Deep Learning for NLP

Universität Bielefeld

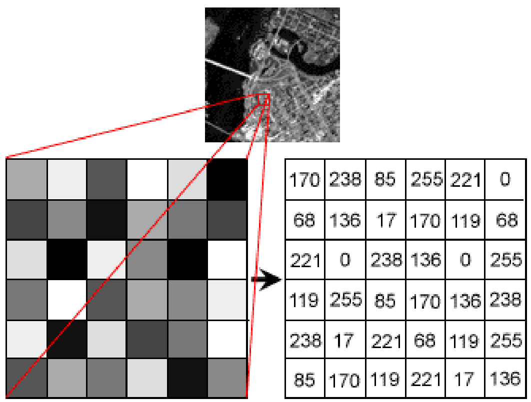
Lecture 4 – Word Embeddings 1: Word2Vec

Dr. Steffen Eger
steffen.eger@uni-bielefeld.de

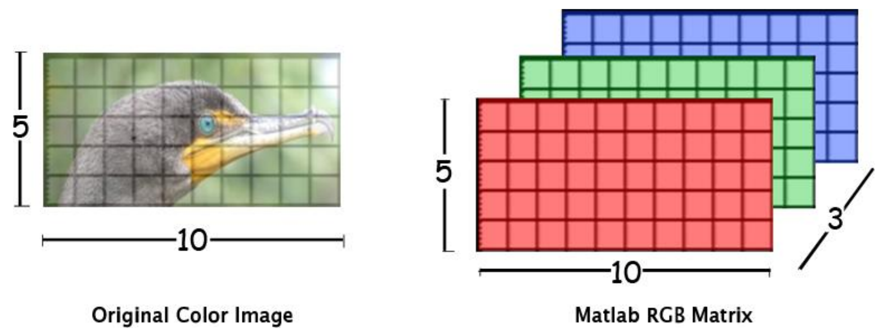
CITEC NLLG

Previous lectures

- How do we build/train neural networks?
- Today and next two weeks: How do we represent the input?
- Representing images:



http://hosting.soonet.ca/eliris/remotesensing/LectureImages/pixel.gif



http://www.comp.leeds.ac.uk/vision/vision_matlab_tutorial/images/section2/MatlRGB.jpg

- How can we represent words?

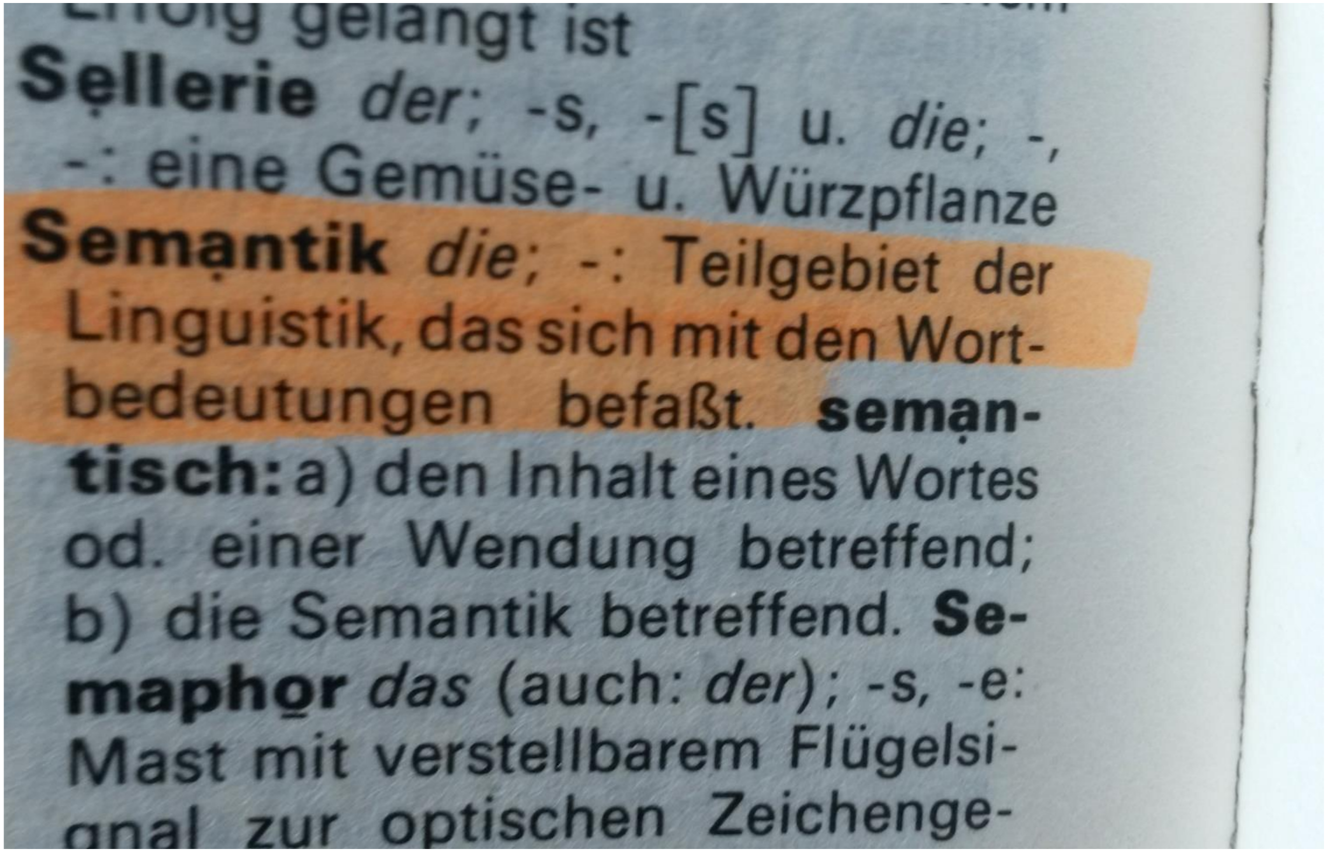
Outline

Word meaning

How can we represent words?

Universität Bielefeld

- As a dictionary entry

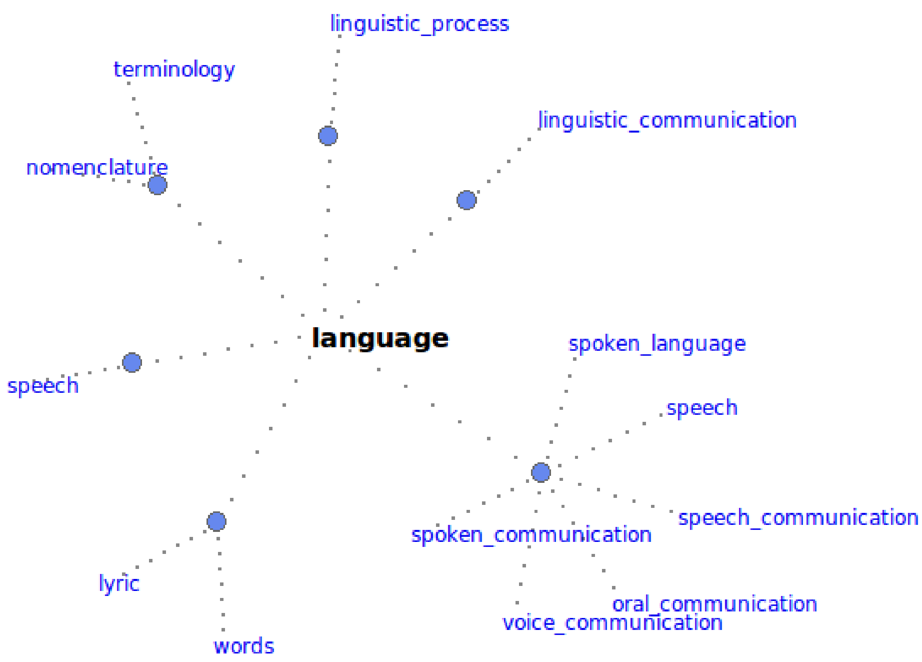


CITEC NLLG

Taxonomy of words

Universität Bielefeld

- Represent words by their **relations to other words**



Picture from: http://kylescholz.com/projects/wordnet/, based on representation from WordNet: https://wordnet.princeton.edu

