 0}^{k=K} \log\Big(p\big(\boldsymbol{x}_{i(n,k)} \big| \boldsymbol{x}_{i(n)}, \boldsymbol{\theta}\big)\Big).$$

Training – Basic Ideas

Gradient-based optimization

- This could be used as a cost function with all parameters described in the set of parameters $\boldsymbol{\theta}$:

$$\tilde{J}(\boldsymbol{\theta}) \;=\; \prod_{n=0}^{N-1} \prod_{k=-K,\,k\neq 0}^{k=K} p\big(\boldsymbol{x}_{i(n,k)} \big| \boldsymbol{x}_{i(n)}, \boldsymbol{\theta}\big).$$

- Since probabilities are non-negative, it turns out that a normalized and logarithmic cost function leads to the same optimum (optima), but has benefits, when computing gradients:

$$J(\boldsymbol{\theta}) \;=\; \frac{1}{N}\,\frac{1}{2K+1} \sum_{n=0}^{N-1} \sum_{k=-K,\,k\neq 0}^{k=K} \log\Big(p\big(\boldsymbol{x}_{i(n,k)} \big| \boldsymbol{x}_{i(n)}, \boldsymbol{\theta}\big)\Big).$$

- To find a good set of parameters $\boldsymbol{\theta}$, a gradient based correction will be used:

$$\boldsymbol{\theta}^{(m+1)} \;=\; \boldsymbol{\theta}^{(m)} + \mu\,\frac{d}{d\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

  with μ being an appropriately chosen step size and m being the iteration index.

Training – Basic Ideas

How to estimate the conditional probabilities

- Our cost function was

$$J(\boldsymbol{\theta}) = \frac{1}{N}\frac{1}{2K+1}\sum_{n=0}^{N-1}\sum_{k=-K,\,k\neq 0}^{k=K}\log\Big(p\big(\boldsymbol{x}_{i(n,k)}\big|\boldsymbol{x}_{i(n)},\boldsymbol{\theta}\big)\Big).$$

Training – Basic Ideas

How to estimate the conditional probabilities

- Our cost function was

$$J(\boldsymbol{\theta}) \;=\; \frac{1}{N}\,\frac{1}{2K+1}\sum_{n=0}^{N-1}\;\sum_{k=-K,\,k\neq 0}^{k=K} \log\Big(p\big(\boldsymbol{x}_{i(n,k)}\big|\boldsymbol{x}_{i(n)},\boldsymbol{\theta}\big)\Big).$$

- Now we need a way to estimate the conditional probabilities:

$$p\big(\underbrace{\boldsymbol{x}_{\text{context}}}_{\text{Context word}}\big|\underbrace{\boldsymbol{x}_{\text{center}}}_{\text{Center word}},\boldsymbol{\theta}\big).$$

Training – Basic Ideas

How to estimate the conditional probabilities

- Now we need a way to estimate the conditional probabilities:

$$\underbrace{p\big(\boldsymbol{x}_{\text{context}}}_{\text{Context word}}\big|\underbrace{\boldsymbol{x}_{\text{center}}}_{\text{Center word}},\boldsymbol{\theta}\big).$$

- A natural way would be to use Euclidian distances:

$$\big|\big|\boldsymbol{x}_{\text{context}}-\boldsymbol{x}_{\text{center}}\big|\big|^2.$$

Whenever this distance is small, the probability would be high. In case of large distances, the probabilities should be small.

How to estimate the conditional probabilities

- Now we need a way to estimate the conditional probabilities:

$$p\big(\underbrace{\boldsymbol{x}_{\text{context}}}_{\text{Context word}}\big|\underbrace{\boldsymbol{x}_{\text{center}}}_{\text{Center word}},\boldsymbol{\theta}\big).$$

- A natural way would be to use Euclidian distances:

$$\big\|\boldsymbol{x}_{\text{context}}-\boldsymbol{x}_{\text{center}}\big\|^2.$$

  Whenever this distance is small, the probability would be high. In case of large distances, the probabilities should be small.

