PGR 210- Natural Language Processing Part

Kristiania University College

By Huamin Ren

Huamin.ren@kristiania.no

Høyskolen Kristiania

Outline of week 44

1. Topic models
   - Truncated SVD
   - LDiA
   - Word2Vec model
   - GloVec (optional)
2. Similary metric

Outline of week 44

1. Topic models
   - ***Truncated SVD***
   - LDiA
   - Word2Vec model
   - GloVec (optional)
2. Similary metric

1.1 Truncated SVD

- What is SVD? singular value decomposition
- Why SVD is contributing to LSA?

SVD is an algorithm for decomposing any matrix into three “factors,” three matrices that can be multiplied together to recreate the original matrix.

Høyskolen Kristiania

Connection between SVD and LSA

- LSA uses SVD to find *the combinations of words* that are responsible, together, for the biggest variation in the data.
  - ✓Line up the axes (dimensions) in your new vectors with the greatest “spread” or variance in the word frequencies
- New basis vectors are generated; each dimension (axes) becomes a combination of word frequencies rather than a single word frequency.

More about SVD

- SVD was in widespread use long before the term “machine learning” even existed
- SVD decomposes a matrix into three square matrices, one of which is diagonal.

Mathematics of SVD

$$W_{\text{mxn}} \Rightarrow U_{\text{mxp}}\, S_{\text{pxp}}\, V_{\text{pxn}}{}^{\text{T}}$$

m: number of terms in vocabulary
n: number of documents
p: number of topics (same as the number of words)

Høyskolen Kristiania

Application of SVD

- Matrix inversion

A matrix can be inverted by ***decomposing*** it into three simpler ***square matrices***, ***transposing matrices***, and then multiplying them back together.

Using SVD, LSA can break down your TF-IDF term-document matrix into three simpler matrices. And they can be multiplied back together to produce the original matrix, without any changes.

Even truncate!

- Truncate those matrices
  - Ignore some rows and columns before multiplying them back together, which reduces the number of dimensions in vector space model

Truncating the topics

- Hint:

```
from sklearn.decomposition import TruncatedSVD

svd = TruncatedSVD(n_components=16, n_iter=100)
print(tfidf_docs.shape)
svd_topic_vectors = svd.fit_transform(tfidf_docs)
```

Implement LSA using the same dataset (sms-spam.csv)

Høyskolen Kristiania

Outline of week 44

1. Topic models
   - Truncated SVD
   - ***LDiA***
   - Word2Vec model
   - GloVec (optional)
2. Similary metric

1.2.1 Introduction to LDiA

- LDiA, Latent Dirichlet allocation
- LSA should be your first choice for most topic modeling, semantic search, or content-based recommendation engines
- BUT LDiA can give slightly better results in some situations

- LDiA assumes a Dirichlet distribution of word frequencies

It’s more precise about the statistics of allocating words to topics than the linear math of LSA.

1.2.2 Principle of LDiA

LDiA creates a semantic vector space model (like your topic vectors) using an approach similar to how your brain worked.

In your thought experiment, you manually allocated words to topics based on how often they occurred together in the same document. *The topic mix for a document can then be determined by the word mixtures in each topic by which topic those words were assigned to*. This makes an LDiA topic model much easier to understand, because the words assigned to topics and topics assigned to documents tend to make more sense than for LSA.

- LDiA assumes that each document is a mixture (linear combination) of some arbitrary number of topics
- It assumes that each topic can be represented by a distribution of words (term frequencies)
- The probability or weight for each of these topics within a document, as well as the probability of a word being assigned to a topic, is assumed to start with a Dirichlet probability distribution

Outline of week 44

1. Topic models
   - Truncated SVD
   - LDiA
   - ***Word2Vec model***
   - GloVec (optional)
2. Similary metric

1.3.1 Introduction to Word2Vec

- Word2vec was first presented publicly in 2013 at the ACLconference

Word2vec embeddings were four times more accurate (45%) compared to equivalent LSA models (11%) at answering analogy question

- Word2vec could transform the natural language vectors of token occurrence counts and frequencies into a much lower-dimensional Word2vec vectors

In this lower-dimensional space, you can do your math and then convert back to a natural language space.

