AGH University of Science and Technology

Seminar in *Artificial Intelligence*

Word embedding

Marcin Trebunia, Dominik Rygiel, Konrad Adamczyk

Department of Telecommunications

27.05.2019

Agenda

1. **Introduction**
   - Why do we need word embedding
   - Types of encoding
   - General concept
2. **Word embedding models**
   - Training approaches
   - Word2Vec
   - GloVe
   - FastText
3. **Applications**
   - Natural Language Processing
   - Other domains
4. **Problems and limitations**

Introduction

What is word embedding?

Introduction

Word embeddings are one of the few currently successful applications of unsupervised learning. Their main benefit arguably is that they don't require expensive annotation, but can be derived from large unannotated corpora that are readily available. Pre-trained embeddings can then be used in downstream tasks that use small amounts of labeled data.

NLP Research Scientist, Sebastian Ruder

Introduction

What a **lovely** day.
What a **nice** day.

Why we need to encode text?

Machine learning models take vectors (arrays of numbers) as input.

One hot encoding

$$
\begin{aligned}
\text{What} &= [1 \;\; 0 \;\; 0 \;\; 0 \;\; 0] \\
\text{a} &= [0 \;\; 1 \;\; 0 \;\; 0 \;\; 0] \\
\text{lovely} &= [0 \;\; 0 \;\; 1 \;\; 0 \;\; 0] \\
\text{nice} &= [0 \;\; 0 \;\; 0 \;\; 1 \;\; 0] \\
\text{day} &= [0 \;\; 0 \;\; 0 \;\; 0 \;\; 1]
\end{aligned}
$$

One hot encoding (cont.)

- Words completely independent of each other
- Inefficient approach: vector is sparse

One hot encoding (cont.)

Example:

- Dictionary of 10,000 words
- One hot encode each word
- Each vector's elements are 99.99% zeros!

Unique number encoding

$$
\begin{aligned}
\text{What} &= [1] \\
\text{a} &= [2] \\
\text{lovely} &= [3] \\
\text{nice} &= [4] \\
\text{day} &= [5]
\end{aligned}
$$

Unique number encoding (cont.)

+ Efficient - dense vector
– Encoding arbitrary - does not catch relationships between words
– Can be challenging for a model to interpret

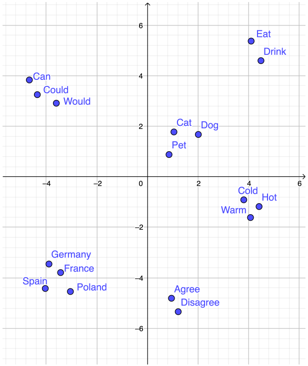
Word embedding

$$\begin{aligned}
\text{What} &= [1.2 \quad -0.1 \quad 4.3 \quad 3.2] \\
\text{a} &= [0.4 \quad 2.5 \quad -0.9 \quad 0.5] \\
\text{lovely} &= [2.1 \quad 0.3 \quad 0.1 \quad 0.4] \\
\text{nice} &= [2.0 \quad 0.4 \quad 0.3 \quad 0.5] \\
\text{day} &= [3.0 \quad -0.6 \quad 3.5 \quad -0.8]
\end{aligned}$$

Word embedding

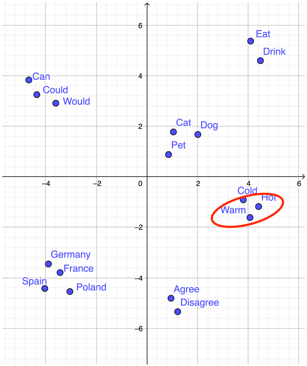
- Words with similar context occupy close spatial positions
- The cosine of the angle between words' vectors should be close to 1 (angle close to 0)

Word Embedding



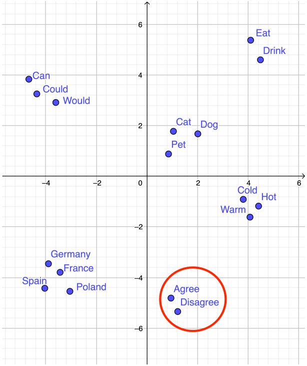
Example result of word embedding

Word Embedding



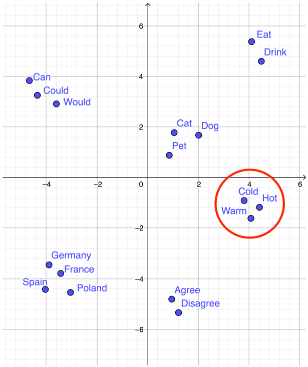
Words are synonyms

Word Embedding



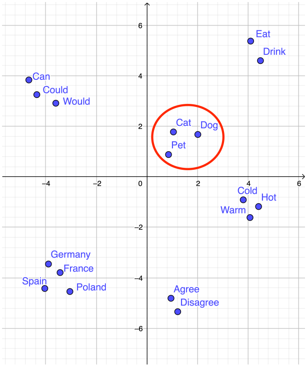
Words are antonyms

Word Embedding



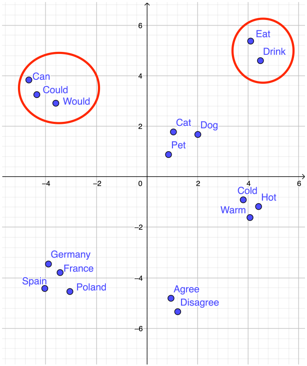
Words are value on a scale

Word Embedding



Words are hyponym - hypernym

Word Embedding



Words appear in similar context

Word embedding models

- Training approaches
- word2vec
- GloVe
- FastText

How to train my embedding model?