- Beside the distance approach also dot product approaches combined with exponential functions can be used:

$$p\big(\boldsymbol{x}_{\text{context}}\big|\boldsymbol{x}_{\text{center}},\boldsymbol{\theta}\big) \;\propto\; e^{\boldsymbol{x}_{\text{context}}^{\mathrm{T}}\boldsymbol{x}_{\text{center}}}.$$

Training – Basic Ideas

How to estimate the conditional probabilities

- Beside the distance approach also dot product approaches combined with exponential functions can be used:

$$p\big(\boldsymbol{x}_{\text{context}}\big|\boldsymbol{x}_{\text{center}},\boldsymbol{\theta}\big) \;\propto\; e^{\boldsymbol{x}_{\text{context}}^{\text{T}}\,\boldsymbol{x}_{\text{center}}}.$$

- To ensure the rules of probabilities a normalization is added:

$$p\big(\boldsymbol{x}_{\text{context}}\big|\boldsymbol{x}_{\text{center}},\boldsymbol{\theta}\big) \;=\; \frac{e^{\boldsymbol{x}_{\text{context}}^{\text{T}}\,\boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{n}^{\text{T}}\,\boldsymbol{x}_{\text{center}}}}.$$

Training – Basic Ideas

How to estimate the conditional probabilities

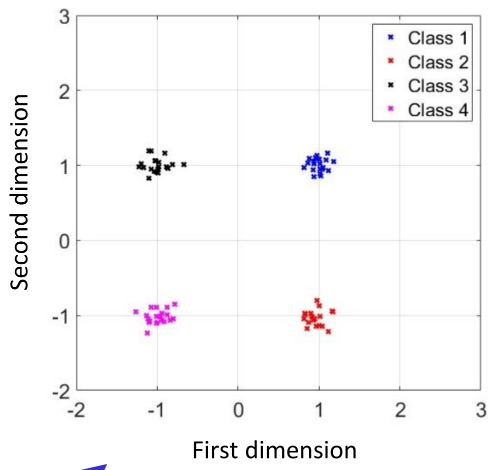
- To ensure the rules of probabilities a normalization is added:

$$p\left(\boldsymbol{x}_{\text{context}} \middle| \boldsymbol{x}_{\text{center}}, \boldsymbol{\theta}\right) = \frac{e^{\boldsymbol{x}_{\text{context}}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_n^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}.$$

- To visualize this, a simple example with a small vector dimension

$$\boldsymbol{x}_n = \left[x_{n,0},\, x_{n,1}\right]^{\mathrm{T}}$$

  is depicted on the left.



4 times 20 vectors

Training – Basic Ideas

How to estimate the conditional probabilities

- To ensure the rules of probabilities a normalization is added:

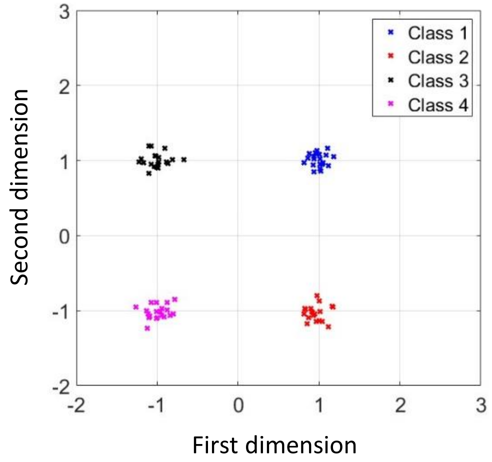
$$p\big(\boldsymbol{x}_{\text{context}}\big|\boldsymbol{x}_{\text{center}},\boldsymbol{\theta}\big) \;=\; \frac{e^{\boldsymbol{x}_{\text{context}}^{\mathrm{T}}\,\boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_n^{\mathrm{T}}\,\boldsymbol{x}_{\text{center}}}}.$$

