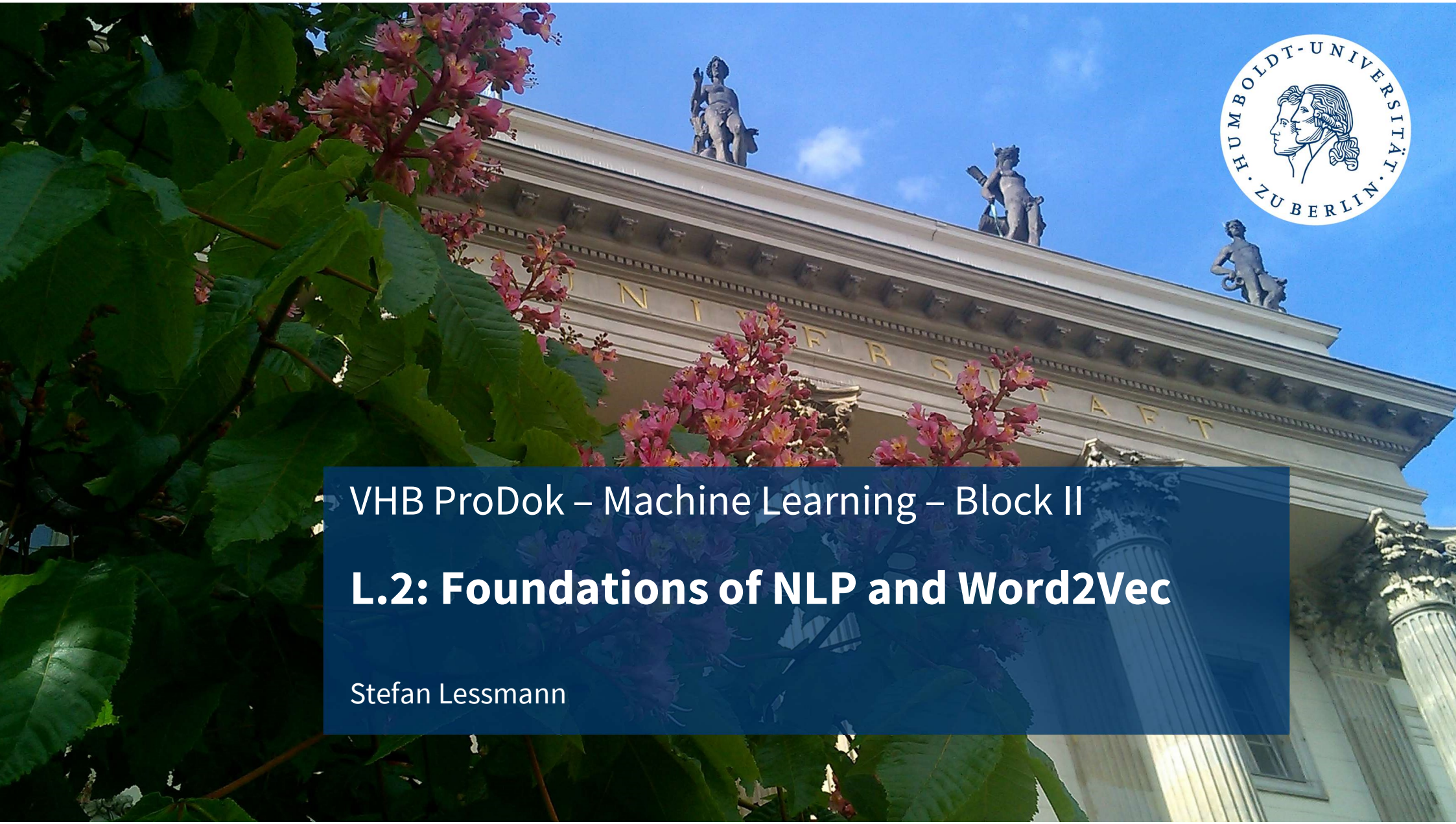
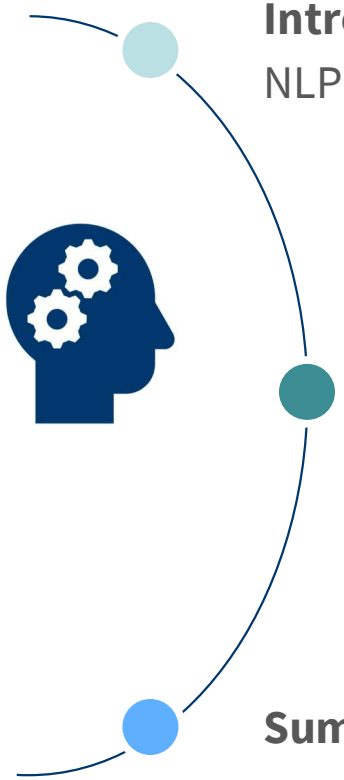


VHB ProDok – Machine Learning – Block II

L.2: Foundations of NLP and Word2Vec

Stefan Lessmann

Agenda



Introduction

NLP applications and bag of words model

Word Embeddings

Theory, motivation, and the word-to-vec (W2V) algorithm

Summary

Introduction

NLP applications and bag of words model

Natural Language Processing (NLP)

A set of linguistic, statistical & ML techniques that capture the informational content of textual data

- **Motivation**
  - Much data not readily available in transactional databases
  - Communication with customers, reports from production workers or mechanics, social media, etc.
  - Text data = asset → information within textual data can contribute greatly toward analytic models
- **Business research examples**
  - Consumer behavior: textual data can shed light on a variety of behavioral and attitudinal traits
  - Innovation management: patent data as cue on technological progress
  - ESG/CSR reports can shed light on companies' approach toward sustainability
- **Common approach**
  - Extract information from text to create auxiliary features for descriptive/explanatory/predictive models
  - Financial forecasting and market efficiency, accounting/tax fraud detection, …
- **NLP enables computers to process, analyze, and ultimately understand textual data including contextual nuances and capturing the informational content within such data**

Common NLP Applications

Good solutions to upstream tasks help solve downstream tasks

Upstream applications

- **Part-of-speech tagging**
  - Identify grammatical role of a piece of text
  - Noun, verb, adjective, adverb, etc.
  - Rule-/ dictionary-/ model-based
- **Named Entity recognition**
  - Locate and classify named entities
  - "*The price of Apple rose yesterday*"
- **Phrase detection**
- **Language modeling**
- **…**

Downstream applications

- **Text classification & clustering**
- **Text summarization**
- **Sentiment analysis**
- **Topic modeling**
- **Machine translation**
- **Question answering**
- **Text generation**
- **…**

Upstream NLP Exemplified
Part-of-Speech (POS) Tagging

- **Identify the grammatical role of a token**
  - Noun, verb, adjective, etc.
  - Typically done at the level of words
  - Dictionary-, rule-, model-based
- **Standard (up-stream) NLP task**
  - Typically supports other NLP tasks
  - Example is lemmatization: dictionary form of a word may depend on its grammatical role

NNP	VBZ	VBN	TO	VB	RB
Ronaldo	is	expected	to	play	tomorrow
NNP	VBZ	VBZ	TO	NN	RB
Ronaldo	is	expected	to	play	tomorrow

No.	Tag	Description
1.	CC	Coordinating conjunction
7.	JJ	Adjective
8.	JJR	Adjective, comparative
13.	NNS	Noun, plural
14.	NNP	Proper noun, singular
15.	NNPS	Proper noun, plural
19.	PRP$	Possessive pronoun
20.	RB	Adverb
21.	RBR	Adverb, comparative
24.	SYM	Symbol
25.	TO	*to*
27.	VB	Verb, base form
29.	VBG	Verb, gerund or present participle
30.	VBN	Verb, past participle
32.	VBZ	Verb, 3rd person singular present

https://catalog.ldc.upenn.edu/docs/LDC95T7/cl93.html

Intuition:

```
Given dictionary with known nouns,
verbs, etc.
Go through corpus
  Calc frequency of nouns following ‘to’
  Calc frequency of verbs following to’
```

$$P(NN|TO) = 0.0047$$
$$P(VB|TO) = 0.83$$

Bag of Word Model and Document Term Matrix

Early form to represent textual data as a document-term matrix (DTM)

- **Corpus**
  - Collection of documents
  - E-mails, news articles, tweets, etc.
- **Vocabulary**
  - Set of unique words in a language/corpus
  - After NLP preprocessing (stop word filtering, removal of word forms, etc.)
- **Text representation**
  - Documents are characterized by the occurrence (or counts, or weights) of words
  - Very similar to structural data where feature values characterize observations
  - Facilitates document-level analysis (e.g., classification) or word-level analysis

Vocabulary

Docs. in corpus

ID	MOVIE	WORST	ENJOY	I	COOL	…
1	0	1	0	1	…	…
2	0	0	1	0	…	…
3	1	0	1	0	…	…
4	1	1	1	1	…	…
…	…	…	…	…	…	…

NLP Preprocessing Pipeline

Step 1: Tokenization

The lecture ended on time.

- **Split text as a stream of characters into individual tokens**
  - Can be done at word level (e.g., using whitespaces)
  - Typically done at sub-word level today
    - "Boyfriend" → "boy", "friend"
    - More flexible and less tokens
    - Can even consider character level
- **Results in a bag of words/tokens**
  - Information on token order is lost
  - Risk to change semantics of a piece of text

Token	Token	Token	Token	Token	Token
The	Lecture	ended	on	time	.

ID	T-1	T-2	T-3	T-4	T-5	…
1	0	1	0	1	…	…
2	0	0	1	0	…	…
3	1	0	1	0	…	…
4	1	1	1	1	…	…

NLP Preprocessing Pipeline

Step 2: Filtering

The lecture ended on time.

- **Filters for stop word, punctuations, numbers and symbols**
  - Often considered irrelevant for downstream task
  - Reduce number of distinct tokens
- **Based on dictionaries**
  - Specific to individual languages
  - Risk to change semantics of a piece of text

Token	Token	Token	Token	Token	Token
The	Lecture	ended	on	time	.