Word vectors

Universität Bielefeld

- “One-hot” vector, sparse representation

der	die	und	in	....	für
1	0	0	0		0
0	1	0	0		0
0	0	1	0		0
0	0	0	1		0
0	0	0	0		0
0	0	0	0		0
0	0	0	0		0
0	0	0	0		0
0	0	0	0		0
0	0	0	0		1
...	...	...	...		...

- Dimensionality of vector equals size of vocabulary

CITEC NLLG

Word vectors

Universität Bielefeld

- “One-hot” vector, sparse representation

der	die	und	in	...	für
1	0	0	0		0
0	1	0	0		0
0	0	1	0		0
0	0	0	1		0
0	0	0	0		0
0	0	0	0		0
0	0	0	0		0
0	0	0	0		0
0	0	0	0		0
0	0	0	0		1
...	...	...	...		...

Problem: relations between words are not represented

Word meaning can be inferred from context

- Famous example by McDonald and Ramscar (2001):

    1. He filled the **wampimuk**, passed it around and we all drank some.

Word meaning can be inferred from context

Universität Bielefeld

- Famous example by McDonald and Ramscar (2001):

  1. He filled the **wampimuk**, passed it around and we all drank some.

     jar

     cup

     glass

     goblet

     …

Word meaning can be inferred from context

Universität Bielefeld

- Famous example by McDonald and Ramscar (2001):

1. He filled the wampimuk, passed it around and we all drank some.

   jar

   cup

   glass

   goblet

   …

2. We found a little hairy **wampimuk** sleeping behind the tree.

CITEC NLLG

Word meaning can be inferred from context

Universität Bielefeld

- Famous example by McDonald and Ramscar (2001):

1. He filled the wampimuk, passed it around and we all drank some.

   jar
   cup
   glass
   goblet
   ...

2. We found a little hairy **wampimuk** sleeping behind the tree.

   cat
   bear
   racoon
   mole
   ...

Distributional hypothesis

Universität Bielefeld

- Firth (1957): ***"You shall know a word by the company it keeps."***

Outline

Computational semantics: Count models

How can we model the distributional hypothesis?

Universität Bielefeld

- By calculating co-occurrence counts
  - capture in which contexts a word appears
- Context is modeled using a window over the words

- Consider the following example (from Richard Socher's lecture):
- Corpus
  - *I like deep learning .*
  - *I like NLP .*
  - *I enjoy flying .*
- Window size = 1, left and right neighbor
  - In real tasks, window size is usually bigger (5-10)

CITEC NLLG

How can we model the distributional hypothesis?

- By calculating co-occurrence counts
  - capture in which contexts a word appears
- Context is modeled using a window over the words

Such models have been called „count models“ in the literature
See: Baroni et al. Don‘t count predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In: ACL 2014

- Consider the following example by Rich
- Corpus
  - *I like deep learning .*
  - *I like NLP .*
  - *I enjoy flying .*
- Window size = 1, left and right neighbor
  - In real tasks, window size is usually bigger (5-10)

Co-occurrence matrix

Universität Bielefeld

- Example by Richard Socher:
  - *I like deep learning .*
  - *I like NLP .*
  - *I enjoy flying .*

Window size = 1

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Co-occurrence matrix

Universität Bielefeld

- Example by Richard Socher:
  - *I like deep learning .*
  - *I like NLP .*
  - *I enjoy flying .*

Window size = 1

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Co-occurrence matrix

Universität Bielefeld

- Example by Richard Socher:
  - *I like deep learning .*
  - *I like NLP .*
  - *I enjoy flying .*

Window size = 1

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Co-occurrence matrix

Universität Bielefeld

- Example by Richard Socher:
  - *I like deep learning .*
  - *I like NLP .*
  - *I enjoy flying .*

Window size = 1

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

CITEC NLLG

Co-occurrence counts

- Assumption: If we collect co-occurrence counts over thousands of sentences, the vectors for “enjoy” and “like” will have similar vector representations.

Co-occurrence counts

- Assumption: If we collect co-occurrence counts over thousands of sentences, the vectors for “enjoy” and “like” will have similar vector representations.

- Problem:
  - Vectors become very large with real data

    → We need to apply dimensionality reduction

Outline

Universität Bielefeld

Computational semantics: NN models

CITEC NLLG

Background idea: language models

- Based on the concept of **language modeling**:
- Common problem in NLP, popular application is auto-completion
  - Given a sequence of words, predict the following word
  - *The same procedure as every* ______

- Idea:

  (Classical) Language modeling is too restrictive because it only considers the left context. What about the right context?

word2vec

Universität Bielefeld

- Most popular toolkit for training word representations:

word2vec

https://code.google.com/archive/p/word2vec/

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean:
Distributed Representations of Words and Phrases and their Compositionality
In Proceedings of NIPS, 2013.

