Word embeddings

Victor Kitov

v.v.kitov@yandex.ru

Yandex
School of Data Analysis.

Standard word representations

- Denote V=vocabulary size.
- Standard word representations use sparse vectors $x \in \mathbb{R}^V$
    - $x_w = \mathbb{I}[w \text{ occured in the document}]$
    - $x_w = TF_w = \#[w \text{ occured in the document}]$
    - $x_w = TF_w IDF_w,\ IDF_w = \frac{N}{N_w}$
        - N - number of all documents
        - N_w - number of documents, containing w at least once.
- V is large, so **dense word representations (word embeddings)** $x \in \mathbb{R}^K$, $K << V$ are preferred
    - less inputs=>less parameters=>less overfitting
    - handle synonyms, like "car" and "automobile"

Interpretable word embeddings

- $x \in \mathbb{R}^K$, where x^i is some i-th interpretable feature, e.g.
    - x^1: part of speech
    - x^2: gender (for nouns)
    - x^3: tense (for verbs)
    - x^4: starts from capital letter
    - x^5: #[letters]
    - x^6: category: machine learning, physics, biology, ...
    - x^7: subcategory: supervised, unsupervised, semi-supervised learning
    - ...
- Need to invent features for each task and extract them.
- Want this to be done automatically!

Uninterpretable word embeddings

- Clustering words with similar meaning to similar representations.
- **Distributional hypothesis:**
  words have similar meaning $<=>$ they co-occur together frequently.
- "accuracy of SVM", "SVM gave accuracy", "lower accuracy, compared to SVM"
  - SVM and accuracy are connected!
- Typical dimensionality of embedding $\in [300, 500]$.

Table of Contents

SVD decomposition

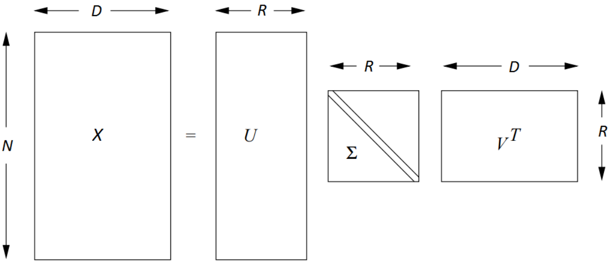
Every matrix $X \in \mathbb{R}^{NxD}$, rank $X = R$, can be decomposed into the product of three matrices:

$$X = U\Sigma V^T$$

where

- $U \in \mathbb{R}^{NxR}$, $\Sigma \in \mathbb{R}^{RxR}$, $V^T \in \mathbb{R}^{RxD}$
- $\Sigma = diag\{\sigma_1, \sigma_2, ...\sigma_R\}$, $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_R \geq 0$
- $U^T U = I$, $V^T V = I$, where $I \in \mathbb{R}^{RxR}$ is identity matrix.

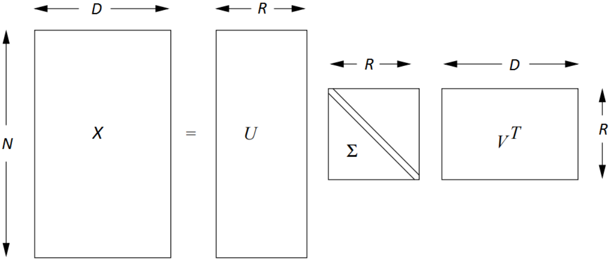
Interpretation of SVD



For X_{ij} let i denote objects and j denote properties.

- Columns of U - orthonormal basis of columns of X
- Rows of V^T- orthonormal basis of rows of X
- Σ - scaling.
- Efficient representations of low-rank matrix!

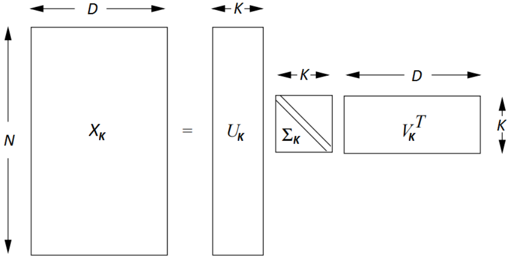
Interpretation of SVD



For X_{ij} let i denote objects and j denote properties.

- Rows of U are normalized coordinates of rows in V^T
- $\Sigma = diag\{\sigma_1, ...\sigma_R\}$ shows the magnitudes of presence of each row from V^T.

Truncated SVD decomposition



Simplification to rank $K \le R$:

$$X_K = U_K \Sigma_K V_K$$

$$\Sigma = diag\{\sigma_1, \sigma_2, ...\sigma_K, \sigma_{K+1}, ...\sigma_R\} \longrightarrow diag\{\sigma_1, \sigma_2, ...\sigma_K\} = \Sigma_K$$
$$U = [u_1, u_2, ...u_K, u_{K+1}, ...u_R] \longrightarrow [u_1, u_2, ...u_K] = U_K$$
$$V = [v_1, v_2, ...v_K, v_{K+1}, ...v_R] \longrightarrow [v_1, v_2, ...v_K] = V_K$$

- Now rows of U give reduced representation of rows of X.