Token	Token	Token	Token	Token	Token
~~The~~	Lecture	ended	~~on~~	Time	~~.~~

NLP Preprocessing Pipeline

Step 3: Stemming / Lemmatization

- **Reduce number of distinct tokens**
  - Reducing a term to its most basic element (*stem*)
  - Determining the dictionary form (*lemma*) of word
    - *run*, *runs*, *ran, running* → run
    - am, are, is → be
- **Lemmatization typically superior & more costly**
  - No variance due to stemmer choice
  - Dictionaries and morphological analysis needed
  - Dictionaries are specific to a language

Token	Token	Token
Lecture	end~~ed~~	Time

Token	Token	Token
Lecture	end	Time

	Standard stemmers		
	Porter	**Snowball**	**Lancaster**
Generate	Gener	Generat	Gen
Generates	Gener	Generat	Gen
Generated	Gener	Generat	Gen
Generating	Gener	Generat	Gen
General	Gener	General	Gen
Generally	Gener	General	Gen
Generic	Gener	Generic	Gen
Generically	Gener	Generic	Gen
Generous	Gener	Generous	Gen
Generously	Gener	Generous	Gen
Organization	Organ	organ	Org
urgency	urgen	urgen	Urg

Shortcomings of Bag of Words Model and Count-Based Representations

No understanding of the rich sources of information in language

- **Ignores word order**
  - Contextual information is lost
  - "This is interesting" vs. "Is this interesting"
- **Problems with homonyms**
- **Problems with synonyms**
  - Further complicated through connotations of words
    - "Beth, now 25 years old, became a kind of big sister to Mary"
    - "Beth, now 25 years old, became a kind of large sister to Mary"
  - These sentences to not have the same meaning
- **Several technical challenges related to DTM**
  - High dimensionality (size of vocabulary)
  - Sparsity
  - Long tail

Vocabulary

Docs. in corpus

ID	MOVIE	WORST	ENJOY	…
1	0	1	0	…
2	0	0	1	…
3	1	0	1	…
4	1	1	1	…
…	…	…	…	…

Shortcomings of Bag of Words Model and Count-Based Representations

Example text and one-hot representations

Motels have parking lots

Hotels charge for car parking

Parking is expensive

	X_1	X_2	X_3	...						X_{10}
Motels	1	0	0	0	0	0	0	0	0	0
Have	0	1	0	0	0	0	0	0	0	0
Parking	0	0	1	0	0	0	0	0	0	0
Lots	0	0	0	1	0	0	0	0	0	0
Hotels	0	0	0	0	1	0	0	0	0	0
Charge	0	0	0	0	0	1	0	0	0	0
For	0	0	0	0	0	0	1	0	0	0
Car	0	0	0	0	0	0	0	1	0	0
Is	0	0	0	0	0	0	0	0	1	0
Expensive	0	0	0	0	0	0	0	0	0	1

Shortcomings of Bag of Words Model and Count-Based Representations

No notion of word similarity and thus ability to capture semantic or sintactic patterns

	X_1	X_2	X_3	...					X_{10}
Motels	1	0	0	0	0	0	0	0	0	0
Have	0	1	0	0	0	0	0	0	0	0
Parking	0	0	1	0	0	0	0	0	0	0
Lots	0	0	0	1	0	0	0	0	0	0
Hotels	0	0	0	0	1	0	0	0	0	0
Charge	0	0	0	0	0	1	0	0	0	0
For	0	0	0	0	0	0	1	0	0	0
Car	0	0	0	0	0	0	0	1	0	0
Is	0	0	0	0	0	0	0	0	1	0
expensive	0	0	0	0	0	0	0	0	0	1

Motel = [1 0 0 0 0 0 0 0 0 0]

Hotel = [0 0 0 0 1 0 0 0 0 0]

$$\cos\theta = \frac{\overrightarrow{hotel} \cdot \overrightarrow{motel}}{\left\|\overrightarrow{hotel}\right\| \cdot \left\|\overrightarrow{motel}\right\|} = 0$$

Word Embeddings

Theory, motivation, and the word-to-vec (W2V) algorithm

Word Embeddings

An embedding represents a word (i.e. token) as numerical vector

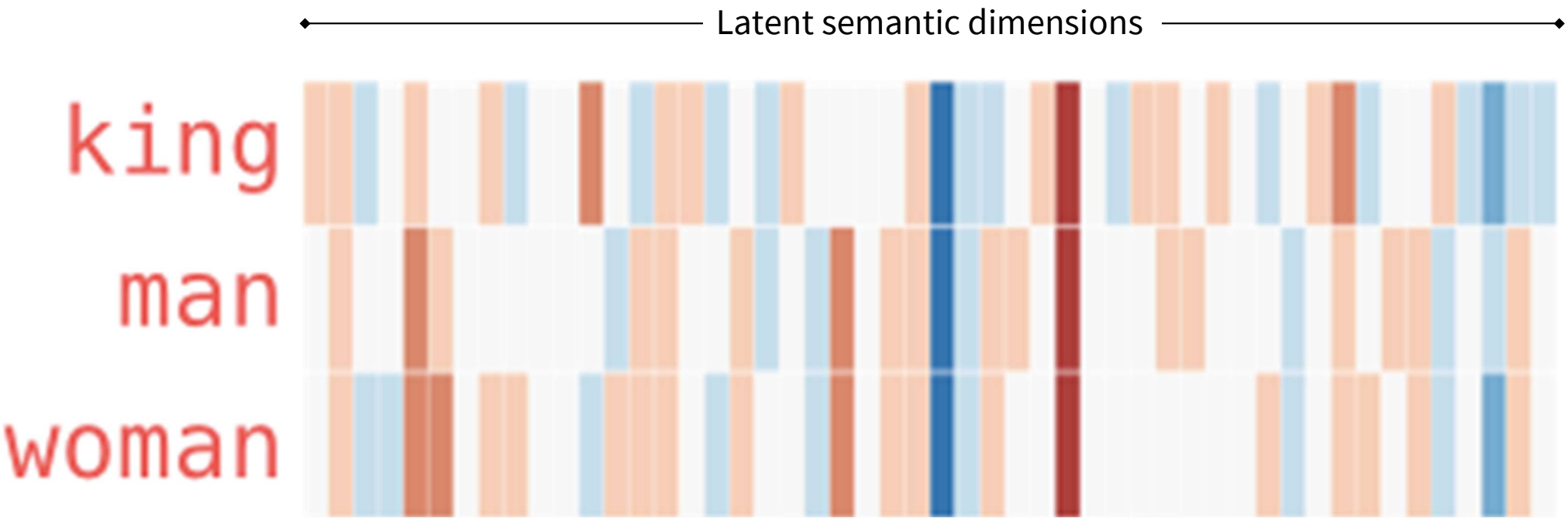
- **Desiderata**
  - Create low dimensional, dense representation of words as vectors of real numbers
  - Ensure that word vectors carry meaning and encompass the semantics of a word
  - Ensure that learnt word representations can be used in downstream tasks (i.e. transfer learning)
- **Distributional hypothesis**
  - The more semantically similar two words are, the more distributionally similar they are
  - The more distributionally similar two words are, the more they will tend to occur in similar contexts
- **Modeling by means of linear algebra**
  - Collect distributional information in high-dimensional vectors
  - Define distributional (i.e., semantic) similarity as vector similarity
  - Perform dimensionality reduction
    - LSA: term co-occurrences (paradigmatic similarity)
    - LDA: context co-occurrence (topical similarity)

You shall know a word by the company it keeps

Firth (1957)

Word Embeddings

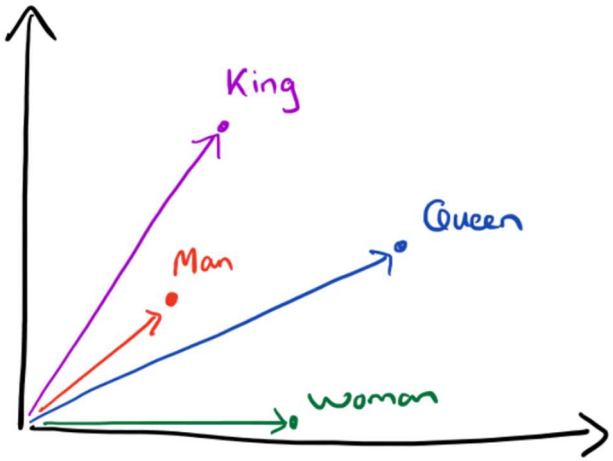
Visual intuition of word embeddings aka word vectors