- Two different (language modeling) auxiliary tasks: CBOW and Skip-gram

Auxiliary Tasks

Universität Bielefeld

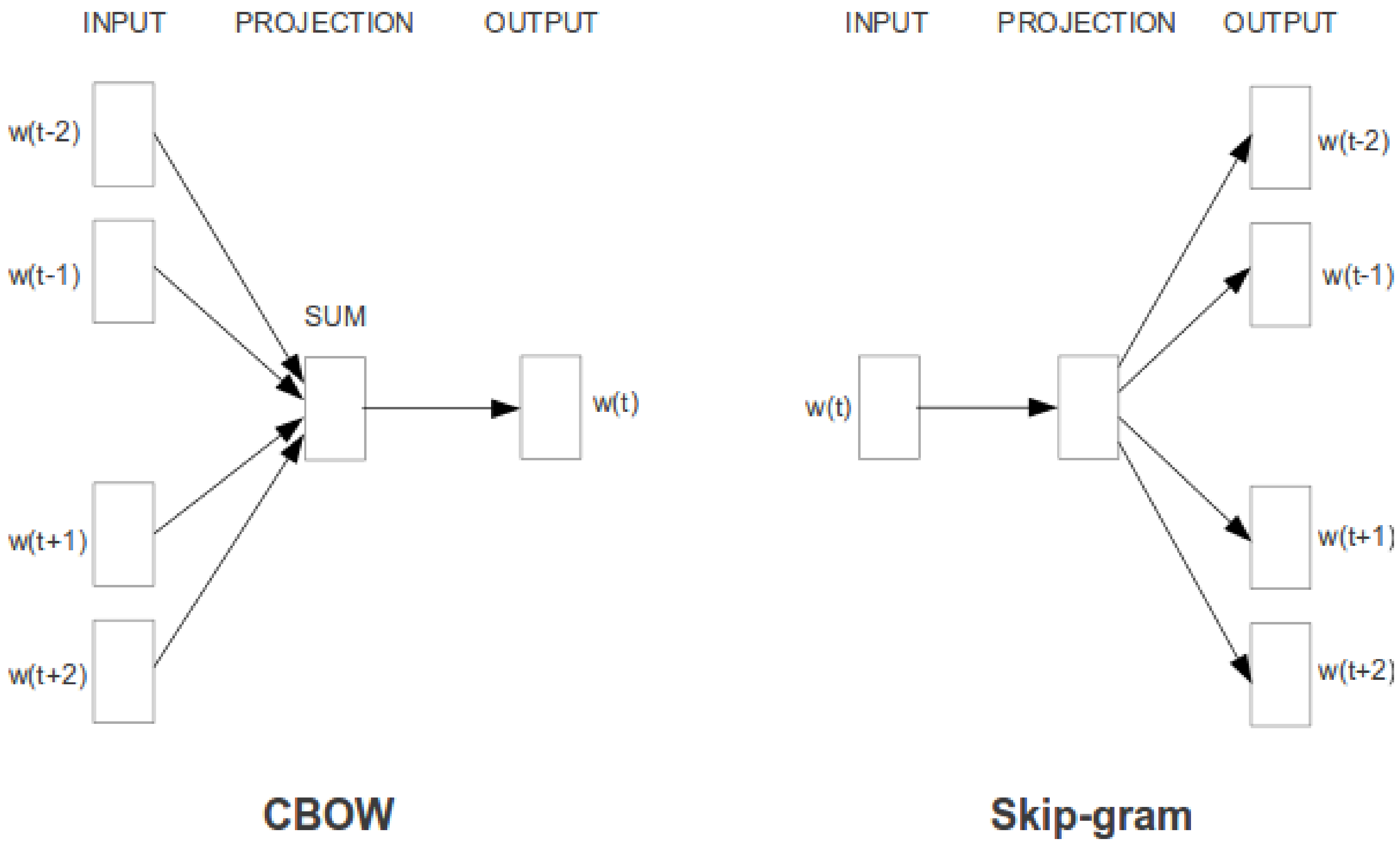
- CBOW: Given a context, predict the missing word
  - *same procedure ____ every year*
  - *as long ___ you sing*
  - *please stay ___ you are*

- Skip-gram: given a word, predict the context words
  - ____ _______ *as* ______ _______
  - If window size is two, we aim to predict:

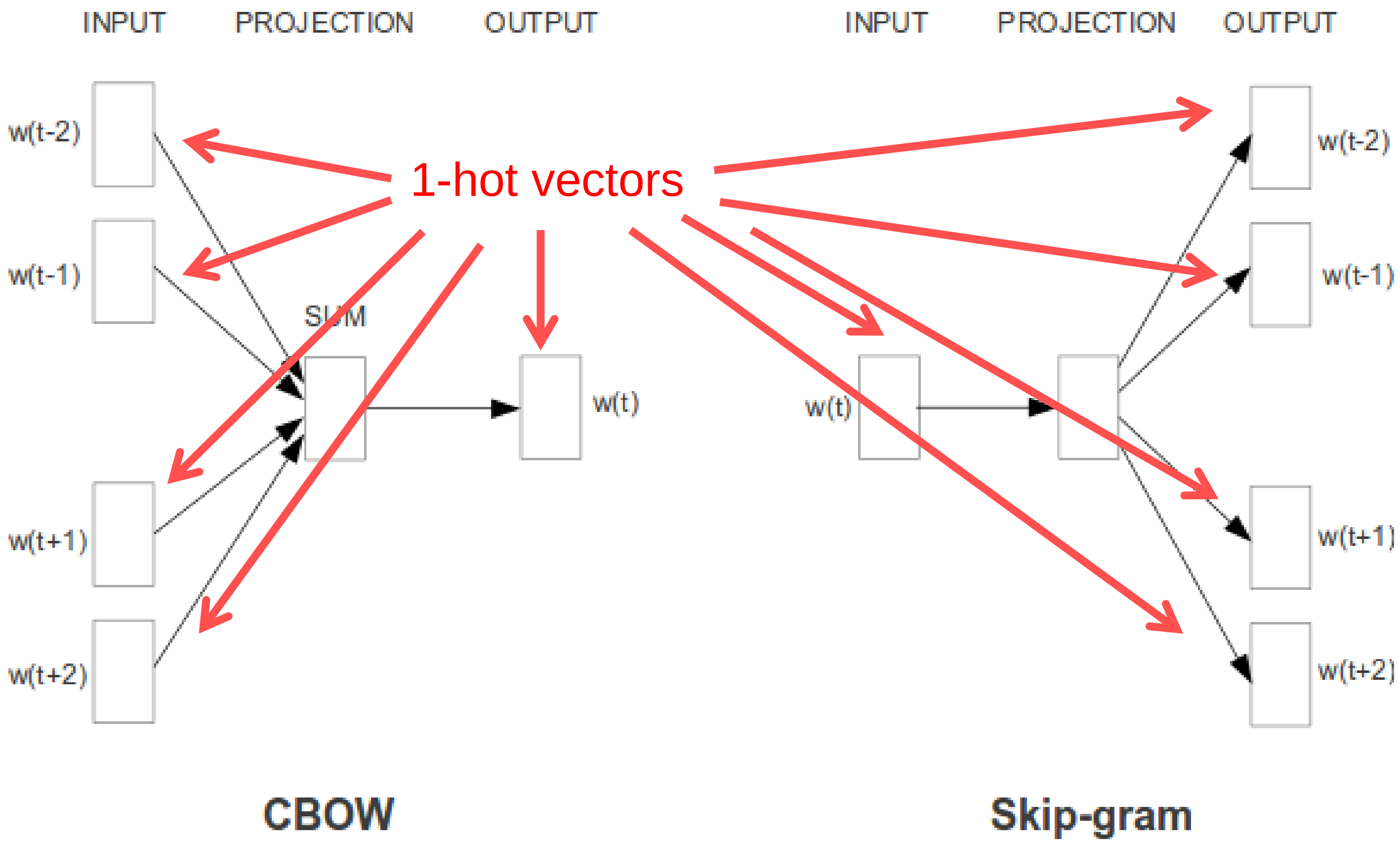
    (w,c_{-2}), (w,c_{-1}), (w,c_{1}) and (w,c_{2})