Høyskolen Kristiania

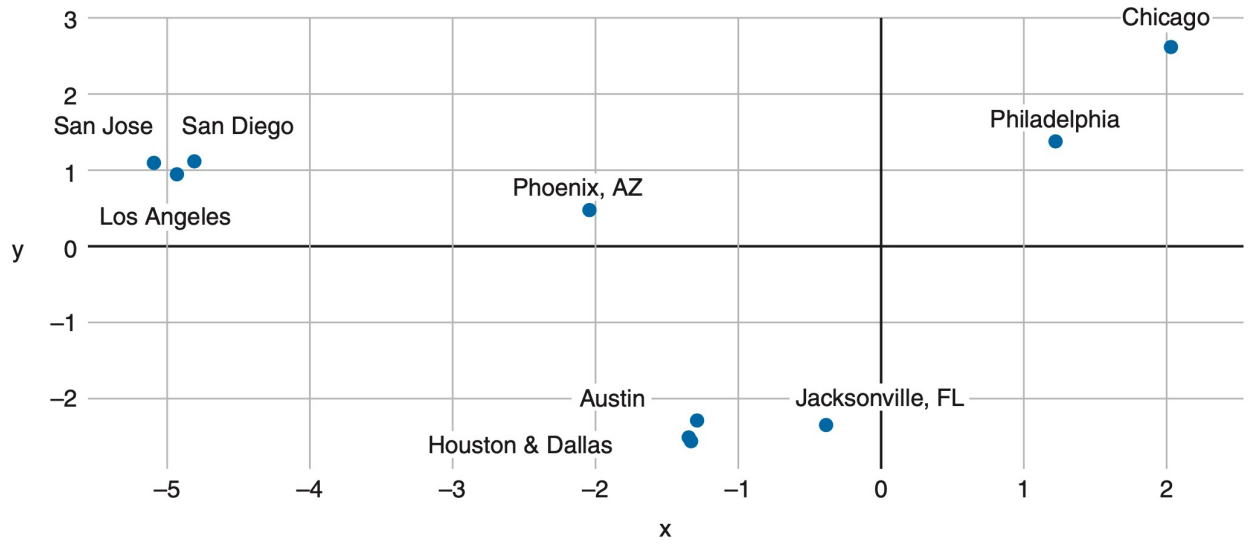
$$\vec{x}_{coffee} - \vec{x}_{coffees} \approx \vec{x}_{cup} - \vec{x}_{cups} \approx \vec{x}_{cookie} - \vec{x}_{cookies}$$

Distance between the singular and plural versions of a word

Høyskolen Kristiania

1.3.2 Reasons to use Word2Vec

- Useful for reasoning, analogy problems, pattern matching, modelling and visualization
- Can further do:
  - Visualize word vectors on a 2D semantic map
  - Great for chatbot and search engine



1.3.3 How to compute Word2Vec representations – use pre-trained model?

- Using gensim.word2vec module
  - Download pre-trained Word2vec models on Google News documents.
  - Limit the loaded words, if too memory intensive.
  - gensim.KeyedVectors.most_similar() method provides an efficient way to find the nearest neighbors for any given word vector.

A word vector model with a limited vocabulary will lead to a lower performance of your NLP pipeline if your documents contain words that you haven't loaded word vectors for.

Høyskolen Kristiania

1.3.4 How to generate your own word vector representations?

- How to create your own domain-specific word vector models?