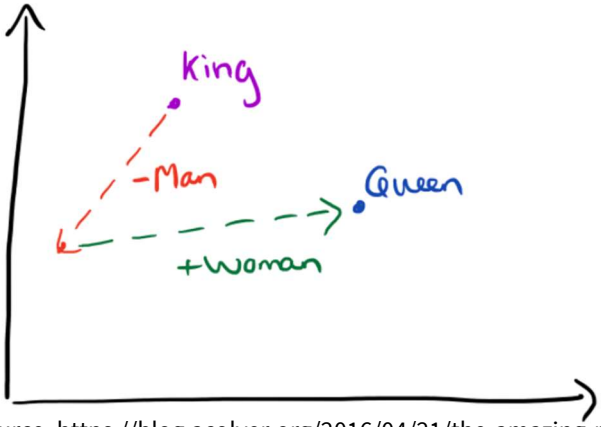


Source: http://jalammar.github.io/illustrated-word2vec/

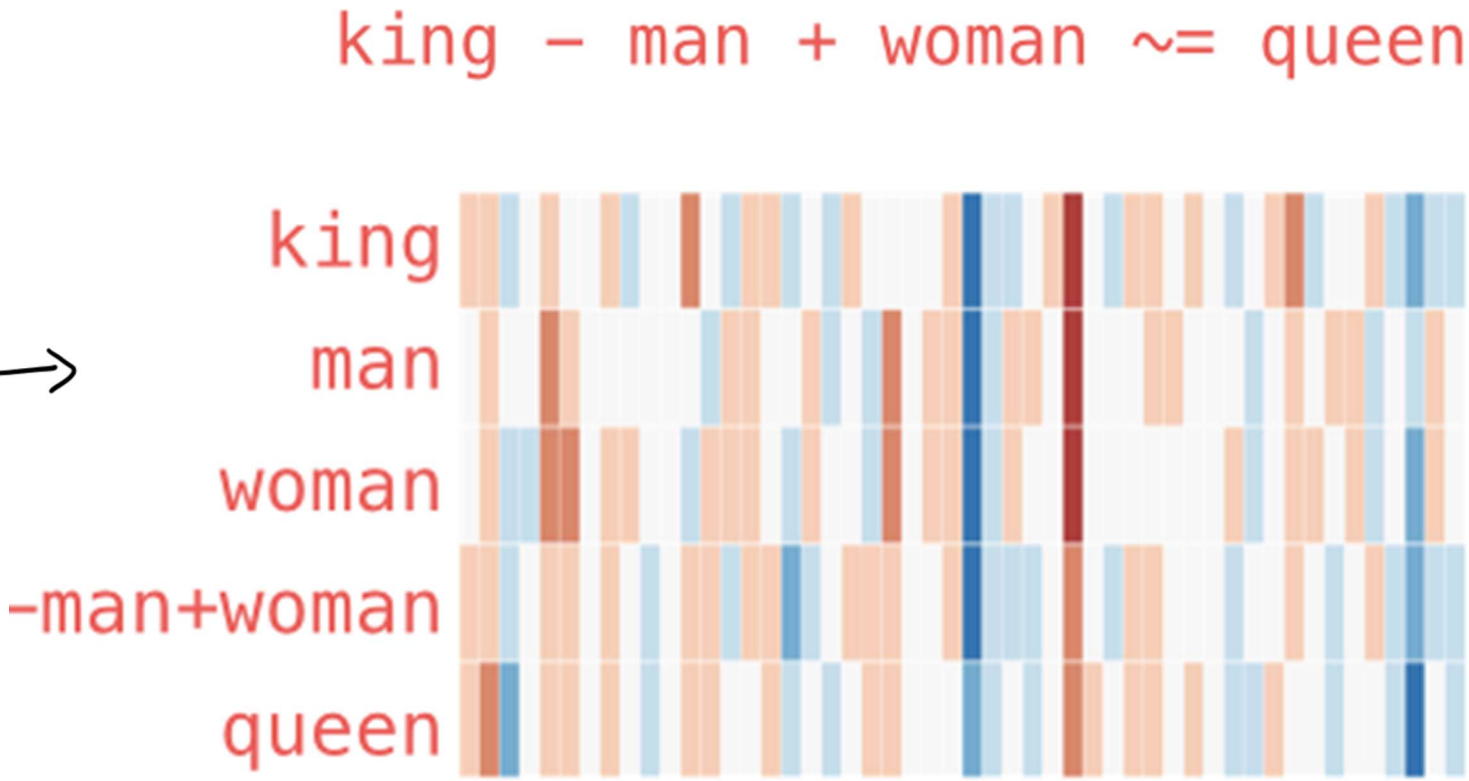
Word Embeddings

Famous example of how semantic relationships are verifiable by algebra





Source: https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/



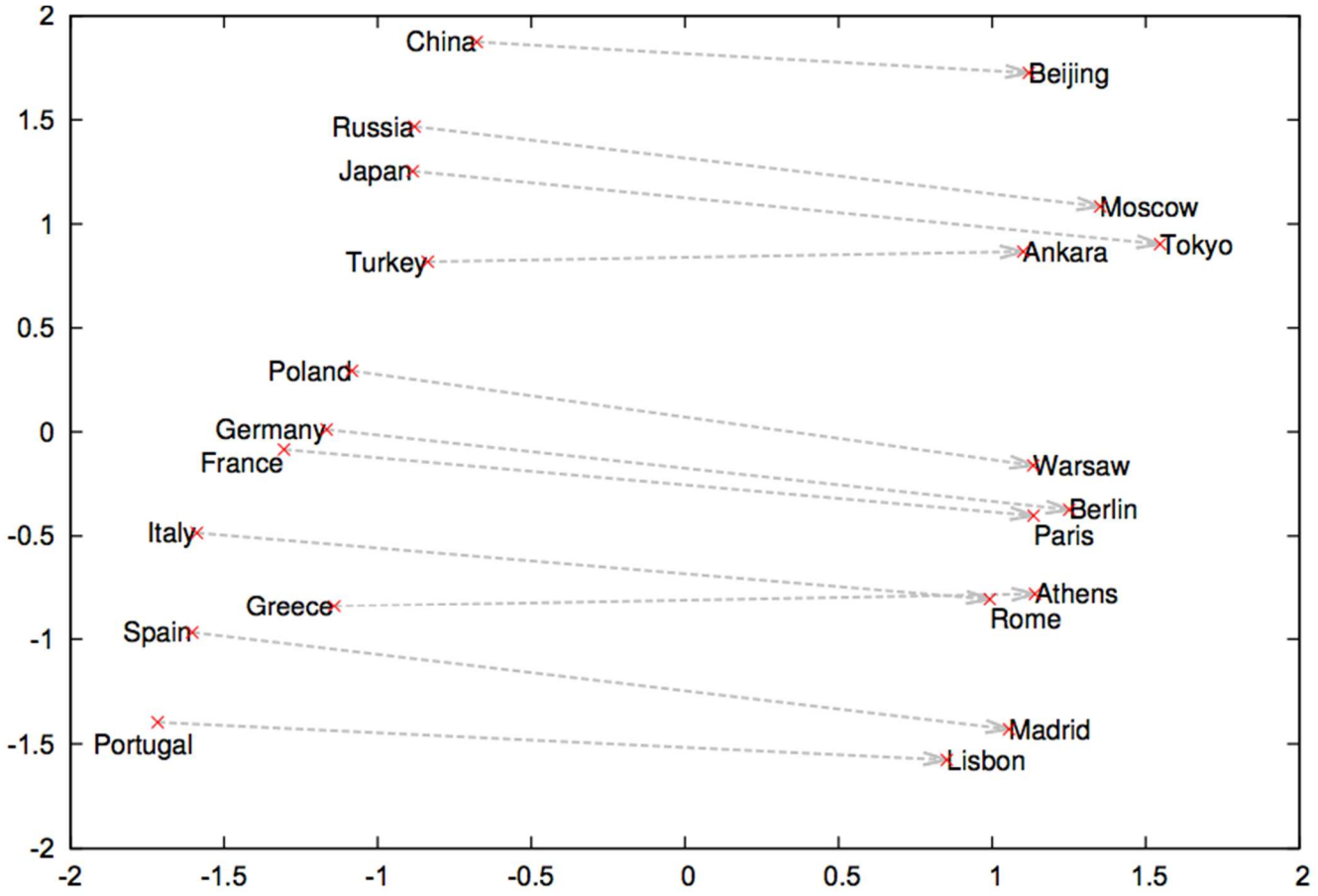
Source: http://jalammar.github.io/illustrated-word2vec/

Word Embeddings

Some examples: countries and capitals

Paris – France + Poland ~=Warsaw

Mapping of 300-dimensional word vectors to low dimensional space for visualization