CITEC NLLG

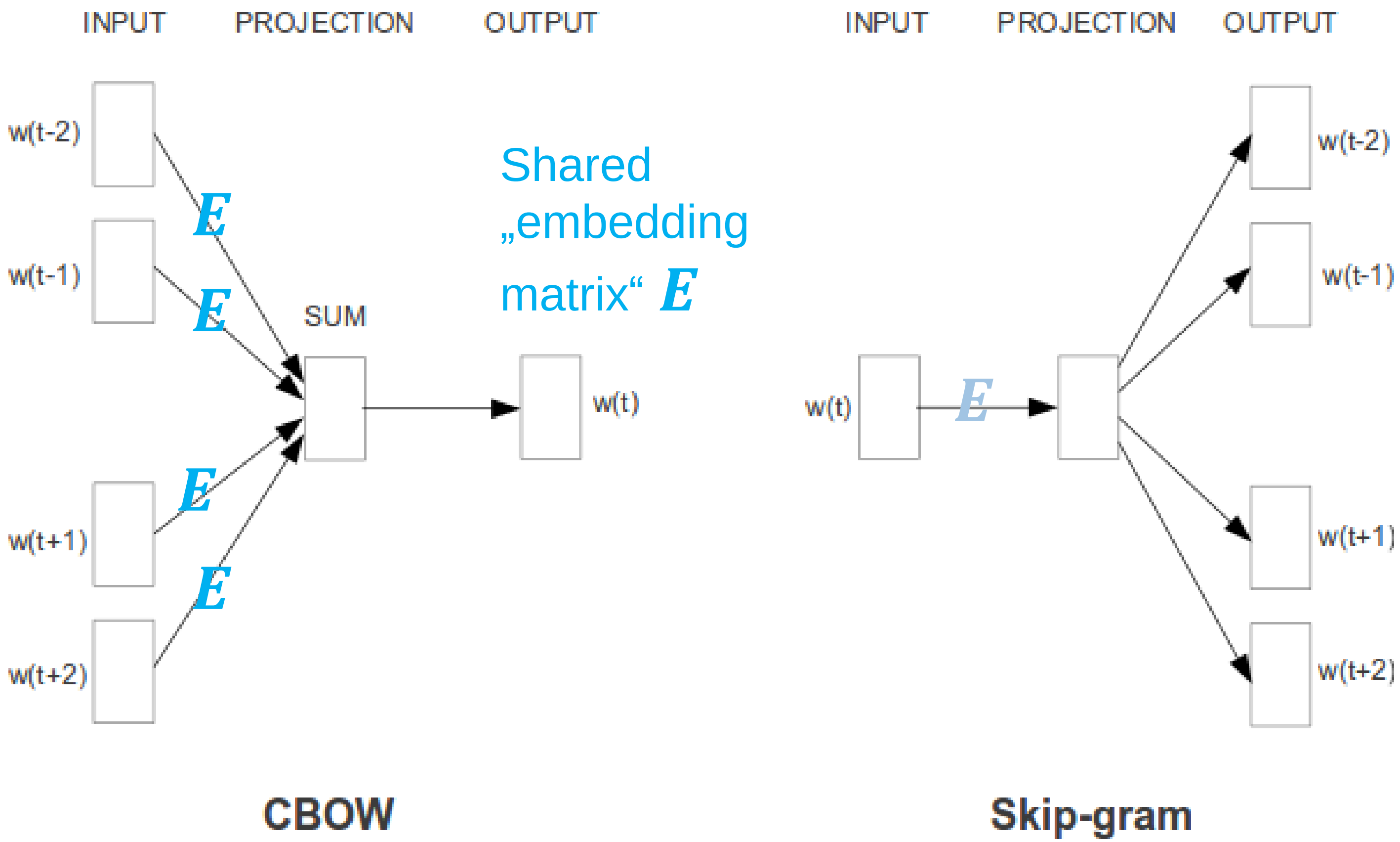
CBOW vs Skip-gram



CBOW vs Skip-gram



CBOW vs Skip-gram

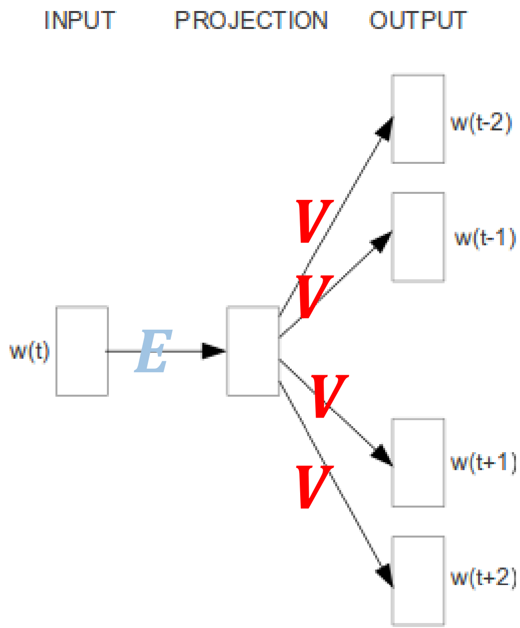


The Skip-gram model: Theory

Universität Bielefeld

We'll take a closer look at the Skip-Gram model

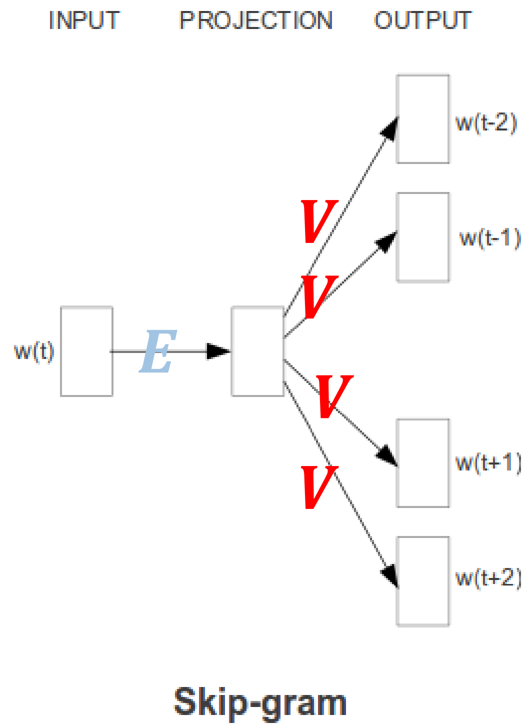
- $\boldsymbol{E}$ has dimension $N \times d$
- N is number of words in the vocabulary
- d is the embedding dimension
- $\boldsymbol{V}$ has dimension $d \times N$



Skip-gram

The Skip-gram model: Theory

- We'll take a closer look at the Skip-Gram model

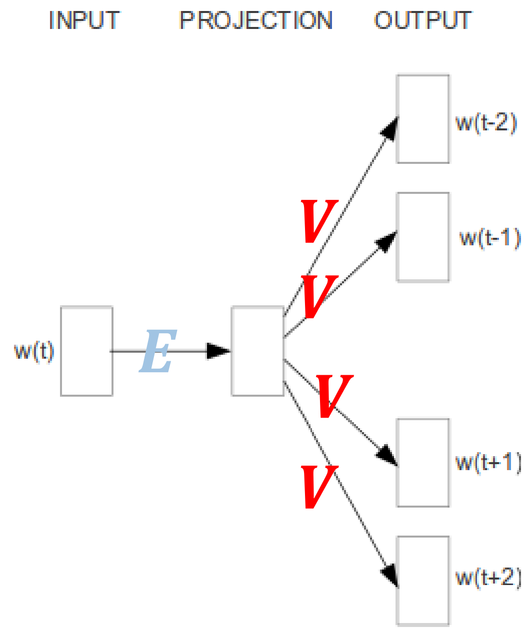


How can we predict different words when V is always the same?

The Skip-gram model: Theory

Universität Bielefeld

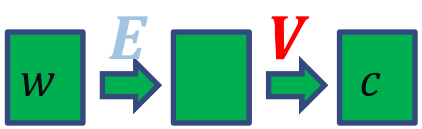
We'll take a closer look at the Skip-Gram model



Skip-gram

How can we predict different words when V is always the same?