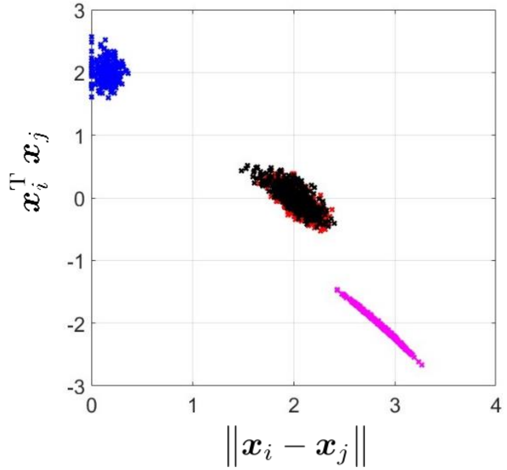
- To visualize this, a simple example with a small vector dimension

$$\boldsymbol{x}_n \;=\; \big[x_{n,0},\, x_{n,1}\big]^{\mathrm{T}}$$

  is depicted on the left.

- First, the Euclidian distance versus the dot product is depicted.





Training – Basic Ideas

How to estimate the conditional probabilities

- To ensure the rules of probabilities a normalization is added:

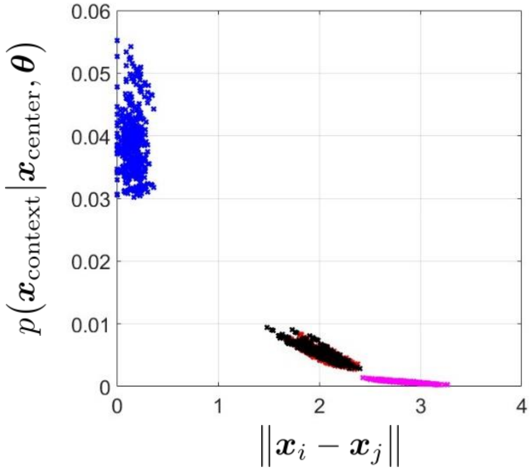
$$p\big(\boldsymbol{x}_{\text{context}}\big|\boldsymbol{x}_{\text{center}}, \boldsymbol{\theta}\big) = \frac{e^{\boldsymbol{x}_{\text{context}}^{\mathrm{T}}\,\boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_n^{\mathrm{T}}\,\boldsymbol{x}_{\text{center}}}}.$$

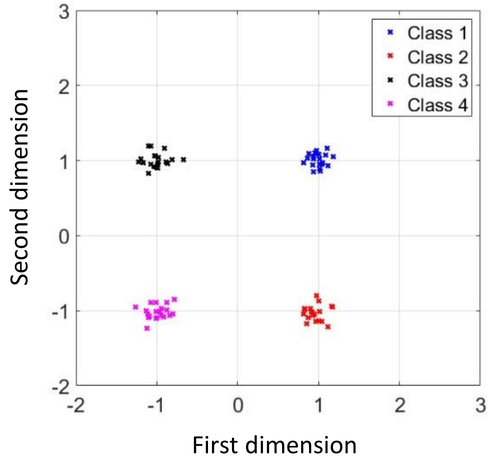
- To visualize this, a simple example with a small vector dimension

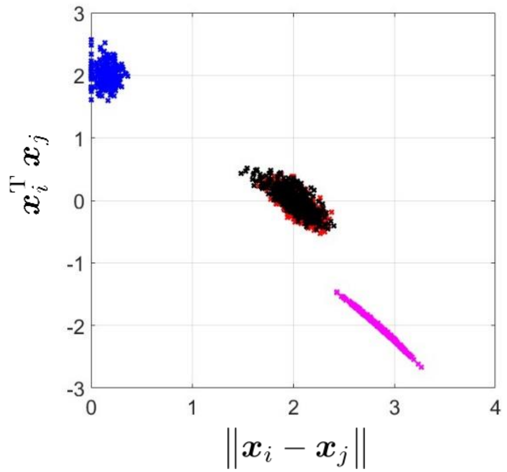
$$\boldsymbol{x}_n = \big[x_{n,0},\, x_{n,1}\big]^{\mathrm{T}}$$

  is depicted on the left.

- First, the Euclidian distance versus the dot product is depicted.
- Secondly, the probability approach vs the Euclidian distance is depicted.