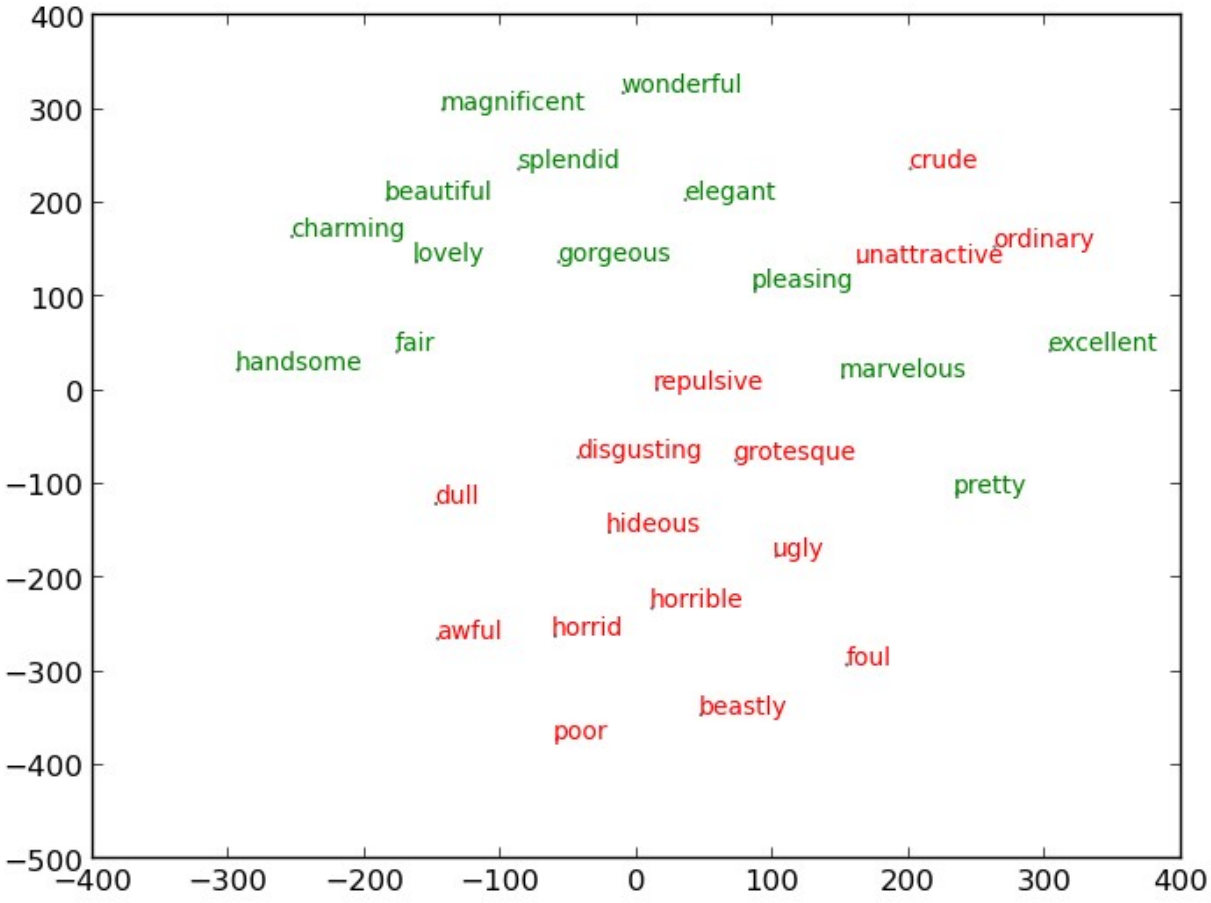


Source: https://algorithmia.com/algorithms/nlp/Word2Vec/docs

Word Embeddings

Some examples: sentiment words

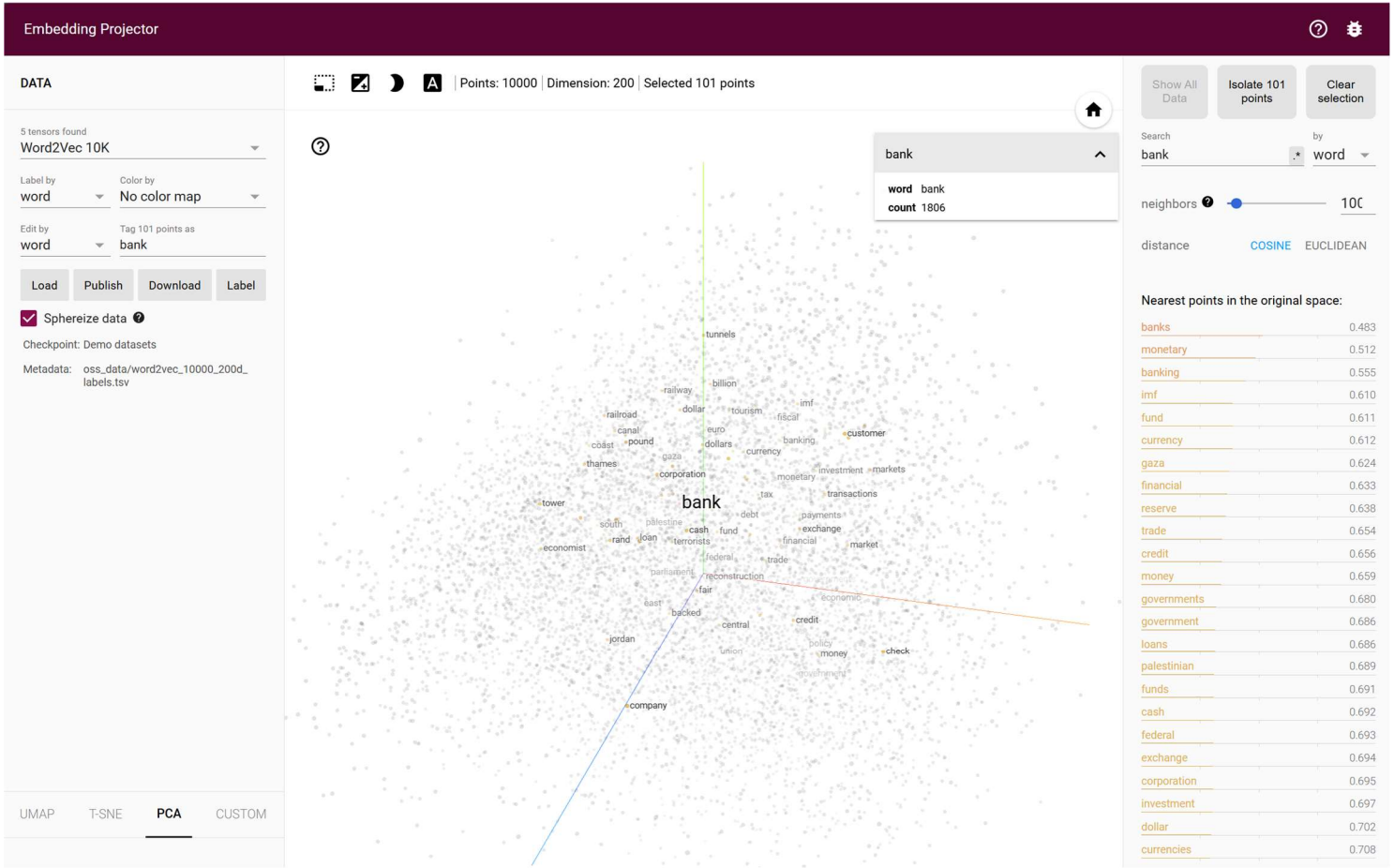
Mapping of 300-dimensional word vectors to low dimensional space for visualization



Source: Chung & Glass (2018)

Tensorflow Embedding Projector

https://projector.tensorflow.org/



The Word-to-Vec Algorithm

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

Abstract

The recently introduced continuous Skip-gram model is an efficient method for learning high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships. In this paper we present several extensions that improve both the quality of the vectors and the training speed. By subsampling of the frequent words we obtain significant speedup and also learn more regular word representations. We also describe a simple alternative to the hierarchical softmax called negative sampling.

An inherent limitation of word representations is their indifference to word order and their inability to represent idiomatic phrases. For example, the meanings of "Canada" and "Air" cannot be easily combined to obtain "Air Canada". Motivated by this example, we present a simple method for finding phrases in text, and show that learning good vector representations for millions of phrases is possible.

Advances in Neural Information Processing Systems 26: Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS 2013), Lake Tahoe, NV, USA.

Tomas Mikolov

FOLLOW

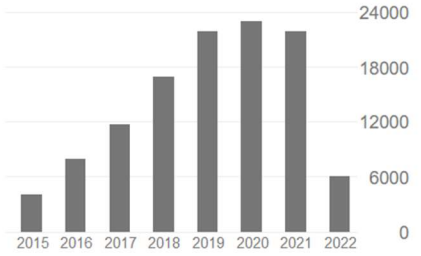
Senior Researcher, CIIRC CTU
Verified email at fb.com

Artificial Intelligence Machine Learning Language Modeling Natural Language Processing

TITLE	CITED BY	YEAR
Distributed representations of words and phrases and their compositionality T Mikolov, I Sutskever, K Chen, GS Corrado, J Dean Neural information processing systems	33397	2013
Efficient estimation of word representations in vector space T Mikolov, K Chen, G Corrado, J Dean arXiv preprint arXiv:1301.3781	28678	2013
Distributed representations of sentences and documents Q Le, T Mikolov International conference on machine learning, 1188-1196	9354	2014
Enriching word vectors with subword information P Bojanowski, E Grave, A Joulin, T Mikolov Transactions of the association for computational linguistics 5, 135-146	8005	2017
Recurrent neural network based language model. T Mikolov, M Karafiát, L Burget, J Cernocký, S Khudanpur Interspeech 2 (3), 1045-1048		

Cited by VIEW ALL

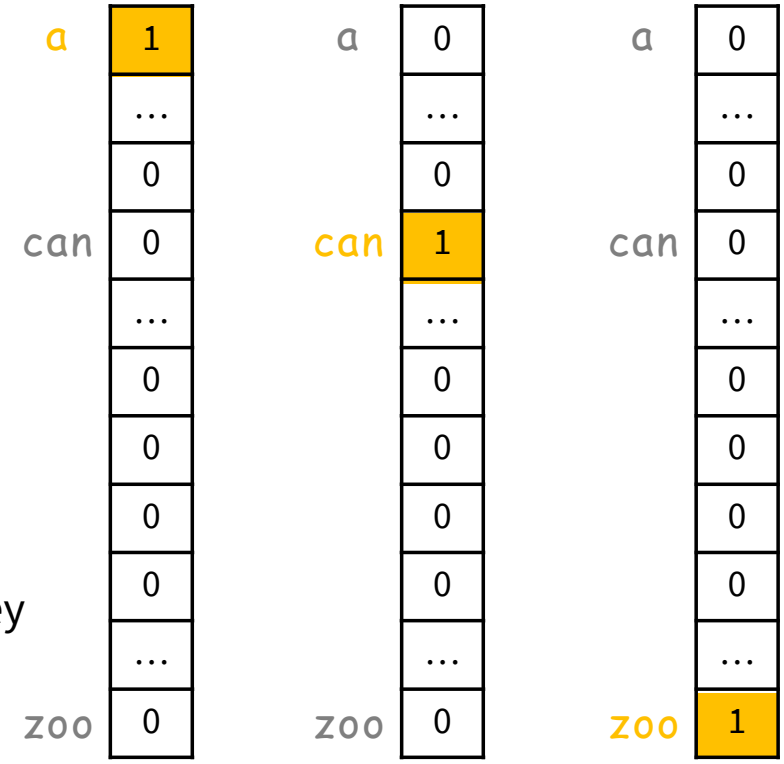
	All	Since 2017
Citations	117461	101844
h-index	47	46
i10-index	83	73



The Word-to-Vec Algorithm

Learn word embeddings using a shallow feedforward neural network

- **Represent every word in a fixed vocabulary as one-hot-vector**
  - Say vocabulary size is V
  - So one-hot-vector of dimension V
- **Core idea in W2V is to predict neighboring words**
- **Recall distributional hypothesis**
  - The more semantically similar two words are, the more distributionally similar they are
  - The more distributionally similar two words are, the more they will tend to occur in similar contexts
  - Predicting neighboring words is like predicting word co-occurrences



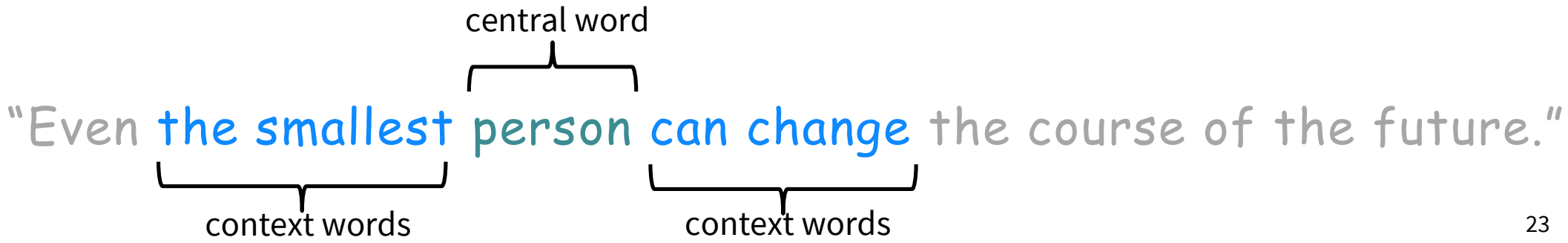
$$\boldsymbol{x} \in \{0,1\}^{V \times 1}$$

The Word-to-Vec Algorithm
Self-supervised training

- **Two flavors of W2V**
  - Predict a center word from surrounding (context) words (continuous bag of words, CBOW)
  - Predict surrounding (context) words given a center word (skip-gram)
- **Example sentence**

"Even the smallest person can change the course of the future."