Embeddings from term-document matrix factorization

- Form term-document matrix $M \in \mathbb{R}^{VxD}$:

$$M_{wd} = \#[\text{word } w \text{ appeared in document } d]$$

- **Latent semantic analysis (LSA):** apply truncated SVD of order K to M:

$$M = U_K \Sigma_K V_K^T$$

  - Rows of U_K give word embeddings!
  - Usually $K = 300$.

- M_{wc} may contain instead of TF: log(TF), TF-IDF.

Co-occurrence matrix

Form co-occurrence matrix $M=\{m_{wc}\}_{w\in V,c\in V}$:

1. Form single document by concatenating all documents from the collection
2. Slide rolling window of width $2W+1$ centered at word w over all words.
    - increase M_{wc} by count, how many times word c appears within $\pm W$ of word w

Embedding dimensionality

For word embeddings apply truncated SVD with $K \in [500, 5000]$.

- co-occurrence in documents: general topics
- co-occurrence in context: more specific topics
  => more degrees of freedom needed, than for LSA.

Table of Contents

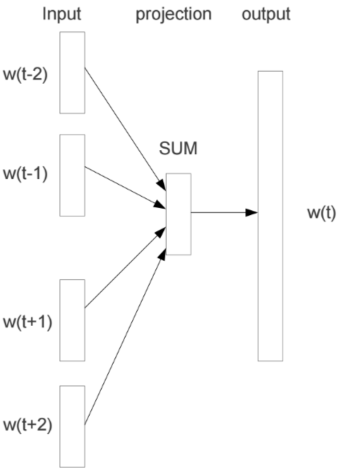
2. Word2vec
   - Models
   - Loss calculation optimizations

Word2vec

- Proposed in 2013[1].
- Computationally efficient:
  - Remove computationally expensive hidden layer.
  - Omit expensive denominator calculation.
- Thus can be trained on much bigger datasets.
  - better embeddings, especially for rare words.
- Comments: for each w models evaluate:
  - target word embedding v_w
  - context word embedding $\tilde{v}_w$
- Target&context embeddings may be averaged or concatenated later.

[1] Mikolov et al. (2013), Mikolov et al. (2013)

Continious bag of words (CBOW)



Continuous bag of words (CBOW)

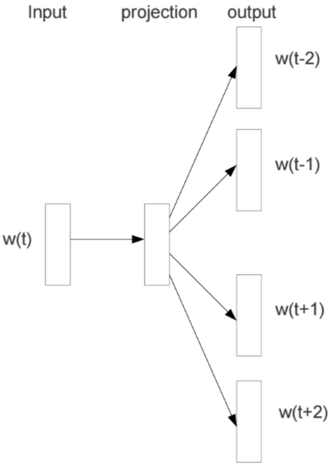
CBOW: predict current word given context.

$$\frac{1}{T}\sum_{t=1}^{T}\ln p(w_t|w_{t-c},..w_{t-1},w_{t+1},...w_{t+c})\to\max_{\theta}$$

where $v_{context}=\sum_{-c\le i\le c,\, i\ne 0}v_{w_{t+i}}$ and

$$p(w_t|w_{t-c},..w_{t-1},w_{t+1},...w_{t+c})=\frac{\exp\left(v_{context}^{T}\tilde{v}_{w_t}\right)}{\sum_{w=1}^{V}\exp\left(v_{context}^{T}\tilde{v}_{w}\right)}$$

Skip-gram model



Skip-gram model

Skip-gram: predict context, given current word:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le i\le c,\, i\neq 0}\ln p(w_{t+i}|w_t)\to\max_{\theta}$$

$$p(w_{t+i}|w_t)=\frac{\exp\left(v_{w_t}^T\tilde{v}_{w_{t+i}}\right)}{\sum_{w=1}^{V}\exp\left(v_{w_t}^T\tilde{v}_{w}\right)}$$

2 Word2vec
- Models
- Loss calculation optimizations

Optimizations

$$p(w_{t+i}|w_t) = \frac{\exp\left(v_{w_t}^T \tilde{v}_{w_{t+i}}\right)}{\sum_{w=1}^{V} \exp\left(v_{w_t}^T \tilde{v}_w\right)}$$

- Summation over all words in vocabulary is impractical.
- Two optimization approaches:
    - hierarchical soft-max
        - calculates probabilities in $O(\log_2 V)$
    - negative sampling
        - uses different optimization criteria

Software & precomputed embeddings

- gensim.models.word2vec
    - python wrapper for finding word2vec
- Word2vec tool & precomputed representations
    - fact c code for finding word2vec
    - pre-trained 300-dimensional vectors for 3 million words and phrases.
        - trained on Google News dataset (about 100 billion words).

Conclusion

- Word embeddings allow to map words to compact dense representations.
- Approaches:
  - matrix factorization
    - term-document matrix
    - co-occurrence matrix
  - neural net based (Skip-gram, CBOW)