Machine Learning Course

Lecture 8

MIPT, 2019

Plan

- NLP tasks
- Word representations
- LSA
- TFIDF

Based on YSDA NLP lectures

Natural Language Processing (NLP)

This is a subfield of computer science about how to program computers to deal with natural languages (English, Russian...).

NLP tasks (1)

- Machine Translation
  Between different languages
- Language modeling
  Model which can predict probability of sentence, or word, given context
- Part of speech tagging
  Determine part of speech for each word
- Parsing
  Determine parse tree for sentence which shows relations between words

NLP tasks (2)

- Natural language generation
  Convert some information (images, digits) into human readable way
- Named entity recognition
  Determine which items in text map to proper classes. For example, people or organizations
- Question answering
  Given a human language question, determine its answer
- Topic modeling
  Extract topics and determine which relate to text
- ...

To deal with NLP tasks we need somehow represent words and texts in computer adapted way.

Word representations

We can regard word as a discrete symbol.

One-hot vectors of dictionary size (ex. 500,000).

mother = [0...0,1,0...0]

cat = [1,0...0]

One-hot problems

Example: in web search, if user searches for “**Seattle motel**”, we would like to match documents containing “**Seattle hotel**”.

But:

motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]

- **These vectors do not contain meaning.**
- **There is no natural similarity measure between two such vectors.**
- **Each word is orthogonal for all other.**

How can we put the meaning into the vector?

Meaning from context

What is the meaning of word **bicycle**?

- Marie rode a **bicycle**
- **Bicycle** wheel was punctured
- The **bicycle** has a beautiful white bell.
- …

After these sentences we can say that bicycle is something what you can ride, it has wheel and bell.

How can we put this information into the vector?

Distribution semantic

We see that there is some distribution on contexts given word.

For example word “wheel” occurs more frequently with word bicycle than word “Moon”.

What other words fit these examples?

- Marie rode a _____
- _____ wheel was punctured
- The _____ has a beautiful white frame.

Distribution semantic

What other words fit these examples?

1. Marie rode a _____
2. _____ wheel was punctured
3. The _____ has a beautiful white bell.

Word	1	2	3	...
Bicycle	+	+	+	...
Bike	+	+	+	...
Car	+	+	-	...
Horse	+	-	-	...

Distributional hypothesis (Firth, 1957)

Words that are used and occur in the same contexts tend to purport similar meanings

Idea: co-occurrence counts

Corpus sentences

He also found five fish swimming in murky water in an old **bathtub**.

We do abhor dust and dirt, and stains on the **bathtub**, and any kind of filth.

Above At the far end of the garden room a **bathtub** has been planted with herbs for the winter.

They had been drinking Cisco, a fruity, wine-based fluid that smells and tastes like a mixture of cough syrup and **bathtub** gin.

Science finds that a surface tension on the water can draw the boats together, like toy boats in a **bathtub**.

In fact, the godfather of gloom comes up with a plot that takes in Windsor Davies (the ghost of sitcoms past), a **bathtub** and a big box of concentrated jelly.

'I'll tell him,' said the Dean from the bathroom above the sound of bathwater falling from a great height into the ample Edwardian **bathtub**.

Co-occurrence counts

the	12
a	9
of	7
and	6
in	5
…	…
like	2
water	2
boat	2
from	2
stain	1
toy	1
god-father	1
Cisco	1
…	…

vector

$$\begin{pmatrix} 12 \\ 9 \\ 7 \\ 6 \\ 5 \\ \vdots \\ 2 \\ 2 \\ 2 \\ 2 \\ 1 \\ 1 \\ 1 \\ 1 \\ \vdots \end{pmatrix}$$

Dimensionality reduction

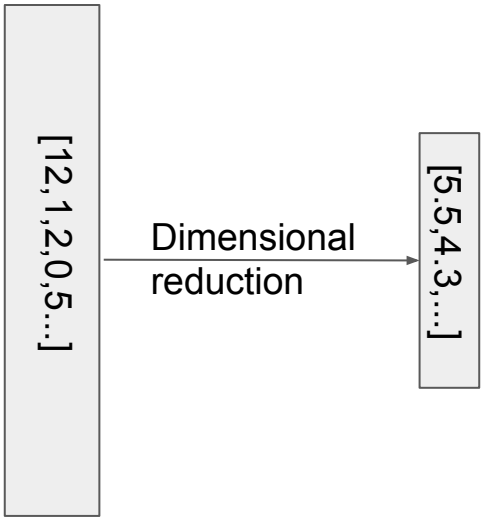
small vector

Context embeddings

v(word) = [12,1,2,0,5,...]

This is a big vector (vocabulary size)

We can squeeze it with dimensional reduction methods



Bag-of-words

Bag-of-words is a model where a text is represented as as bag (multiset) of its words.

We do not consider word order here.

```
Sentence: "John","likes","to","watch","movies","Mary","likes","movies","too"
BoW = {"John":1,"likes":2,"to":1,"watch":1,"movies":2,"Mary":1,"too":1}
```

N-gram Bag-of-words

Instead of words we do the same with n-grams:

```
[
    "John likes",
    "likes to",
    "to watch",
    "watch movies",
    "Mary likes",
    "likes movies",
    "movies too",
]
```

Singular Value Decomposition (SVD)

SVD: $M = U\Sigma V^T \quad U^T U = I$

- M (m x n) - real matrix
- U (m x m) - orthogonal matrix
- $\sum$ (m x n)- diagonal matrix with non-negative real numbers on the diagonal (singular values)
- V (m x m) - orthogonal matrix

Latent Semantic Analysis (LSA)

Let X be the matrix where t_i is i-th term (word) and d_j is j-th document.