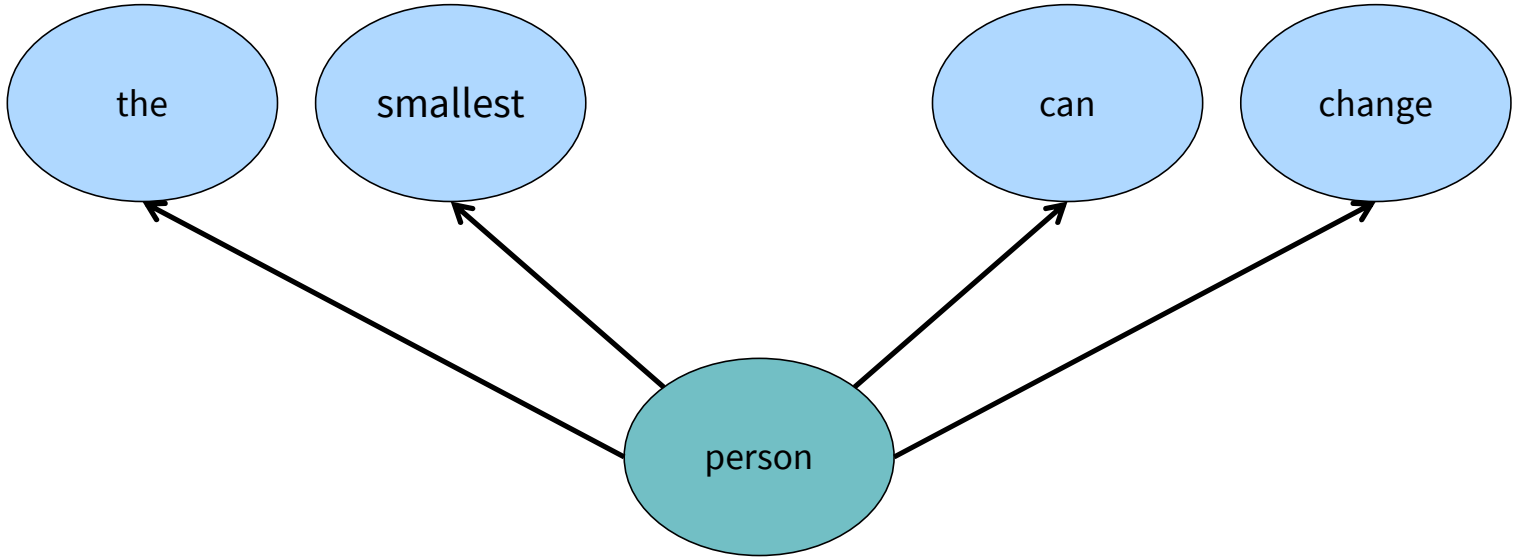
  - Quantify surrounding words by defining a context window
  - Assume we consider a context window of size 2 and take person as our central word



Self-Supervised Training Under the Skip-Gram Model

Conditional probability of generating context words given the center word

"Even the smallest person can change the course of the future."



$$P(\text{the}|\text{person}) \cdot P(\text{smallest}|\text{person}) \cdot P(\text{can}|\text{person}) \cdot P(\text{change}|\text{person})$$

Self-Supervised Training Under the Skip-Gram Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- Training tuples (y_t, x_i) when parsing the example sentence with context size 2
  - (the|even), (smallest|even)

Self-Supervised Training Under the Skip-Gram Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- Training tuples (y, x) from parsing the example sentence with context size 2
  - (the|even), (smallest|even)
  - (even, the), (smallest, the), (person, the)

Self-Supervised Training Under the Skip-Gram Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- Training tuples $(y, x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2})$ from parsing the example sentence with context size 2
  - (the|even), (smallest|even)
  - (even, the), (smallest, the), (person, the)
  - (even, smallest), (the, smallest), (person, smallest), (can, smallest)

Self-Supervised Training Under the Skip-Gram Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- Train $p(y|x)$ using the example sentence with context size 2
  - (the|even), (smallest|even)
  - (even, the), (smallest, the), (person, the)
  - (even, smallest), (the, smallest), (person, smallest), (can, smallest)
  - (the, person), (smallest, person), (can, person), (change, person)

Self-Supervised Training Under the Skip-Gram Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- Training tuples for sentence with context size 2
  - (the|even), (smallest|even)
  - (even, the), (smallest, the), (person, the)
  - (even, smallest), (the, smallest), (person, smallest), (can, smallest)
  - (the, person), (smallest, person), (can, person), (change, person)
  - (smallest, can), (person, can), (change, can), (the, can)

Self-Supervised Training Under the Skip-Gram Model

Generating training data from a (large) corpus

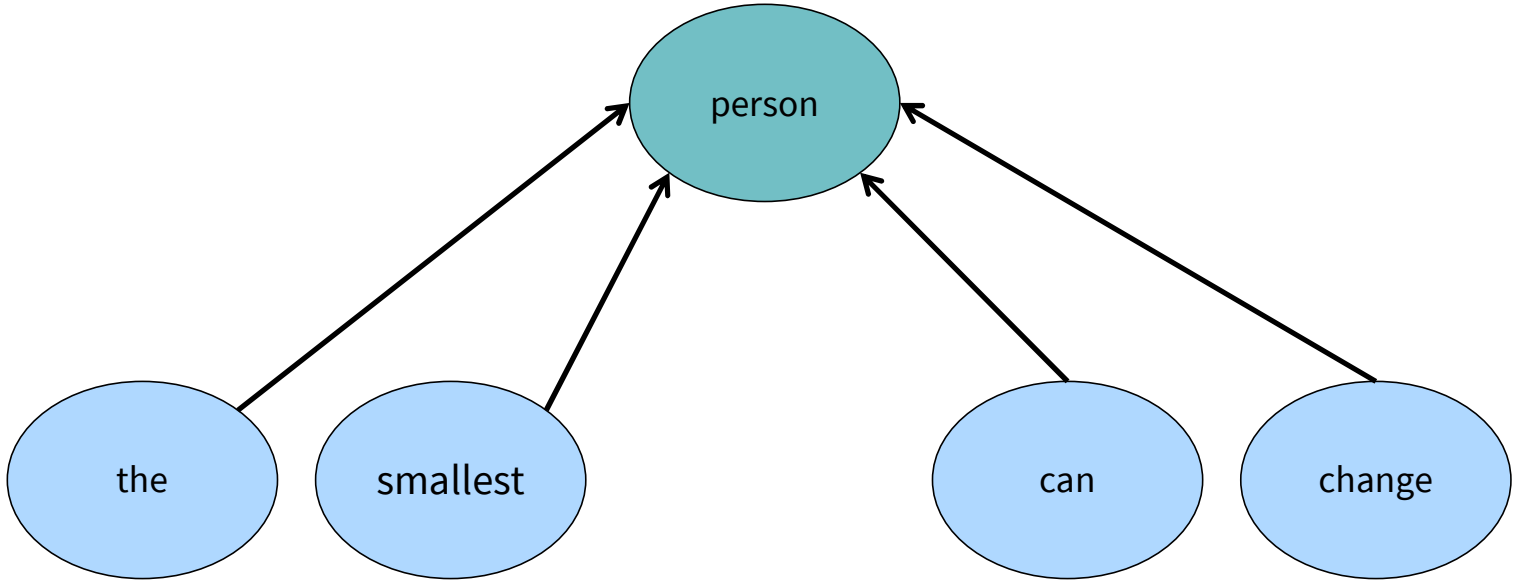
"Even the smallest person can change the course of the future."

- **Training tuples (y_i, x_i) when parsing the example sentence with context size 2**
  - (the|even), (smallest|even)
  - (even, the), (smallest, the), (person, the)
  - (even, smallest), (the, smallest), (person, smallest), (can, smallest)
  - (the, person), (smallest, person), (can, person), (change, person)
  - (smallest, can), (person, can), (change, can), (the, can)
  - (person, change), (can, change), (the, change), (course, change)
  - (can, the), (change, the), (course, the), (of, the)
  - (change, course), (the , course), (of, course), (the, course)
  - (the, of), (course, of), (the, of), (future, of)
  - (course, the), (of, the), (future, the)
  - (of, future), (the , future)

Self-Supervised Training Under the Continuous Bag of Word Model

Conditional probability of context words generating a center word

"Even the smallest person can change the course of the future."



$$P(\text{person}|\text{the}, \text{smallest}, \text{can}, \text{change})$$

Self-Supervised Training Under the CBOW Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- **Training tupels (y_i, X_i) when parsing example sentence with context size 2**
  - ...
  - (smallest, {even, the, person, can})

$$P(x_t \mid x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2})$$

Self-Supervised Training Under the CBOW Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- **Training tupels (y_i, X_i) when parsing example sentence with context size 2**
  - ...
  - (smallest, {even, the, person, can})
  - (person, {the, smallest, can, change})

$$P(x_t \mid x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2})$$

Self-Supervised Training Under the CBOW Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- **Training tupels (y_i, X_i) when parsing example sentence with context size 2**
  - ...
  - (smallest, {even, the, person, can})
  - (person, {the, smallest, can, change})
  - (can, {smallest, person, change, the})

$$P(x_t \mid x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2})$$

Self-Supervised Training Under the CBOW Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- **Training tupels (y_i, X_i) when parsing example sentence with context size 2**
  - ...
  - (smallest, {even, the, person, can})
  - (person, {the, smallest, can, change})
  - (can, {smallest, person, change, the})
  - (change, {smallest, can, the, course})

$$P(x_t \mid x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2})$$

Self-Supervised Training Under the CBOW Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- **Training tupels (y_i, X_i) when parsing example sentence with context size 2**
  - ...
  - (smallest, {even, the, person, can})
  - (person, {the, smallest, can, change})
  - (can, {smallest, person, change, the})
  - (change, {smallest, can, the, course})
  - (the, {can, change, course, of})

$$P(x_t \mid x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2})$$

Self-Supervised Training Under the CBOW Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- **Training tupels (y_i, X_i) when parsing example sentence with context size 2**
  - ...
  - (smallest, {even, the, person, can})
  - (person, {the, smallest, can, change})
  - (can, {smallest, person, change, the})
  - (change, {smallest, can, the, course})
  - (the, {can, change, course, of})
  - (course, {change, the, of, the})

$$P(x_t \mid x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2})$$

Self-Supervised Training Under the CBOW Model

Generating training data from a (large) corpus

"Even the smallest person can change the course of the future."