Keep in mind, you need *a lot of documents* to do this

Høyskolen Kristiania

1.3.5 Look deeper into Word2Vec

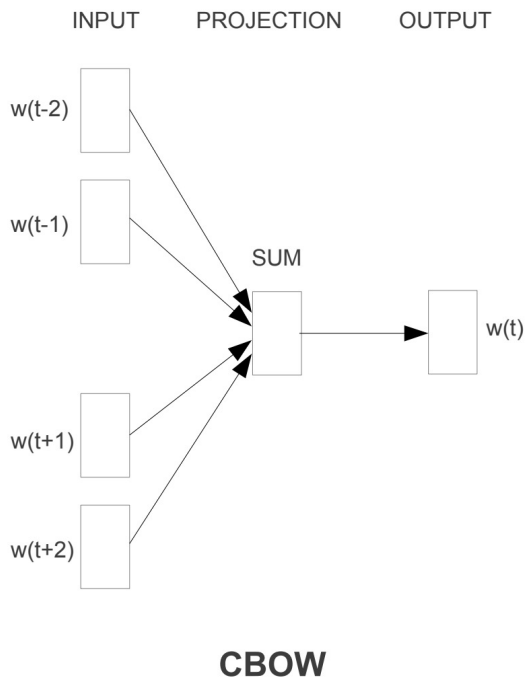
- This model is a predictive deep learning based model to compute and generate high quality, distributed, and continuous dense vector representations of words that capture *contextual* and *semantic* similarity.
- Essentially unsupervised models: take in massive textual corpora, create a vocabulary of possible words, and generate dense word embeddings for each word in the vector space representing that vocabulary.
- The dimensionality of this dense vector space is much lower than the high-dimensional sparse vector space built using traditional Bag of Words models.