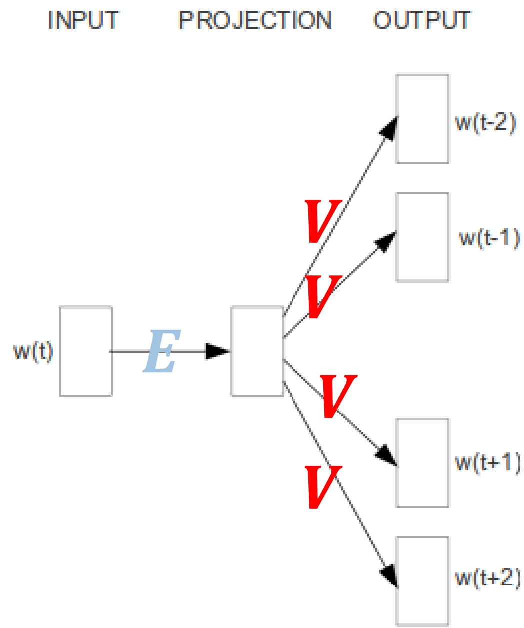
Turns out the original model is actually this:



The Skip-gram model: Theory

We'll take a closer look at the Skip-Gram model

- $\boldsymbol{e} = \boldsymbol{e}(\boldsymbol{w}) = \boldsymbol{wE}$ has dimension $1 \times d$: It's the *embedding* of word w
- The activation of the projection layer is *linear (= identity: f(x)=x)*
- $\boldsymbol{eV}$ has dimension $1 \times N$
- $\boldsymbol{V} = [\boldsymbol{v}_1 \cdots \boldsymbol{v}_N]$: each $\boldsymbol{v}_i$ can be seen as an(other) embedding of a vocab. word



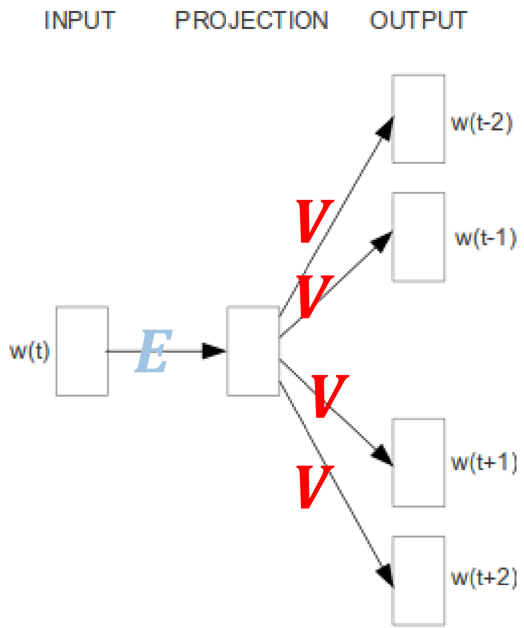
Skip-gram

The Skip-gram model: Theory

Universität Bielefeld

We‘ll take a closer look at the Skip-Gram model

- $\boldsymbol{eV} = [\boldsymbol{ev}_1, \ldots, \boldsymbol{ev}_N]$ has dimension $1 \times N$
- $\boldsymbol{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N]$: each $\boldsymbol{v}_i$ can be seen as an(other) embedding of a vocab. word
- The output layer has softmax activation function
- softmax($\boldsymbol{eV}$)= $[\exp(\boldsymbol{ev}_1), \ldots, \exp(\boldsymbol{ev}_N)]\,/Z$, where Z is normalizer



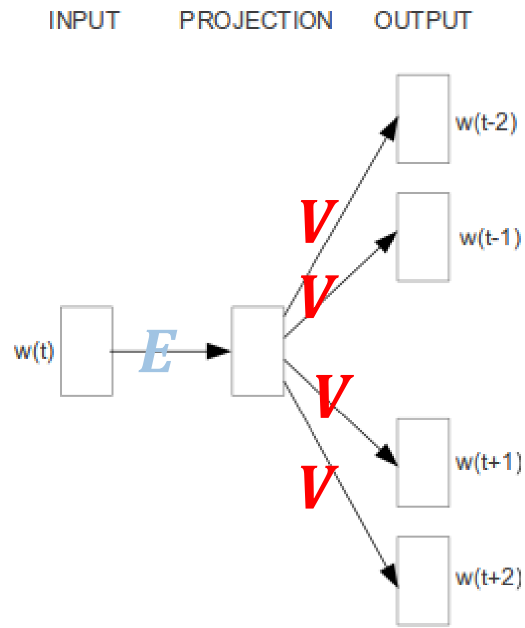
Skip-gram

The Skip-gram model: Theory

Universität Bielefeld

We'll take a closer look at the Skip-Gram model

- Could just run this model with SGD
- With methods we learned
- After training, we're interested in the $\boldsymbol{E}$ matrix, which holds the word embeddings
- **However, the practical implementation is different from this (see below)**



Skip-gram

CITEC NLLG

DISCUSS

Discuss two (or more) limitations of the Word2Vec model

The Skip-gram model: Illustration

- Preparation:

  - Download a lot of (**unlabeled**) data, e.g. all the poems of W.S.

All the world’s a stage and all the men and women merely players. They totter …

  - Tokenize it

All the world ’s a stage and all the men and women merely players . They totter …

  - For word2vec: Define either one of two **auxiliary** tasks: predict middle words or predict contexts

For skip-gram:

All the world 's a stage and all the men and women merely players . They totter …

- Training data (maybe this is our first batch):
  - x=world   t=the
  - x=world   t=All
  - x=world   t=‘s
  - x=world   t=a

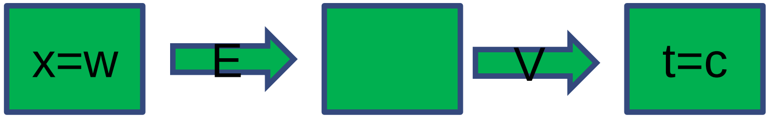
  - Of course, the 1-hot vectors of this

For skip-gram:

All the world 's a stage and all the men and women merely players . They totter …

- Training data (maybe this is our first batch):
  - x=world t=the
  - x=world t=All
  - x=world t='s
  - x=world t=a

Feed in to the network; update params



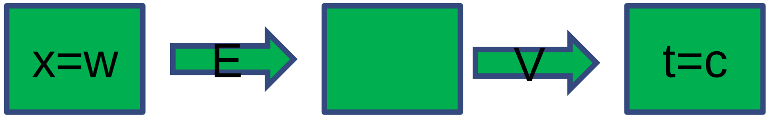
  - Of course, the 1-hot vectors of this

For skip-gram:

All the world 's a stage and all the men and women merely players . They totter …

- Training data (maybe this is our first batch):
  - x=world   t=the
  - x=world   t=All
  - x=world   t='s
  - x=world   t=a

Feed in to the network; update params



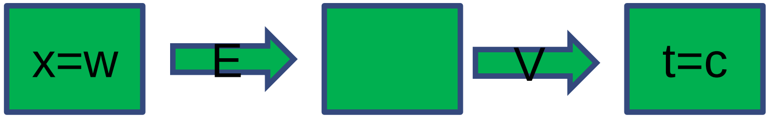
  - Of course, the 1-hot vectors of this
- Notation: We write (x,t) or (w,c) for a training data sample

For skip-gram:

All the world 's a stage and all the men and women merely players .
They totter …

- Training data (maybe this is our second batch):
  - x=‘s   t=world
  - x=‘s   t=the
  - x=‘s   t=a
  - x=‘s   t=stage

Feed in to the network; update params



  - Of course, the 1-hot vectors of this
- Notation: We write (x,t) or (w,c) for a training data sample

For skip-gram:

Universität Bielefeld

All the world 's a stage and all the men and women merely players . They totter …

Context window size is a hyperparam.