Training of Vector Models

Back to the gradient-based update:

- We started with the following cost function:

$$J(\boldsymbol{\theta}) \;=\; \frac{1}{N}\,\frac{1}{2K+1}\sum_{n=0}^{N-1}\;\sum_{k=-K,\,k\neq 0}^{k=K}\log\Big(p\big(\boldsymbol{x}_{i(n,k)}\big|\boldsymbol{x}_{i(n)},\boldsymbol{\theta}\big)\Big).$$

Training of Vector Models

Back to the gradient-based update:

- We started with the following cost function:

$$J(\boldsymbol{\theta}) \;=\; \frac{1}{N}\,\frac{1}{2K+1}\sum_{n=0}^{N-1}\;\sum_{k=-K,\,k\neq 0}^{k=K}\log\Big(p\big(\boldsymbol{x}_{i(n,k)}\big|\boldsymbol{x}_{i(n)},\boldsymbol{\theta}\big)\Big).$$

- This cost function can be (partially) differentiated to obtain gradients:

$$\boldsymbol{\theta}^{(m+1)} \;=\; \boldsymbol{\theta}^{(m)} + \mu\,\frac{d}{d\boldsymbol{\theta}}J(\boldsymbol{\theta}),$$

Training of Vector Models

Back to the gradient-based update:

- We started with the following cost function:

$$J(\boldsymbol{\theta}) = \frac{1}{N}\frac{1}{2K+1}\sum_{n=0}^{N-1}\sum_{k=-K,\,k\neq 0}^{k=K}\log\Big(p\big(\boldsymbol{x}_{i(n,k)}\big|\boldsymbol{x}_{i(n)},\boldsymbol{\theta}\big)\Big).$$

- This cost function can be (partially) differentiated to obtain gradients:

$$\boldsymbol{\theta}^{(m+1)} = \boldsymbol{\theta}^{(m)} + \mu\,\frac{d}{d\boldsymbol{\theta}}J(\boldsymbol{\theta}),$$

- For a specific center word, this gradient can be refined as:

$$\boldsymbol{x}_{\text{center}}^{(m+1)} = \boldsymbol{x}_{\text{center}}^{(m)} + \mu\,\frac{\partial}{\partial\boldsymbol{x}_{\text{center}}}J\Big(\ldots,\,\boldsymbol{x}_{\text{center}}^{(m)},\,\ldots\Big).$$

Training of Vector Models

$$p\big(\boldsymbol{x}_{\text{context},k}\big|\boldsymbol{x}_{\text{center}},\boldsymbol{\theta}\big) \;=\; \frac{e^{\boldsymbol{x}^{\text{T}}_{\text{context},k}\,\boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}^{\text{T}}_{\text{context},n}\,\boldsymbol{x}_{\text{center}}}}.$$

Back to the gradient-based update:

- We started with the following cost function:

$$J(\boldsymbol{\theta}) \;=\; \frac{1}{N}\,\frac{1}{2K+1}\sum_{n=0}^{N-1}\;\sum_{k=-K,\,k\neq 0}^{k=K}\log\Big(p\big(\boldsymbol{x}_{i(n,k)}\big|\boldsymbol{x}_{i(n)},\boldsymbol{\theta}\big)\Big).$$

- We can insert our ansatz for the conditional probabilities:

$$J(\boldsymbol{\theta}) \;=\; \frac{1}{N}\,\frac{1}{2K+1}\sum_{n=0}^{N-1}\;\sum_{k=-K,\,k\neq 0}^{k=K}\log\left(\frac{e^{\boldsymbol{x}^{\text{T}}_{\text{context},i(n,\,k)}\,\boldsymbol{x}_{\text{center},i(n)}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}^{\text{T}}_{\text{context},n}\,\boldsymbol{x}_{\text{center}}}}\right).$$

- For a specific center word, this gradient can be refined as:

$$\boldsymbol{x}^{(m+1)}_{\text{center}} \;=\; \boldsymbol{x}^{(m)}_{\text{center}} + \mu\,\frac{\partial}{\partial\boldsymbol{x}_{\text{center}}}J\Big(\ldots,\,\boldsymbol{x}^{(m)}_{\text{center}},\,\ldots\Big).$$