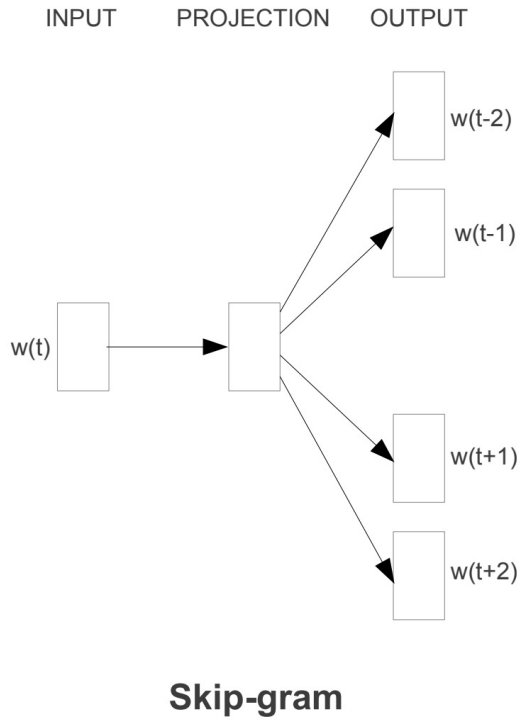
https://arxiv.org/pdf/1301.3781.pdf

Høyskolen Kristiania

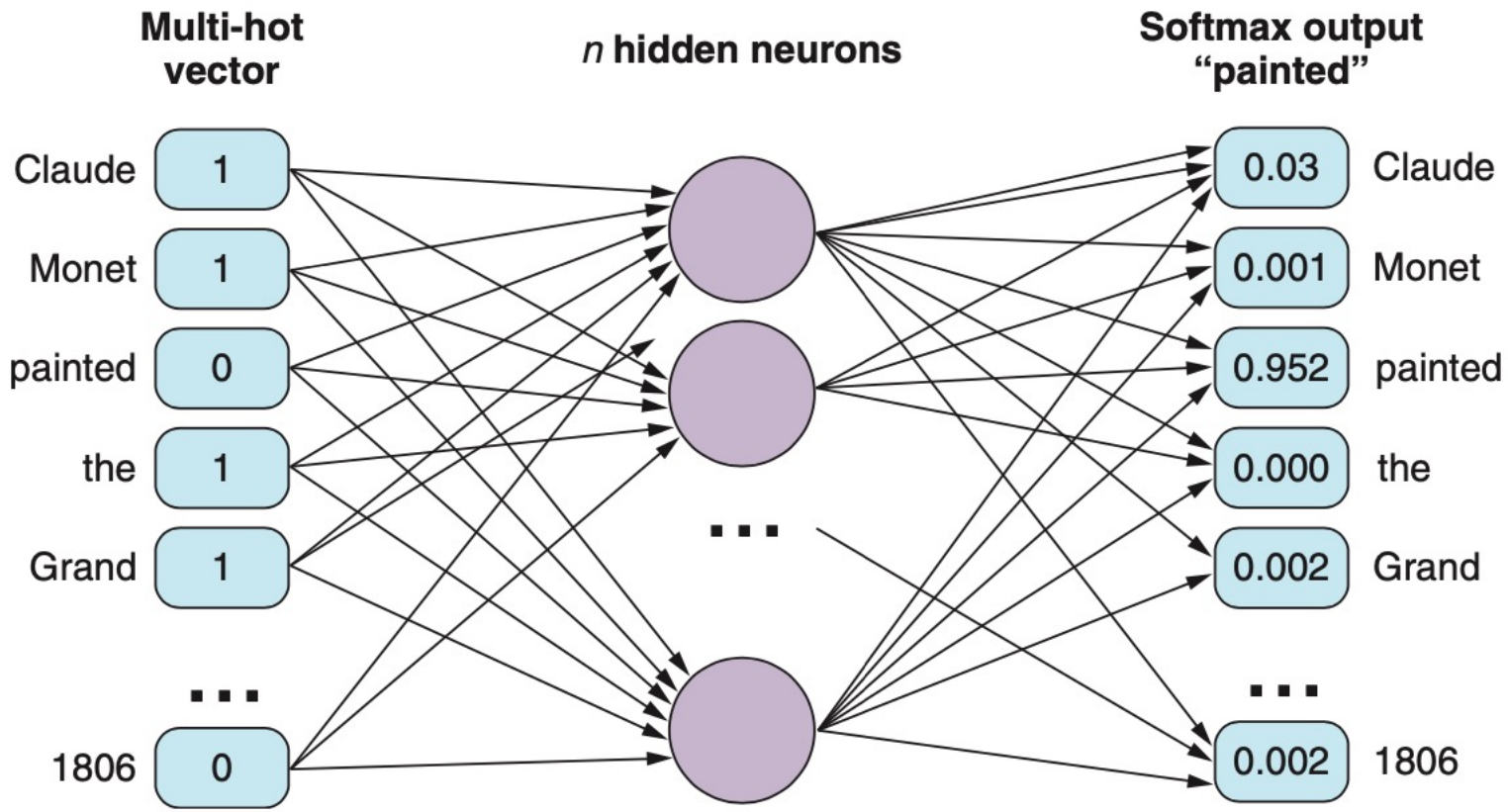
Two different model architectures

- The Continuous Bag of Words (CBOW) model
- The Skip-Gram model





øyskolen
ristiania



Outline of week 44

1. Topic models
   - Truncated SVD
   - LDiA
   - Word2Vec model
   - ***GloVec (optional)***
2. Similary metric

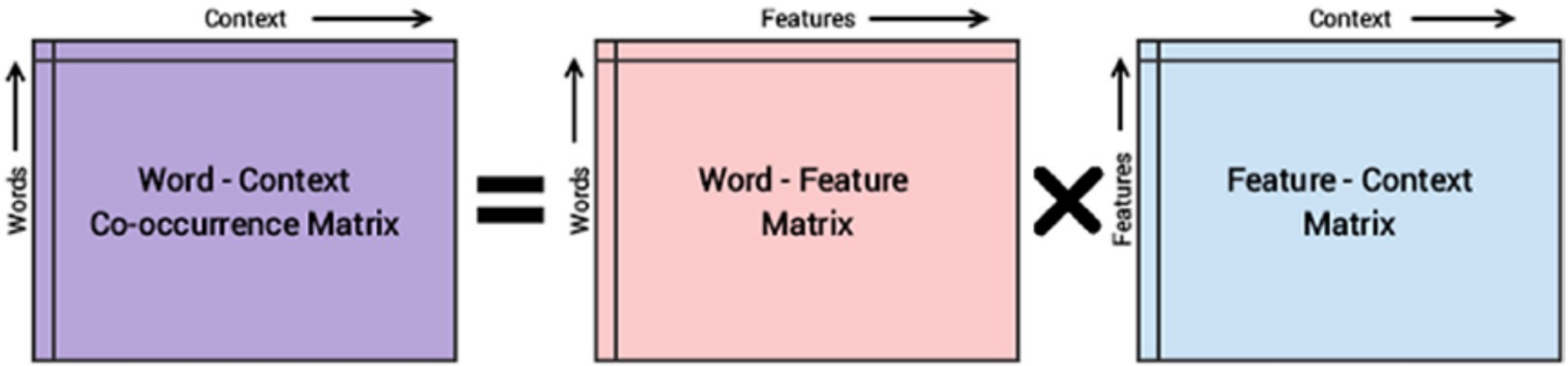
1.4.1

- Word2vec was a breakthrough, but it relies on a neural network model that must be trained using backpropagation.
- GloVec is applying direct optimization of the global vectors of word co-occurrences (co-occurrences across the entire corpus)

https://nlp.stanford.edu/pubs/glove.pdf?fileGuid=WyYwxqq8kWjKdWgd

Høyskolen Kristiania

1.4.2 Conceptual model for the GloVe model's implementation



Høyskolen Kristiania

1.4.3

- Considering the Word-Context (*WC*) matrix, Word-Feature (*WF*) matrix, and Feature-Context (*FC*) matrix, we try to factorize

$$\boldsymbol{WC} = \boldsymbol{WF}\ \boldsymbol{FC}$$

- Typically initialize *WF* and *FC* with some random weights and attempt to multiply them to get *WC'* (an approximation of WC) and measure how close it is to *WC*.
- Do this multiple times using *Stochastic Gradient Descent* (SGD) to minimize the error.

Høyskolen Kristiania

- WF (*Word-Feature):* word embeddings for each word, where *F* can be present to a specific number of dimensions.
- Similarity between Word2Vec and: build a vector space where the position of each word is influenced by its neighboring words based on their context and semantics.
- Difference in the two models: Word2Vec starts with local individual examples of word co-occurrence pairs; GloVe starts with global aggregated co-occurrence statistics across all words in the corpus.

Høyskolen Kristiania

Outline of week 44

1. Topic models
   - Truncated SVD
   - LDiA
   - Word2Vec model
   - GloVec (optional)
2. ***Similary metric***

2.1 Similarity scores