- Training data (maybe this is our second batch):
  - x='s   t=world
  - x='s   t=a

Feed in to the network; update params

  - Of course, the 1-hot vectors of this
- Notation: We write (x,t) or (w,c) for a training data sample

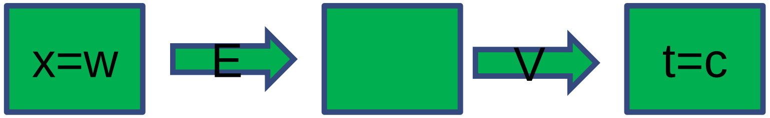
For skip-gram:

All the world 's a stage and all the men and women merely players . They totter …

Context window size is a hyperparam.

- Training data (maybe this is our second batch):
  - x='s t=All
  - x='s t=the
  - x='s t=world
  - x='s t=a
  - x='s, t=stage, x='s, t=and

Feed in to the network; update params



  - Of course, the 1-hot vectors of this
- Notation: We write (x,t) or (w,c) for a training data sample

Quizz

What if the window size is arbitrarily large in Skip-Gram?

A: this can only work when embedding size is huge
B: quality of representations decreases
C: quality of representations increases
D: all words will get same embeddings

Skip-gram: practical realization

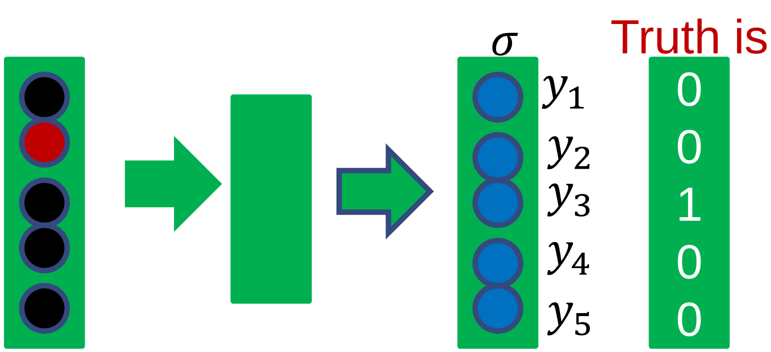
Problem:

- Computing the softmax in the output layer is very costly, particulary when the size of the vocabulary is large
- Because we have to normalize

Skip-gram: practical realization

Idea: replace softmax by sigmoid activation function in the output layer

- No normalization → cheaper
- So, at each step in the optimization:
  - input vector is a 1-hot vector of the center word w
  - Truth is a 1-hot vector of target/context word c
  - As loss, we choose cross-entropy loss: $-\sum_i t_i \log y_i$
  - $y_i = \sigma(\boldsymbol{e}(\boldsymbol{w}) \cdot \boldsymbol{v}_i)$



Skip-gram: practical realization

Universität Bielefeld

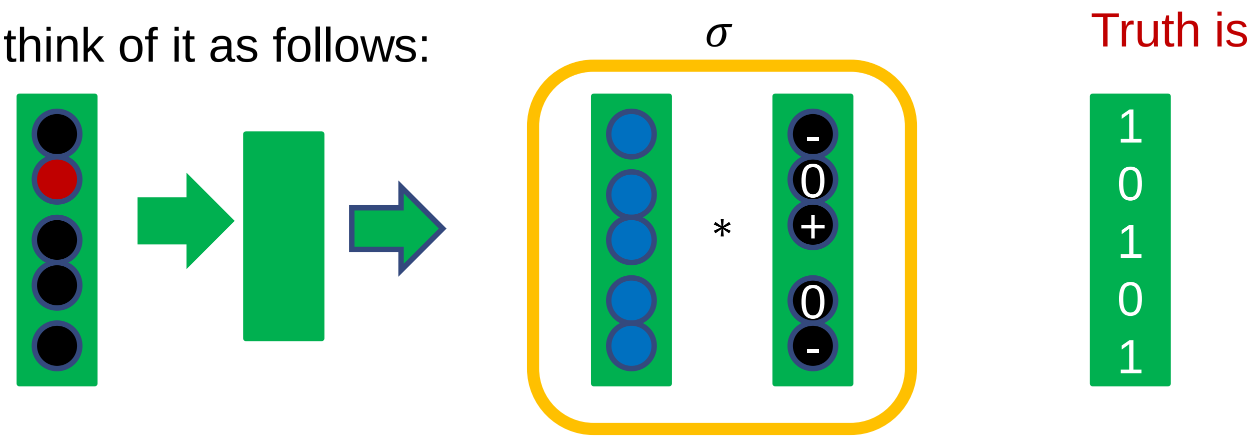
- **But now a different problem pops up**:
  - What if our model learns that $\boldsymbol{E}$ and $\boldsymbol{V}$ are matrices with huge entries?
  - Then $y_i = 1$ for all output units and our loss becomes 0, which is the optimum
  - Note that this would not happen with softmax, because $y_i = 1/N$ in this case

- **Solution**:
  - We choose random words (*negative samples*)
  - For an input (w,c), we set the target vector as a vector with 1‘s at the position of c and the positions of the random words
  - Choose the activations of the negative samples as $\mathrm{f}(\mathrm{x}) = 1 - \sigma(x) = \sigma(-x)$
    - When $f(x) = 1$, then $\sigma(x) = 0$

CITEC NLLG

Skip-gram: practical realization

- Can think of it as follows:



Loss $= -\sum_i t_i \log y_i$

- With cross-entropy loss, the loss becomes:

$$\sigma(-x) = 1 - \sigma(x)$$

$$-[\ \log(\sigma(\boldsymbol{e}(\boldsymbol{w}) \cdot \boldsymbol{v_c})) + \sum_r \log(\sigma(-\boldsymbol{e}(\boldsymbol{w}) \cdot \boldsymbol{v_r}))\ ]$$

  - r are the negative samples

- This is not quite a standard neural network, but close

Toolkits for training word representations

Universität Bielefeld

word2vec

https://code.google.com/archive/p/word2vec/

GloVe

http://nlp.stanford.edu/projects/glove/

- GloVe aims at reconciling the advantages of global co-occurrence counts and local context windows
- Applies additional trick: take the sum of the target/center vector **e(w)** and the context vector $\mathbf{v}_c$ of each word as representation
- Many more, but these are two popular ones
- Terminology:
  - **word representations ≈ word embeddings ≈ word vectors**
  - context-counting vs context-predicting representations, sparse vs dense

CITEC NLLG

Pre-Trained Embeddings

- Word2vec
  - trained on Google news (100 billion tokens)
  - vectors with Freebase naming, trained on news (100 billion tokens)
- GloVe
  - trained on Wikipedia (6 billion tokens)
  - trained on CommonCrawl (42 and 840 billion tokens)
  - trained on Twitter (27 billion tokens)

- Omer Levy: dependency-based embeddings trained on Wikipedia
  https://levyomer.wordpress.com/2014/04/25/dependency-based-word-embeddings/

- There are many embeddings nowadays, in all possible languages
  https://fasttext.cc/docs/en/crawl-vectors.html

Outline

Evalution of word embeddings

The look and feel of word representations

Universität Bielefeld

...