Training of Vector Models

Back to the gradient-based update:

- When going over a training text, we can correct the center word vector with a correction going into the following direction

$$\boldsymbol{x}_{\text{center}}^{(m+1)} = \boldsymbol{x}_{\text{center}}^{(m)} + \mu \, \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} J\Big(\ldots,\, \boldsymbol{x}_{\text{center}}^{(m)},\, \ldots\Big).$$

$$= \boldsymbol{x}_{\text{center}}^{(m)} + \mu \, \frac{1}{N} \, \frac{1}{2K+1} \, \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},i(n,\,k)}^{\mathrm{T}} \boldsymbol{x}_{\text{center},i(n)}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \right).$$

$p\big(w_{n+3} \big| w_n\big)$

… like to be in the gym for …

Center word at position n

Context words at position $n+1,\ n+2,\ \ldots$

- We can insert our ansatz for the conditional probabilities:

$$J(\boldsymbol{\theta}) = \frac{1}{N} \, \frac{1}{2K+1} \sum_{n=0}^{N-1} \, \sum_{k=-K,\, k \neq 0}^{k=K} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},i(n,\,k)}^{\mathrm{T}} \boldsymbol{x}_{\text{center},i(n)}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \right).$$

Training of Vector Models

Back to the gradient-based update:

- When going over a training text, we can correct the center word vector with a correction going into the following direction

$$\boldsymbol{x}_{\text{center}}^{(m+1)} = \boldsymbol{x}_{\text{center}}^{(m)} + \mu \frac{1}{N} \frac{1}{2K+1} \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},i(n,\,k)}^{\text{T}} \boldsymbol{x}_{\text{center},i(n)}}}{\sum_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\text{T}} \boldsymbol{x}_{\text{center}}}} \right).$$

- Let us focus now on the gradient for a specific center word and a specific context word:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\text{T}} \boldsymbol{x}_{\text{center}}}}{\sum_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\text{T}} \boldsymbol{x}_{\text{center}}}} \right).$$

Back to the gradient-based update:

- Let us focus now on the gradient for a specific center word and a specific context word:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \right).$$

- First of all, we can change the log of a ratio into a subtraction of individual logs:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \right) = \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}} \right) - \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\sum_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}} \right).$$

Back to the gradient-based update:

- First of all, we can change the log of a ratio into a subtraction of individual logs:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}\right) = \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}\right) - \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\sum_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}\right).$$

- In the first term the log and the exponential function are inverse to each other and, thus, can be cancelled:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}\right) = \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}} - \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\sum_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}\right).$$

Back to the gradient-based update:

- In the first term the log and the exponential function are inverse to each other and, thus, can be cancelled:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\text{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\text{T}} \boldsymbol{x}_{\text{center}}}}\right) = \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \boldsymbol{x}_{\text{context},k}^{\text{T}} \boldsymbol{x}_{\text{center}} - \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\sum_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\text{T}} \boldsymbol{x}_{\text{center}}}\right).$$

- Now the computation of the first gradient is pretty simple:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\text{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\text{T}} \boldsymbol{x}_{\text{center}}}}\right) = \boldsymbol{x}_{\text{context},k} - \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\sum_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\text{T}} \boldsymbol{x}_{\text{center}}}\right).$$

Training of Vector Models

Back to the gradient-based update:

- Now the computation of the first gradient is pretty simple:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \right) = \boldsymbol{x}_{\text{context},k} - \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\sum_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}} \right).$$

- Next we can apply the “chain rule” for the second part:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \right) = \boldsymbol{x}_{\text{context},k} - \frac{1}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \sum_{i=0}^{N-1} e^{\boldsymbol{x}_{\text{context},i}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}.$$

Training of Vector Models

Back to the gradient-based update:

- Next we can apply the "chain rule" for the second part:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\frac{e^{\boldsymbol{x}^{\mathrm{T}}_{\text{context},k}\boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}^{\mathrm{T}}_{\text{context},n}\boldsymbol{x}_{\text{center}}}}\right) = \boldsymbol{x}_{\text{context},k} - \frac{1}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}^{\mathrm{T}}_{\text{context},n}\boldsymbol{x}_{\text{center}}}} \frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \sum_{i=0}^{N-1} e^{\boldsymbol{x}^{\mathrm{T}}_{\text{context},i}\boldsymbol{x}_{\text{center}}}.$$

- Now exchanging the order of the partial differentiation and the sum (both are linear operators) in the last term:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\frac{e^{\boldsymbol{x}^{\mathrm{T}}_{\text{context},k}\boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}^{\mathrm{T}}_{\text{context},n}\boldsymbol{x}_{\text{center}}}}\right) = \boldsymbol{x}_{\text{context},k} - \frac{1}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}^{\mathrm{T}}_{\text{context},n}\boldsymbol{x}_{\text{center}}}} \sum_{i=0}^{N-1} \frac{\partial\, e^{\boldsymbol{x}^{\mathrm{T}}_{\text{context},i}\boldsymbol{x}_{\text{center}}}}{\partial \boldsymbol{x}_{\text{center}}}.$$

Training of Vector Models

Back to the gradient-based update:

- Now exchanging the order of the partial differentiation and the sum (both are linear operators) in the last term:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}^{\text{T}}_{\text{context},k} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}^{\text{T}}_{\text{context},n} \boldsymbol{x}_{\text{center}}}} \right) = \boldsymbol{x}_{\text{context},k} - \frac{1}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}^{\text{T}}_{\text{context},n} \boldsymbol{x}_{\text{center}}}} \sum_{i=0}^{N-1} \frac{\partial\, e^{\boldsymbol{x}^{\text{T}}_{\text{context},i} \boldsymbol{x}_{\text{center}}}}{\partial \boldsymbol{x}_{\text{center}}}.$$

- Applying again the “chain rule” for the last term leads to:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}^{\text{T}}_{\text{context},k} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}^{\text{T}}_{\text{context},n} \boldsymbol{x}_{\text{center}}}} \right) = \boldsymbol{x}_{\text{context},k} - \frac{1}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}^{\text{T}}_{\text{context},n} \boldsymbol{x}_{\text{center}}}} \sum_{i=0}^{N-1} e^{\boldsymbol{x}^{\text{T}}_{\text{context},i} \boldsymbol{x}_{\text{center}}}\, \boldsymbol{x}_i.$$

Training of Vector Models

Back to the gradient-based update:

- Applying again the "chain rule" for the last term leads to:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \right) = \boldsymbol{x}_{\text{context},k} - \frac{1}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \sum_{i=0}^{N-1} e^{\boldsymbol{x}_{\text{context},i}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}} \boldsymbol{x}_i.$$

- Finally we rearrange the sums a bit:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \right) = \boldsymbol{x}_{\text{context},k} - \sum_{i=0}^{N-1} \frac{e^{\boldsymbol{x}_{\text{context},i}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \boldsymbol{x}_{\text{context},i}.$$

Training of Vector Models

Back to the gradient-based update:

$$p\left(\boldsymbol{x}_{\text{context},i} \middle| \boldsymbol{x}_{\text{center}}, \boldsymbol{\theta}\right) = \frac{e^{\boldsymbol{x}_{\text{context},i}^{\text{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\text{T}} \boldsymbol{x}_{\text{center}}}}$$

- Finally we rearrange the sums a bit:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\text{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\text{T}} \boldsymbol{x}_{\text{center}}}}\right) = \boldsymbol{x}_{\text{context},k} - \sum_{i=0}^{N-1} \frac{e^{\boldsymbol{x}_{\text{context},i}^{\text{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\text{T}} \boldsymbol{x}_{\text{center}}}} \boldsymbol{x}_{\text{context},i}.$$

- By remembering the definition of the conditional probability, we can rewrite the “weight”:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log\left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\text{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\text{T}} \boldsymbol{x}_{\text{center}}}}\right) = \boldsymbol{x}_{\text{context},k} - \sum_{i=0}^{N-1} p\left(\boldsymbol{x}_{\text{context},i} \middle| \boldsymbol{x}_{\text{center}}, \boldsymbol{\theta}^{(m)}\right) \boldsymbol{x}_{\text{context},i}.$$