- **Training tupels (y_i, X_i) when parsing example sentence with context size 2**
  - ...
  - (smallest, {even, the, person, can})
  - (person, {the, smallest, can, change})
  - (can, {smallest, person, change, the})
  - (change, {smallest, can, the, course})
  - (the, {can, change, course, of})
  - (course, {change, the, of, the})
  - (of, {the , course, the, future})
  - ...

$$P(x_t \mid x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2})$$

Self-Supervised Training in Word2Vec

- **Setting up a classification task by parsing a (large) corpus**
- **Skip-Gram model**
  - Conditional probability of a center word generating surrounding context words
  - Formally

$$P(x_{t-2}|x_t) \cdot P(x_{t-1}|x_t) \cdot P(x_{t+1}|x_t) \cdot P(x_{t+2}|x_t)$$

- **Continuous bag of words model**
  - Conditional probability of context words generating a center word
  - Formally

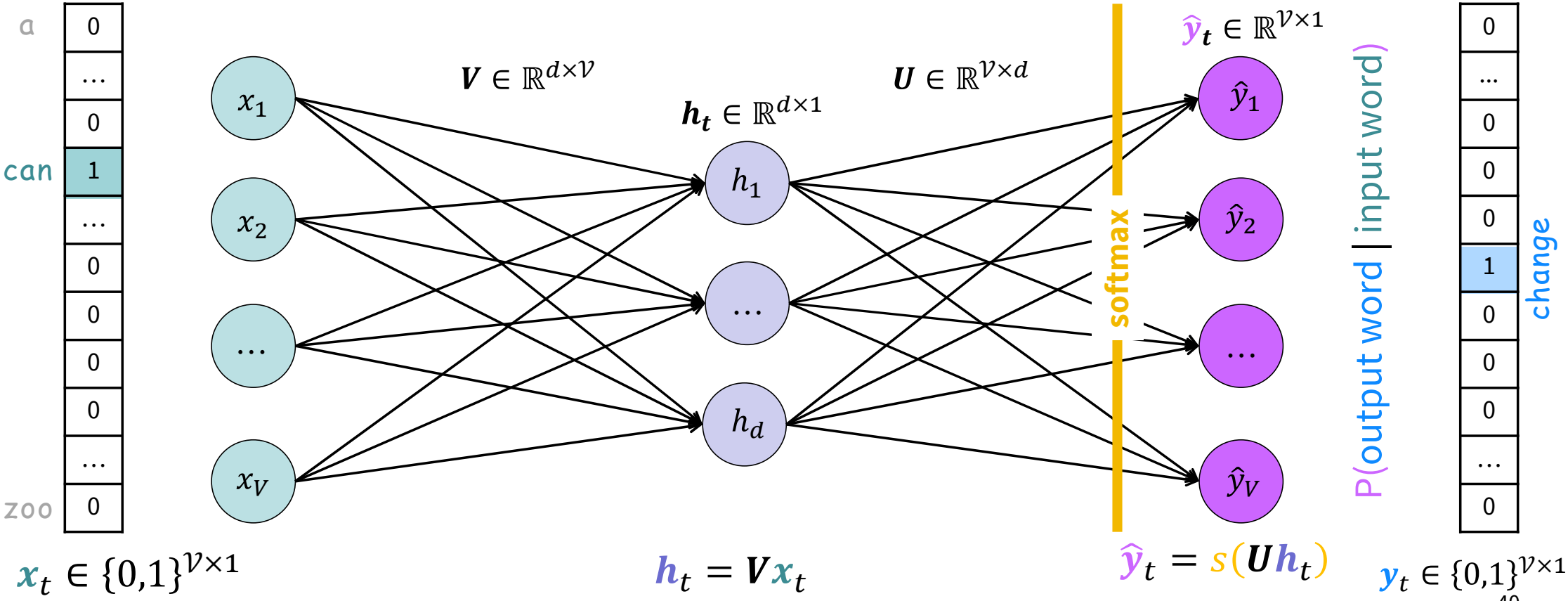
$$P(x_t | x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2})$$

- **Size of the context window is a meta-parameter**
- **No need for target labels or text annotation → self-supervision**

Setting up the Neural Network

Shallow NN with hidden layer size d and two weight matrices $\boldsymbol{V}$ and $\boldsymbol{U}$

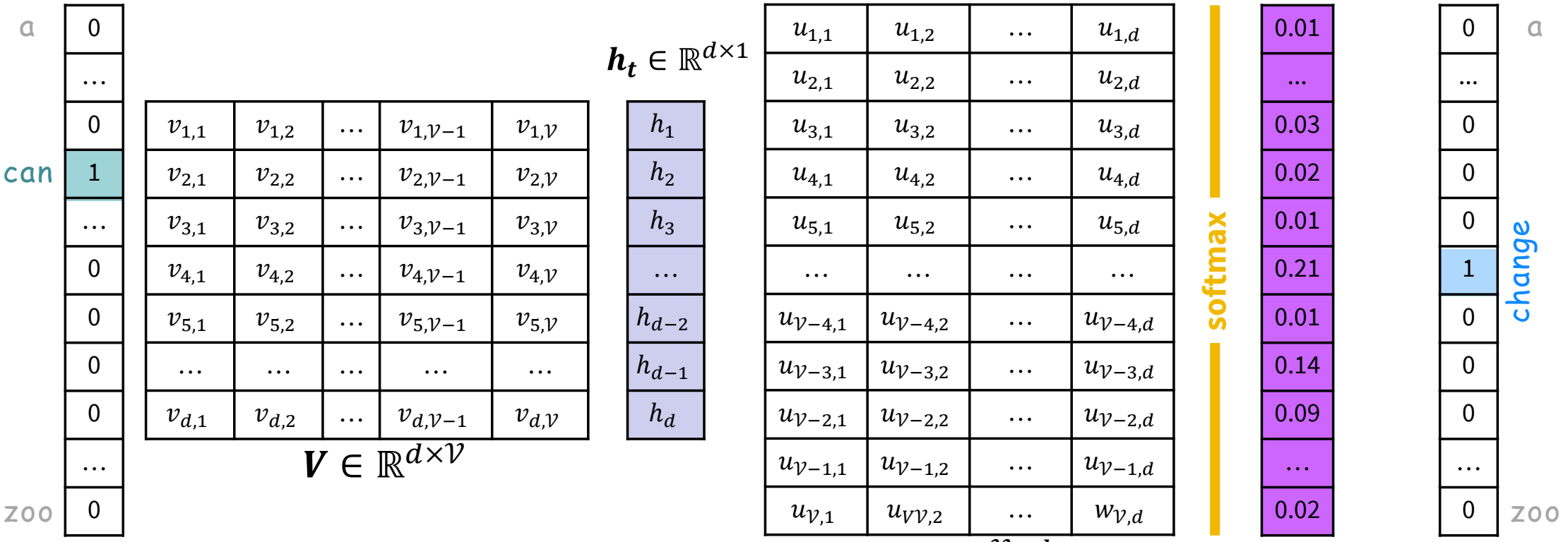
"Even the smallest person can change the course of the future."



Setting up the Neural Network

Shallow NN with hidden layer size d and two weight matrices $\boldsymbol{V}$ and $\boldsymbol{U}$

"Even the smallest person can change the course of the future."



Setting up the Neural Network

Dot product with one-hot-vector is like indexing

"Even the smallest person can change the course of the future."

$v_{1,1}$	$v_{1,2}$	…	$v_{1,V-1}$	$v_{1,V}$
$v_{2,1}$	$v_{2,2}$	…	$v_{2,V-1}$	$v_{2,V}$
$v_{3,1}$	$v_{3,2}$	…	$v_{3,V-1}$	$v_{3,V}$
$v_{4,1}$	$v_{4,2}$	…	$v_{4,V-1}$	$v_{4,V}$
$v_{5,1}$	$v_{5,2}$	…	$v_{5,V-1}$	$v_{5,V}$
…	…	…	…	…
$v_{d,1}$	$v_{d,2}$	…	$v_{d,V-1}$	$v_{d,V}$

$\boldsymbol{V} \in \mathbb{R}^{d \times V}$

x_1
x_2
…
x_{V-1}
x_V

$\boldsymbol{x_t} \in \{0,1\}^{V \times 1}$

$h_1 = v_{1,1}x_1 + v_{1,2}x_2 + \cdots + v_{1,V-1}x_{V-1} + v_{1,V}x_V$
$h_2 = v_{2,1}x_1 + v_{2,2}x_2 + \cdots + v_{2,V-1}x_{V-1} + v_{2,V}x_V$
…
$h_{d-1} = v_{d-1,1}x_1 + v_{d-1,2}x_2 + \cdots + v_{d-1,V-1}x_{V-1} + v_{d-1,V}x_V$
$h_d = v_{d,1}x_1 + v_{d,2}x_2 + \cdots + v_{d,V-1}x_{V-1} + v_{d,V}x_V$

$\boldsymbol{h_t} \in \mathbb{R}^{d \times 1}$

$$\boldsymbol{h_t = V x_t}$$

Setting up the Neural Network

Dot product with one-hot-vector is like indexing

"Even the smallest person can change the course of the future."