wiegen 0.0427915 -0.401344 -0.0667862 0.21649 0.00169907 -0.0687786 -0.195405 -0.0534437 -0.676733 -0.23975 -0.159674 -0.0402676 -0.0617923 0.284718 0.358523 -0.285709 -0.00736682 -0.254635 -0.22907 -0.186109 ...

einlegen 0.365857 -0.0339146 0.198442 -0.0961315 0.156193 0.253468 0.169963 -0.232588 -0.422901 -0.0750184 0.0236783 0.249385 -0.0122247 -0.584567 -0.0711365 0.254896 0.382103 0.352294 0.825432 0.277691 0.773015 ...

sprengstoff 0.06961 0.118456 0.00497905 0.581913 -0.326157 -0.0674812 -0.0926074 -0.254514 -0.458406 -0.225093 0.0424881 -0.142328 -0.138707 0.481305 0.183707 -0.626077 0.396159 0.156636 0.157851 -0.441935...

vulkan 0.0322022 -0.429981 0.352328 -0.0530384 -0.366048 0.44187 -0.265227 -0.223954 -0.369078 -0.203064 0.158458 0.169517 0.448234 0.497058 -0.20855 0.046978 0.180444 0.290595 0.00907329 0.130582 0.0378717 -0.339296 0.399039...

...

Evaluating word representations

Universität Bielefeld

- Extrinsic

  by the performance of a model that uses the word representations for solving a task

  - Named entity recognition (accuracy), machine translation (BLEU score), summarization (ROUGE score), information retrieval (coverage)…
  - Compare performance of two models that only differ in the word representations they use

- Intrinsic

  by using the representations directly

  - Word Similarity Task
  - Word Analogy Task
  - Word Intrusion Task

Extrinsic Evaluation: Setup

- Say, our task is POS tagging
- Our labeled training data

Word	Label
The	DET
cat	NN
on	PREP
the	DET
mat	NN
.	PUNC

Extrinsic Evaluation: Setup

Universität Bielefeld

- Say, our task is POS tagging
- Our labeled training data; replace words with their embeddings

x	t
E(The)	1-hot(DET)
E(cat)	1-hot(NN)
E(on)	1-hot(PREP)
E(the)	1-hot(DET)
E(mat)	1-hot(NN)
E(.)	1-hot(PUNC)

CITEC NLLG

Extrinsic Evaluation: Setup

Universität Bielefeld

- Say, our task is POS tagging
- Our labeled training data; replace words with their embeddings; usually add some context

x	***t***
E(SOS);E(The);E(cat)	1-hot(DET)
E(The);E(cat);E(on)	1-hot(NN)
E(cat); E(on); E(on)	1-hot(PREP)
E(on); E(the); E(mat)	1-hot(DET)
E(the); E(mat); E(.)	1-hot(NN)
E(mat); E(.); E(EOS)	1-hot(PUNC)

CITEC NLLG

DISCUSS

Universität Bielefeld

Assume you would use an MLP as a model in the previous setting. How would inputs, outputs, etc. look like?

CITEC NLLG

Extrinsic Evaluation: Setup

- Say, our task is POS tagging

- Our labeled training data; replace words with their embeddings; usually add some context [not necessary with other architectures such as RNN; see later lectures]

$\boldsymbol{x}$	$\boldsymbol{t}$
E(SOS);E(The);E(cat)	1-hot(DET)
E(The);E(cat);E(on)	1-hot(NN)
E(cat); E(on); E(on)	1-hot(PREP)
E(on); E(the); E(mat)	1-hot(DET)
E(the); E(mat); E(.)	1-hot(NN)
E(mat); E(.); E(EOS)	1-hot(PUNC)

- Now train with different embeddings and look which one performs best

Word Similarity Task

Universität Bielefeld

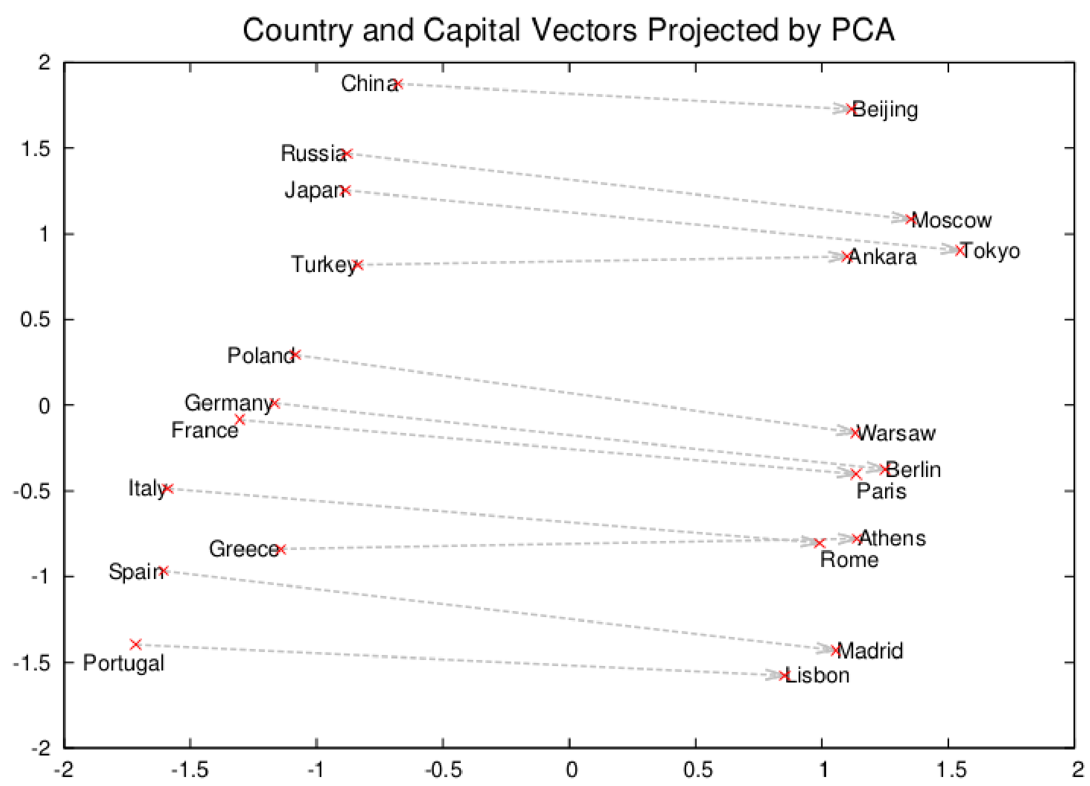
- Similar words should have similar representations
  - Similarity Dataset: http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/
    Scores from 0 to 10 by human raters
  - Intrinsic evaluation of embeddings:
    → quantify similarity by similarity of word vectors
    → evaluate correlation with juman judgements

Word 1	**Word 2**	**Human (mean)**	**Learned vectors**
tiger	cat	7.35	cossim(tiger, cat)
book	paper	7.46	cossim(book, paper)
plane	car	5.77	cossim(plane, cat)
smart	student	4.62	cossim(smart, student)
stock	phone	1.62	cossim(stock, phone)
....			
			....

Relations between word vectors

Universität Bielefeld

- Mikolov et al. (2013)



CITEC NLLG

How to find analogies?