- Similarity scores (and distances) tell on how similar or how far apart two documents are based on the similarity (or distance) of the vectors you used to represent them
- Some familiar distances:
  - Euclidean or Cartesian distance, or root mean square error (RMSE): 2-norm or L2
  - Squared Euclidean distance, sum of squares distance (SSD): L2
  - Cosine or angular or projected distance: normalized dot product
  - Minkowski distance: p-norm or Lp
  - Fractional distance, fractional norm: p-norm or Lp for $0 < p < 1$
  - City block, Manhattan, or taxicab distance; sum of absolute distance (SAD)

Høyskolen Kristiania

Pairwise distances available in sklearn

- 'cityblock', 'cosine', 'euclidean', 'l1', 'l2', 'manhattan', 'braycurtis', 'canberra', 'chebyshev', 'correlation', 'dice', 'hamming', 'jaccard', 'kulsinski', 'mahalanobis', 'matching', 'minkowski', 'rogerstanimoto', 'russellrao', 'seuclidean', 'sokalmichener', 'sokalsneath', 'sqeuclidean', 'yule'

2.2 Distance

- Distance measures are often computed from similarity measures (scores) and vice versa such that distances are inversely proportional to similarity scores.
- For distances and similarity scores that range between 0 and 1, like probabilities, it's more common to use a formula like this:

  similarity = 1. - distance

  distance = 1. - similarity

Høyskolen Kristiania

$$hd(u,v) = \sum_{i=1}^{n} (u_i \neq v_i)$$

$$norm\_hd(u,v) = \frac{\sum_{i=1}^{n} (u_i \neq v_i)}{n}$$

Høyskolen Kristiania

2.3 Use similarity on?

- **Lexical similarity:** This involves observing the contents of the text documents with regards to its syntax, structure, and content and measuring their similarity based on these parameters.
- **Semantic similarity:** This involves determining the semantics, meaning, and context of the documents and then determining how close they are to each other.

Høyskolen Kristiania

- **Term similarity:** Similarity between individual tokens or words
- **Document similarity:** Similarity between entire text documents

Høyskolen Kristiania