$v_{1,1}$	$v_{1,2}$	…	$v_{1,\mathcal{V}-1}$	$v_{1,\mathcal{V}}$
$v_{2,1}$	$v_{2,2}$	…	$v_{2,\mathcal{V}-1}$	$v_{2,\mathcal{V}}$
$v_{3,1}$	$v_{3,2}$	…	$v_{3,\mathcal{V}-1}$	$v_{3,\mathcal{V}}$
$v_{4,1}$	$v_{4,2}$	…	$v_{4,\mathcal{V}-1}$	$v_{4,\mathcal{V}}$
$v_{5,1}$	$v_{5,2}$	…	$v_{5,\mathcal{V}-1}$	$v_{5,\mathcal{V}}$
…	…	…	…	…
$v_{d,1}$	$v_{d,2}$	…	$v_{d,\mathcal{V}-1}$	$v_{d,\mathcal{V}}$

$\boldsymbol{V} \in \mathbb{R}^{d \times \mathcal{V}}$

$x_1 = 0$
$x_2 = 1$
…
$x_{\mathcal{V}-1} = 0$
$x_{\mathcal{V}} = 0$

$\boldsymbol{x_t} \in \{0,1\}^{\mathcal{V} \times 1}$

$h_1 =$	$0 +$	$v_{1,2}x_2 + \cdots +$	0
$h_2 =$	$0 +$	$v_{2,2}x_2 + \cdots +$	0
		…	
$h_{d-1} =$	$0 +$	$v_{d-1,2}x_2 + \cdots +$	0
$h_d =$	$0 +$	$v_{d,2}x_2 + \cdots +$	0

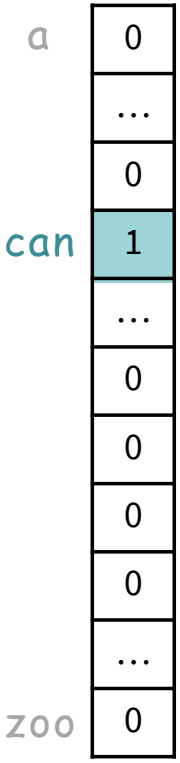
$\boldsymbol{h_t} \in \mathbb{R}^{d \times 1}$

$$\boldsymbol{h_t = V x_t}$$

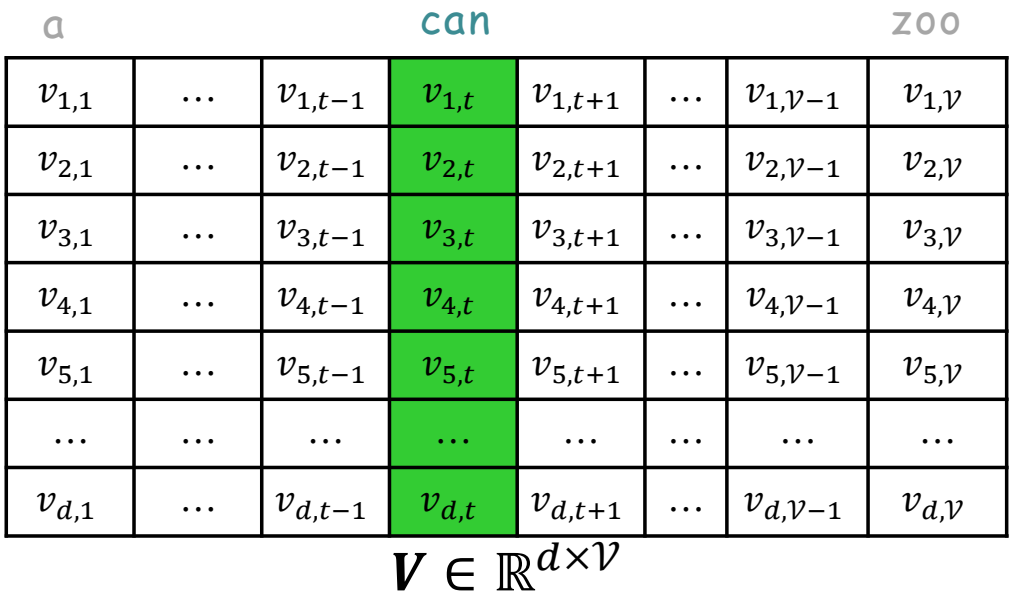
Word embeddings

After training, weight matrix V gives embeddings of words in the vocabulary

"Even the smallest person can change the course of the future."



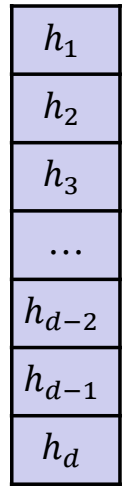
$x_t \in \{0,1\}^{\mathcal{V}\times 1}$



$\boldsymbol{V} \in \mathbb{R}^{d\times\mathcal{V}}$

Word vector of can of embedding dimension d

$\boldsymbol{h_t} \in \mathbb{R}^{d\times 1}$



$h_t = \boldsymbol{V} x_t$

Neural Network Training under Skip-Gram Model

"Even the smallest person can change the course of the future."

Context window of size 2

- **Assume conditional independence of context words given center word**

$$P(\text{smallest, person, change, the} \mid \text{can})$$
$$= P(\text{smallest} \mid \text{can}) \cdot P(\text{person} \mid \text{can}) \cdot P(\text{change} \mid \text{can}) \cdot P(\text{the} \mid \text{can})$$

- **Model conditional probability of observing a context word given center word by softmax operation:**

$$P(w_o \mid w_c) = \frac{\exp(\boldsymbol{u}_o^\top \boldsymbol{v}_c)}{\sum_{i \in \mathcal{V}} \exp(\boldsymbol{u}_i^\top \boldsymbol{v}_c)}$$

With context word w_o and center word w_c
and corresponding embedding vectors $\boldsymbol{u}_o, \boldsymbol{v}_c \in \mathbb{R}^d$

Neural Network Training under Skip-Gram Model

Loss minimization

Recall softmax operation:

$$P(w_o \mid w_c) = \frac{\exp(\boldsymbol{u}_o^\top \boldsymbol{v}_c)}{\sum_{i \in \mathcal{V}} \exp(\boldsymbol{u}_i^\top \boldsymbol{v}_c)}$$

- **Likelihood function**

$$\prod_{t=1}^{N} \prod_{-m \le j \le m, j \ne 0} P\left(w_o^{(t+j)} \mid w_C^{(t)}\right)$$

- **Binary cross-entropy loss (aka log-loss)**

$$\min -\sum_{t=1}^{N} \sum_{-m \le j \le m, j \ne 0} \log\left(P\left(w_o^{(t+j)} \mid w_c^{(t)}\right)\right)$$

  - Minimize over the free parameters in our network
  - That is the weights in the weight matrices U and V
  - Which, as shown above, will be our word embeddings

Neural Network Training under Skip-Gram Model

Problems with high-dimensional softmax

Recall softmax operation:

$$P(w_o \mid w_c) = \frac{\exp(\boldsymbol{u}_o^\top \boldsymbol{v}_c)}{\sum_{i \in \mathcal{V}} \exp(\boldsymbol{u}_i^\top \boldsymbol{v}_c)}$$

- **Minimization problem**

$$\min -\sum_{t=1}^{N} \sum_{-m \le j \le m, j \neq 0} \log\left(P\left(w_o^{(t+j)} \mid w_c^{(t)}\right)\right)$$

- **Say we train using SGD and update weights after seeing one example**

$$\min -\log\left(P(w_o \mid w_c)\right) = \log\left(\frac{\exp(\boldsymbol{u}_o^\top \boldsymbol{v}_c)}{\sum_{i \in \mathcal{V}} \exp(\boldsymbol{u}_i^\top \boldsymbol{v}_c)}\right) = \boldsymbol{u}_o^\top \boldsymbol{v}_c - \log\left(\sum_{i \in \mathcal{V}} \exp(\boldsymbol{u}_i^\top \boldsymbol{v}_c)\right)$$

- **Gradient update**

$$\frac{\partial \log\left(P(w_o \mid w_c)\right)}{\partial \boldsymbol{v}_c} = \boldsymbol{u}_o - \frac{\sum_{j \in \mathcal{V}} \exp(\boldsymbol{u}_j^\top \boldsymbol{v}_c) \boldsymbol{u}_j}{\sum_{i \in \mathcal{V}} \exp(\boldsymbol{u}_i^\top \boldsymbol{v}_c)}$$

Neural Network Training under Skip-Gram Model

Problems with high-dimensional softmax

■ **Gradient update (cont.)**

$$\frac{\partial \log\left(P(w_o|w_c)\right)}{\partial \boldsymbol{v}_c} = \boldsymbol{u}_o - \frac{\sum_{j\in\mathcal{V}} \exp\left(\boldsymbol{u}_j^\top \boldsymbol{v}_c\right)\boldsymbol{u}_j}{\sum_{i\in\mathcal{V}} \exp\left(\boldsymbol{u}_i^\top \boldsymbol{v}_c\right)}$$

$$= \boldsymbol{u}_o - \sum_{j\in\mathcal{V}} \left(\frac{\exp\left(\boldsymbol{u}_j^\top \boldsymbol{v}_c\right)}{\sum_{i\in\mathcal{V}} \exp\left(\boldsymbol{u}_i^\top \boldsymbol{v}_c\right)}\right)\boldsymbol{u}_j$$

$$= \boldsymbol{u}_o - \sum_{j\in\mathcal{V}} p\left(w_j|w_c\right)\boldsymbol{u}_j$$

- Gradient update requires the conditional probabilities of all words in the vocabulary with w_c as the center word
- Gradients for the other (center) word vectors can be obtained in the same way

Word2Vec Adjustments

Network training is infeasible on larger corpora

- **Large corpus essential to learn word meaning**
- **High-dimensional softmax is the bottleneck**
  - $P(w_o \mid w_c) = \frac{\exp(\boldsymbol{u}_o^\top \boldsymbol{v}_c)}{\sum_{i \in \mathcal{V}} \exp(\boldsymbol{u}_i^\top \boldsymbol{v}_c)}$
  - Gradient update requires the conditional probabilities of all words in the vocabulary $\mathcal{V}$
  - Size of the vocabulary is very large
- **Hierarchical softmax**
  - Accelerate computations by calculating conditional probabilities as muti-layer binary tree
  - Proposed by Morin & Bengio (2005)