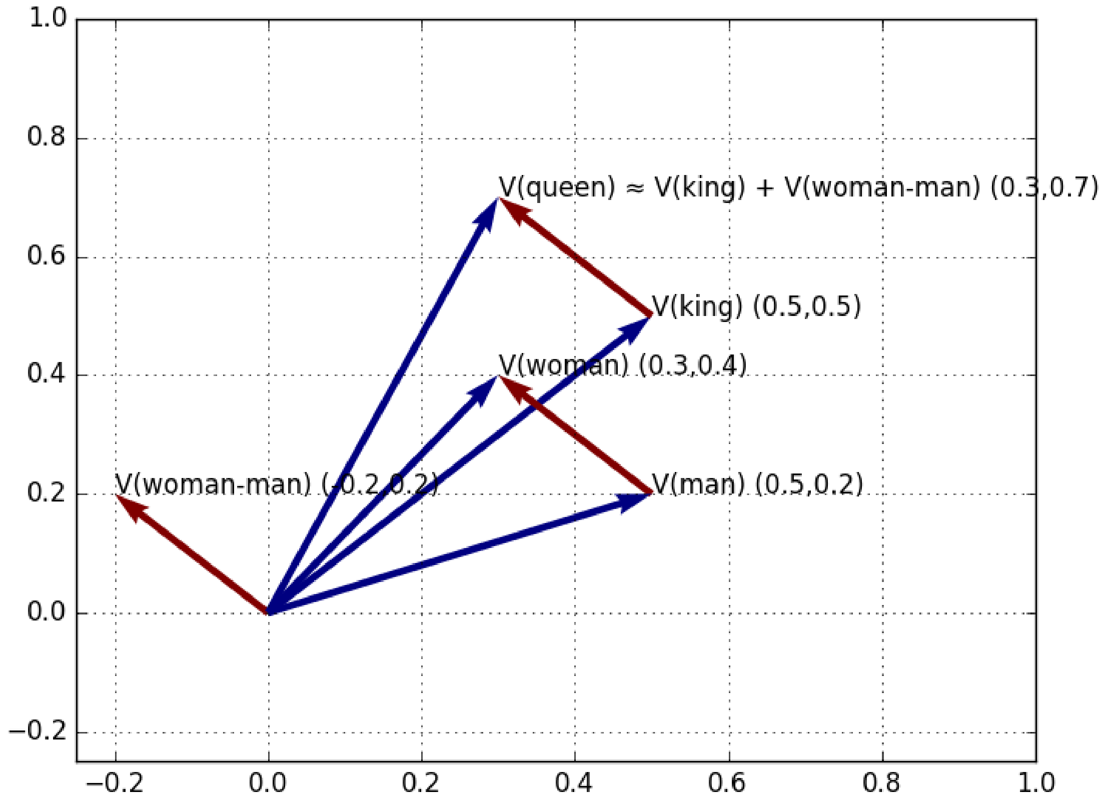
- A is to B as C to ?
  - Germany is to Berlin as France to x
- Find x such that:
  - vec(x) = vec(“Berlin”) - vec(“Germany”) + vec(“France”)

- Most famous example:

$$\text{KING} - \text{MAN} + \text{WOMAN} = \text{QUEEN}$$

KING-MAN+WOMAN=QUEEN

Universität Bielefeld



CITEC NLLG

Semantic analogies

Universität Bielefeld

- All examples from: //code.google.com/p/word2vec/source/browse/trunk/questions-words.txt
- capital-common-countries
  - *Athens Greece Baghdad* ***Iraq***
  - *Athens Greece Berlin* ***Germany***
- currency
  - *Denmark krone Croatia* ***kuna***
  - *Europe euro Hungary* ***forint***
- family
  - *boy girl brother* ***sister***
  - *brother sister dad* ***mom***

Syntactic analogies

Universität Bielefeld

- adjective-to-adverb
  - *amazing amazingly apparent* ***apparently***
- comparative
  - *bad worse big* ***bigger***
- present-participle
  - *code coding dance* ***dancing***
- past-tense
  - *dancing danced decreasing* ***decreased***
- plural
  - *banana bananas bird* ***birds***
- 3rd person verbs
  - *decrease decreases eat* ***eats***

Try it out!

- word2vec code: ./demo-analogy.sh

```
Enter three words (EXIT to break): loud louder slow

Word: loud  Position in vocabulary: 9481

Word: louder  Position in vocabulary: 18502

Word: slow  Position in vocabulary: 2188

                                              Word              Distance
------------------------------------------------------------------------
                                            faster              0.546676
                                            slower              0.545372
                                         efficient              0.445998
                                          downside              0.426637
                                           cheaper              0.419650
                                            slowly              0.418785
                                           slowing              0.418599
                                           gradual              0.417532
                                           quicker              0.404312
```

Practical Guidelines

- Word2Vec and Glove are pretty good tools
- Fast, give good word embeddings
- However, many other embeddings out there (see next lectures)
- Always try out different embeddings --- consider them as another hyperparameter
  - Results may vary drastically with different embeddings

DISCUSS

Universität Bielefeld

When and why are embeddings helpful?

CITEC NLLG

Summary

- Vectors are useful for representing words
  - Dense vs sparse representations
  - Projecting co-occurrence counts to low-dimensional vectors vs directly learning low-dimensional vectors
- Learning low-dimensional vectors
  - Inspired by neural language modeling
  - CBOW and Skip-gram model
  - Negative sampling
- Evaluating word representations
  - Extrinsic vs intrinsic evaluation

- Terminology:

  word representations ≈ word embeddings ≈ word vectors

References

- Levy & Goldberg: word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method, preprint on arxiv: *arXiv:1402.3722,* 2014.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J.: Distributed representations of words and phrases and their compositionality, in *Advances in neural information processing systems*: 3111-3119, 2013.
- McDonald, S., & Ramscar, M.: Testing the distributional hypothesis: The influence of context on judgements of semantic similarity, in *Proceedings of the 23rd annual conference of the Cognitive Science Society*, 2001.
- Bengio, Y., Schwenk, H., Senécal, J. S., Morin, F., & Gauvain, J. L.: Neural probabilistic language models, in *Innovations in Machine Learning*: 137-186, 2006.
- Collobert, R., & Weston, J.: A unified architecture for natural language processing: Deep neural networks with multitask learning, in *Proceedings of the 25th international conference on Machine learning*: 160-167, 2008.
- Derweester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., & Harshman, R.: Indexing by latent semantic analysis , in *Journal of the American Society for Information Science 41*: 391-407, 1990.
- Harris, Z. S.: Distributional structure. *Word 10(2-3)*: 146-162, 1954.
- John R. Firth: A synopsis of linguistic theory 1930–55, in *Studies in Linguistic Analysis* (special volume of the Philological Society), 1–32, 1957.

References

- Baroni and, M., Dinu, B. & Kruszewski, G.: Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014
- Levy, O. & Goldberg, Y.: Neural Word Embedding as Implicit Matrix Factorization, in *Advances in Neural Information Processing Systems 27*, 2014
- Arora, S., Li, Y., Liang, Y., Ma, T. & Risteski, A.: A Latent Variable Model Approach to PMI-based Word Embeddings, in *Transactions of the Association for Computational Linguistics*, 2016
- Stratos, K., Collins, M. & Hsu, D.: Model-based Word Embeddings from Decompositions of Count Matrices, in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, 2015
- Faruqui, M., Tsvetkov, Y., Rastogi, P., Dyer, C.: Problems With Evaluation of Word Embeddings Using Word Similarity Tasks. CoRRabs/1605.02276, 2016
- Ma, X. & Hovy, E.: End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF, *arXiv:1603.01354*, 2016