- CBOW
- Skip-gram

CBOW

Words representation

- Continuous Bag-of-Words
- Prediction of current words based on context
- Context is determined by surrounding words

CBOW
Words representation

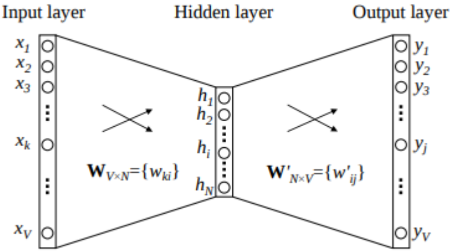


Figure: Simple CBOW model with one word in the context

CBOW

Words representation

$$p(w_j|w_I) = \frac{\exp\left({\mathbf{v}'_{w_j}}^T \mathbf{v}_{w_I}\right)}{\sum_{j'=1}^{V} \exp\left({\mathbf{v}'_{w_{j'}}}^T \mathbf{v}_{w_I}\right)}$$

Skip-gram

Words representation

- Continuous Skip-gram
- Predicting the surrounding words based on the context
- Context is the current word

CBOW vs Skip-gram

Words representation

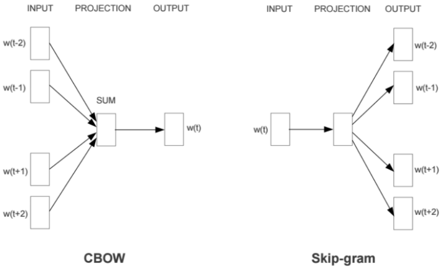


Figure: CBOW vs Skip-gram

Word2Vec

Word embedding models

- Created by researchers at Google in 2013
- Can use either CBOW or skip-gram
- Input is a corpus of text
- Produces vector space with unique word

Word2Vec

Interesting parameters

- Dimensionality!
- Training algorithm - softmax vs negative sampling
- Context window

GloVe

Word embedding models

- Global Vectors for Word Representation
- Comes from Stanford University, open-source
- Kind of extension of word2vec
- Training performed on aggregated, global word-word co-occurence statistics

GloVe

Word embedding models

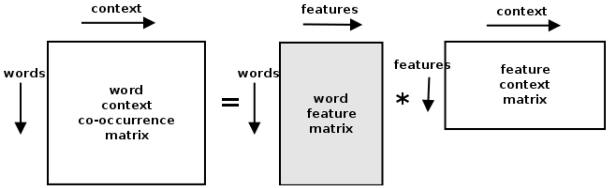


Figure: Co-occurence statistics of words

FastText

Word embedding models

- Incorporate sub-word information!
- Naturally support out-of-vocabulary words
- Uses skip-gram with negative sampling

FastText

Word embedding models

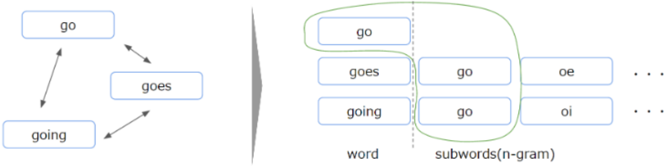


Figure: FastText subwords example

How can we use it?

Natural Language Processing

- If user search for "Dell notebook battery size" we would like to match it also with "Dell laptop battery capacity"
- If user search for "Cracow Motel" we would like to match it also with "Krakow Hotel"

Natural Language Processing

- Analyzing survey responses
- Analyzing comments

Other domains

- Word2vec can catch relationships and contexts in songs the user listens to
- Data can be used for real-time music recommendation

Problems and limitations

- Multiple meanings of a word: solution - *Sense* embeddings
- Inability to handle unknown or out-of-vocabulary (OOV) words
- Scaling to new languages
- No shared representations at sub-word levels

Thank you for your attention!

Q & A

References

Intro to word embeddings.
https://www.tensorflow.org/alpha/tutorials/text/word_embeddings.
Accessed: 2019-05-11.

Introduction to word embedding and word2vec.
https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060f
fbclid=IwAR3c2RpZOmbWC84_mKFtRI6PwTD7vJRxiquKPp2Y3en3_OfDpBsWjjSinv8.
Accessed: 2019-05-11.

References (cont.)

Word embeddings and their challenges.
http://blog.aylien.com/word-embeddings-and-their-challenges/.
Accessed: 2019-05-12.