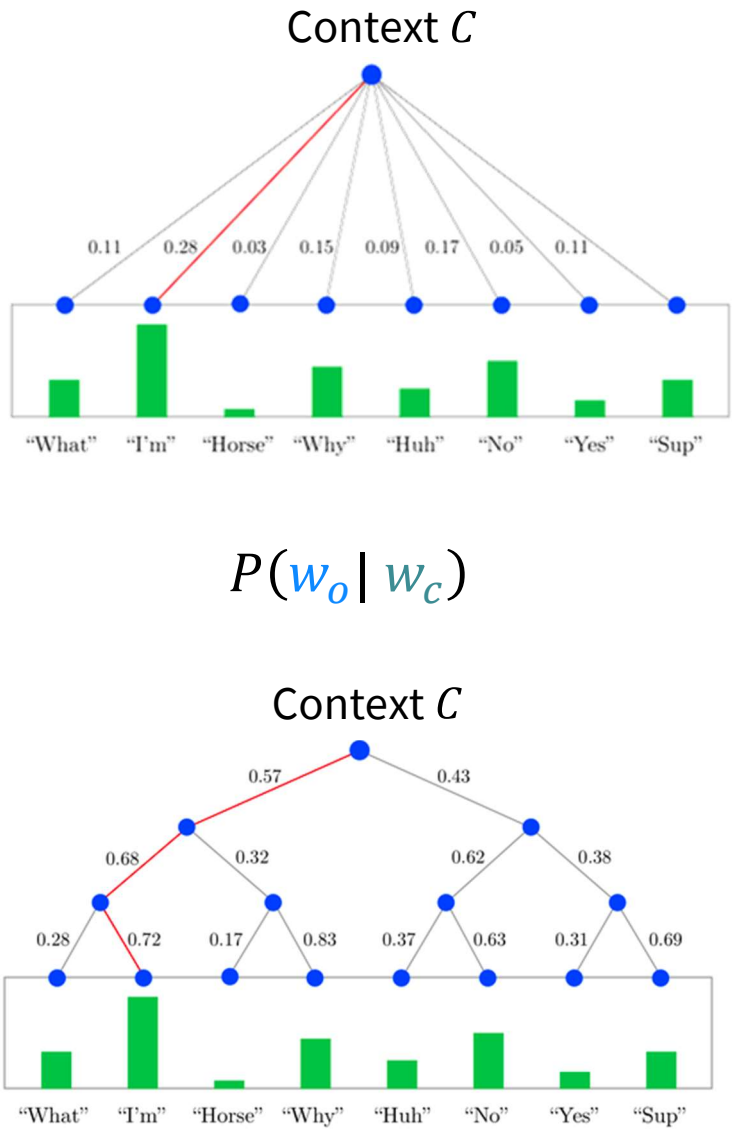


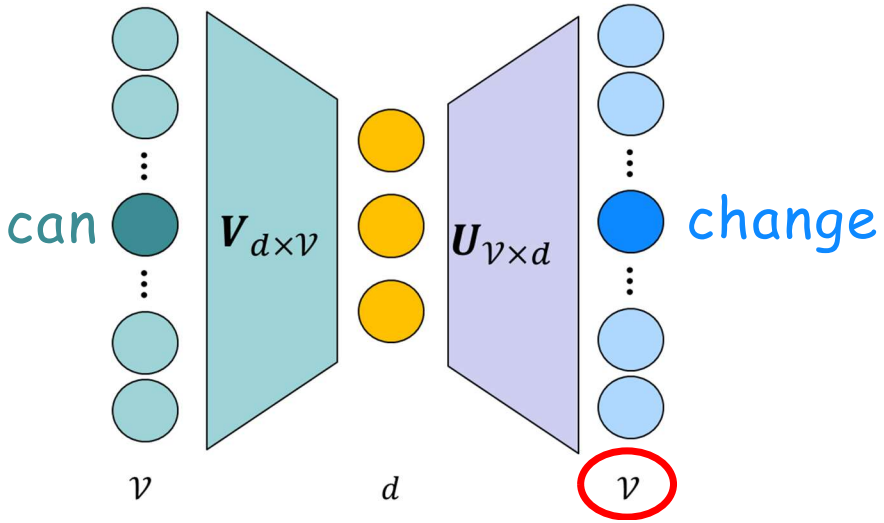
Image: https://www.quora.com/What-is-hierarchical-softmax

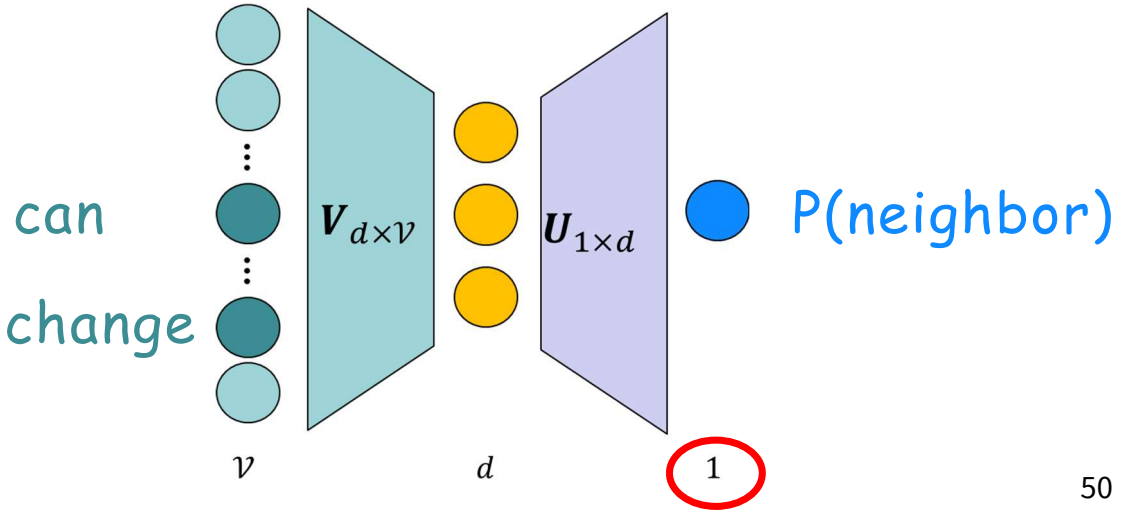
Word2Vec Adjustments

Network training is infeasible on larger corpora

- **Negative sampling**
  - Original W2K task: predict conditional probability of neighboring words
  - Change task to binary classification: predict if two words are neighbors

"Even the smallest person can change the course of the future."





FastText

Exploit morphology information through subword embedding

- **Words have an internal structure, which word2vec does not take into account**
  - Words appear in different forms in different contexts, where forms change with suffixes
  - Examples : dog & dogs is analog to cat & cats; boy & boyfriend is analog to girl & girlfriend, …
  - No recognition of internal word structure in word2vec
- **Fasttext solution: move from word to character-level embedding**
  - Each word is represented as a bag of character n-grams
  - Example for word matter with $n = 3$: <ma, mat, att, tte, ter, er>
  - Better suited for rare words and able to handle new words not part of the vocabulary
- **Target word vector in skip-gram model**
  - Where $\mathbf{z}_g$ is the vector of subword g in the dictionary
  - And the training of (sub)word vectors is the same as in word2vec

$$v_c = \sum_{g \in G_w} \mathbf{z}_g$$

Bojanowski et al. (2017) ; Joulin et al. (2016) ; Grave et al. (2018)

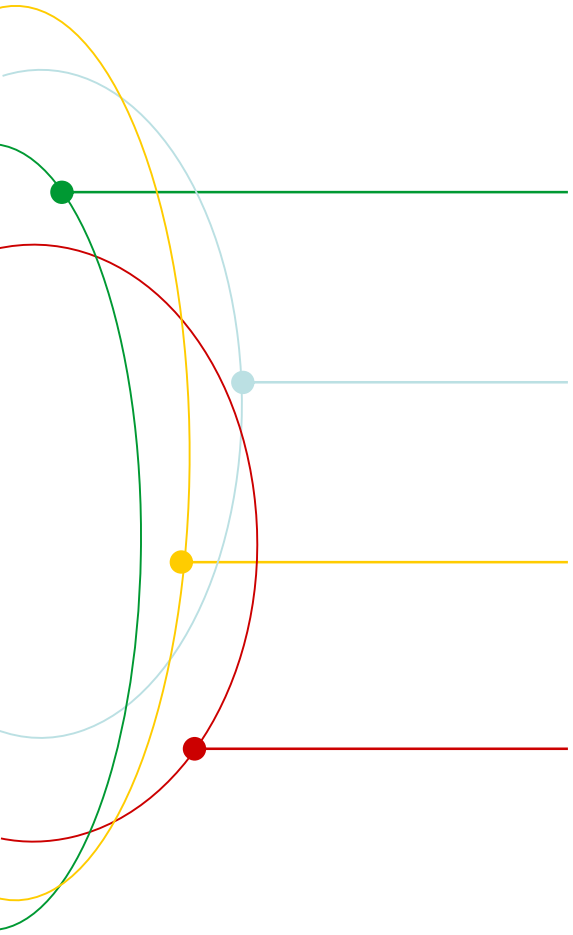
GloVe: Global Vectors for Word Representation

Pennington et al. (2014)

- **Objectives**
  - Create word vectors that capture meaning in a vector space (like word-to-vec)
  - Take advantage of global count statistics instead of only local information
- **GloVe versus Word-to-Vec (W2V)**
  - W2V creates training data by streaming sentences
  - GloVe learns based on a word co-occurrence matrix
  - GloVe word vectors predict co-occurrence ratios
- **Although approach the problem from a different angle, word-to-vec and GloVe are mathematically equivalent**

Summary

Summary



Learning goals

- Bag-of-words model and its shortcomings
- Word embeddings

Findings

- High-dimensional, sparse count-based representations ignore word order and context
- Word embeddings
  - Represent words as low dim. dense vectors
  - Capture semantics and meaning of words
  - Self-supervised training using skip-gram and CBOW
- Word2Vec algorithm and cousins

What next

- Advanced NLP approaches
- Recurrent networks and transformers

Literature

- Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research, 3*, 993-1022.
- Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *TACL,* 5, 135-146.
- Firth, J.R. (1957). A synopsis of linguistic theory 1930-1955. In *Studies in Linguistic Analysis*, pp. 1-32. Oxford: Philological Society.]
- Grave, E., Bojanowski, P., Gupta, P., Joulin, A., & Mikolov, T. (2018). Learning Word Vectors for 157 Languages. *CoRR,* arXiv:1802.06893v2.
- Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). A bag of tricks for efficient text classification. *CoRR,* arXiv:1607.01759v3.
- Q. V. Le, T. Mikolov. (2014). *Distributed Representations of Sentences and Documents*. Paper presented at the Proceedings of the 31st International Conference on Machine Learning (ICML'14), Beijung China. http://jmlr.org/proceedings/papers/v32/le14.html
- Liu, B. (2013). *Web Data Mining* (2nd ed.). Heidelberg: Springer.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). *Distributed Representations of Words and Phrases and their Compositionality*. NIPS 2013.
- Morin, F., & Bengio, Y. (2005). Hierarchical Probabilistic Neural Network Language Model Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research. https://proceedings.mlr.press/r5/morin05a.html
- Pennington, J., Socher, R., & Manning, C. (2014). *Glove: Global Vectors for Word Representation*. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, pp. 1532–1543.
- Rong, X. (2016). word2vec Parameter Learning Explained. Arxiv pre-print arXiv:1411.2738v4.

Thank you for your attention!

Stefan Lessmann

Chair of Information Systems
School of Business and Economics
Humboldt-University of Berlin, Germany

Tel. +49.30.2093.5742
Fax. +49.30.2093.5741

stefan.lessmann@hu-berlin.de
http://bit.ly/hu-wi

www.hu-berlin.de

Photo: Heike Zappe