X (vocab_size x corpus_size)

$$\begin{matrix} & & \mathbf{d}_j & & \\ & & \downarrow & & \\ \mathbf{t}_i^T \rightarrow & \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i,1} & \cdots & x_{i,j} & \cdots & x_{i,n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,j} & \cdots & x_{m,n} \end{bmatrix} \end{matrix}$$

Latent Semantic Analysis (LSA)

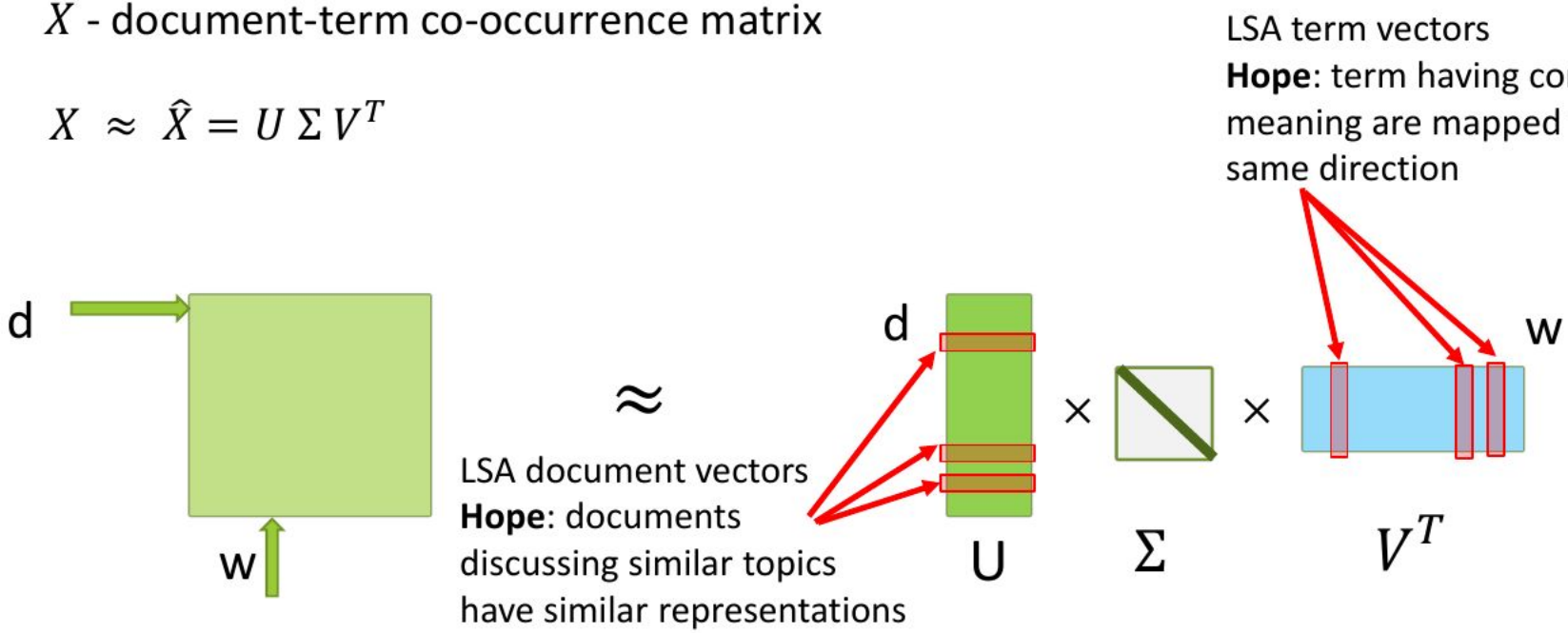
Dot product $t_i^T t_p$ gives correlation between two terms in the corpus, analogically with documents.

- Let's use SVD $$X = U\Sigma V^T$$
- If you take k biggest singular vectors you will get
  the rank k approximation to X with the smallest error (Frobenius norm)
- Now we can treat terms and documents as semantic space
- Finally we have small representations for terms and documents with meaning
  We can compute similarity with cosine metric.

Latent semantic analysis (LSA)

X - document-term co-occurrence matrix

$$X \approx \hat{X} = U \Sigma V^T$$



TF-IDF

Term frequency–inverse document frequency.

We want to represent sentence or document with a vector, based on words we see in this document (also n-grams)

TF-IDF

Term frequency: $$tf(t, d) = \frac{f_{t,d}}{\sum_l f_{l,d}}$$

- $f_{t,d}$ is number of times term t occurs in document d

Another way: $$\mathrm{tf}(t, d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{\max\{f_{t',d} : t' \in d\}}$$

- Such view is needed to prevent a bias towards long documents

TF-IDF

Inverse document frequency: $$\operatorname{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

- N - total number of documents in corpus
- Denominator is the number of documents where t appears

TF-IDF

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

$$\mathrm{tf}(t, d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{\max\{f_{t',d} : t' \in d\}}$$

$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

Probabilities

$$tf(t,d) = \frac{f_{t,d}}{\sum_l f_{l,d}} \qquad tf(t,d) \sim P(t)$$

$$\mathrm{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|} \qquad idf(t,D) \sim \log \frac{1}{P(t|d)}$$

$$tfidf(t,d) \sim -P(t)\log(P(t|d)))$$