Training of Vector Models

Back to the gradient-based update:

- By remembering the definition of the conditional probability, we can rewrite the “weight”:

$$\frac{\partial}{\partial \boldsymbol{x}_{\text{center}}} \log \left(\frac{e^{\boldsymbol{x}_{\text{context},k}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}}{\sum\limits_{n=0}^{N-1} e^{\boldsymbol{x}_{\text{context},n}^{\mathrm{T}} \boldsymbol{x}_{\text{center}}}} \right) = \boldsymbol{x}_{\text{context},k} - \sum_{i=0}^{N-1} p\big(\boldsymbol{x}_{\text{context},i} \big| \boldsymbol{x}_{\text{center}}, \boldsymbol{\theta}^{(m)}\big)\, \boldsymbol{x}_{\text{context},i}.$$

- This finally leads a correction step if a certain context and center word combination is seen during the training:

$$\boldsymbol{x}_{\text{center}}^{(m+1)} = \boldsymbol{x}_{\text{center}}^{(m)} + \tilde{\mu} \left(\boldsymbol{x}_{\text{context},k}^{(m)} - \sum_{i=0}^{N-1} p\big(\boldsymbol{x}_{\text{context},i}^{(m)} \big| \boldsymbol{x}_{\text{center}}^{(m)}\big)\, \boldsymbol{x}_{\text{context},i}^{(m)} \right).$$

with $$p\big(\boldsymbol{x}_{\text{context},i}^{(m)} \big| \boldsymbol{x}_{\text{center}}^{(m)}\big) = \frac{e^{[\boldsymbol{x}_{\text{context},i}^{(m)}]^{\mathrm{T}} \boldsymbol{x}_{\text{center}}^{(m)}}}{\sum\limits_{n=0}^{N-1} e^{[\boldsymbol{x}_{\text{context},n}^{(m)}]^{\mathrm{T}} \boldsymbol{x}_{\text{center}}^{(m)}}}.$$

Final Remarks

Back to the gradient-based update:

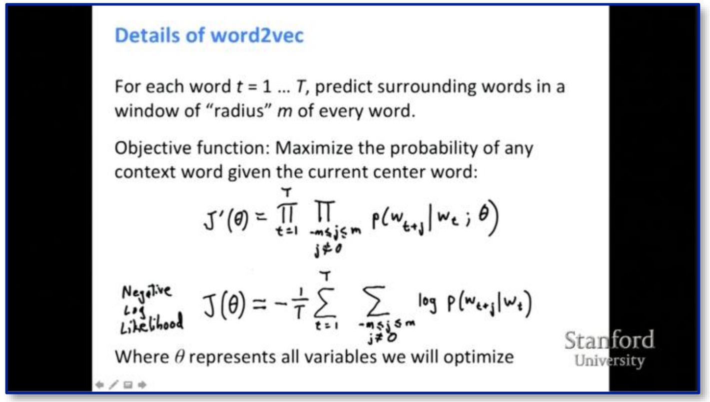
- A similar approach can be used for the training of the context words. However, sometimes the same vectors are used for center and context words.

Final Remarks

Back to the gradient-based update:

- A similar approach can be used for the training of the context words. However, sometimes the same vectors are used for center and context words.
- Stopping of the training is done whenever
  - the cost function (maximization of the “text probability”) does not change significantly any more or
  - after a certain (maximum) amount of iteration steps.

The videos below give further inside into the training of word-to-vector conversions:



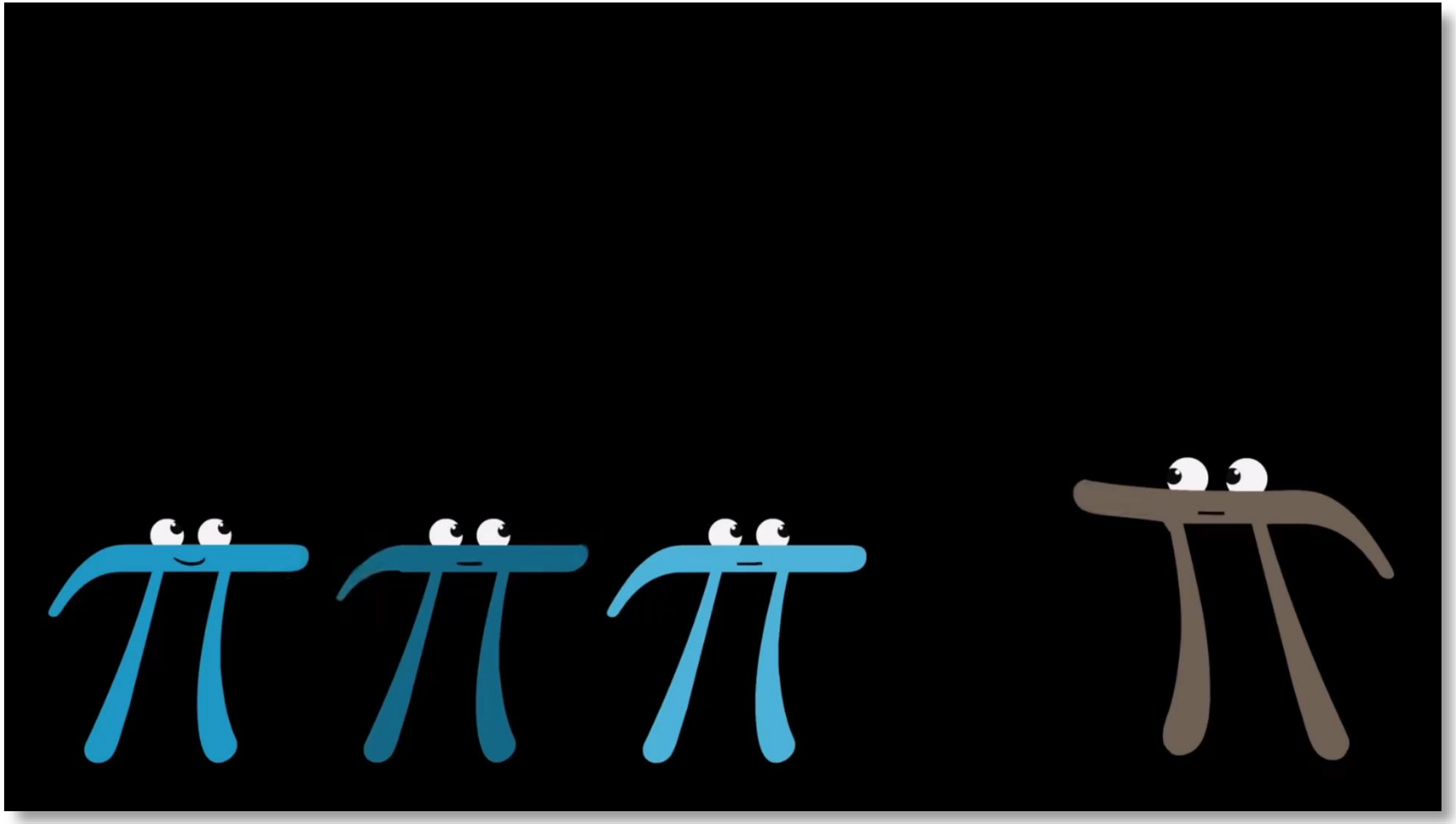
https://www.youtube.com/watch?v=ERibwqs9p38

https://www.youtube.com/watch?v=ASn7ExxLZws

Christian-Albrechts-Universität zu Kiel

Final Remarks

Video from 3Blue1Brown (YouTube)



Summary and Outlook

Summary:

- Motivation
- Focus on text
- Basics (on distance measures)
- Context-word probabilities
- Possibilities to (mathematically) describe words
- Cost functions
- Training – Basic Ideas
- Training of Vector Models
- Final remarks

Next part:

- Gaussian mixture models (GMMs)