Lecture 3

QM 701: Advanced Data Analytics

Fuqua School of Business

2024

Lecture outline

- Word Embeddings
- Word2vec Models
- Applications of Word Embeddings

Word Embeddings

Representing Words with Dense Vectors

Word Embeddings

Two possible approaches to representing word using vectors:

Sparse vectors

- Each vector’s dimension is the size of the vocabulary
- Most elements are zero
- Examples: one-hot encoding or co-occurrence matrix
- Obtained through simple count-based algorithms

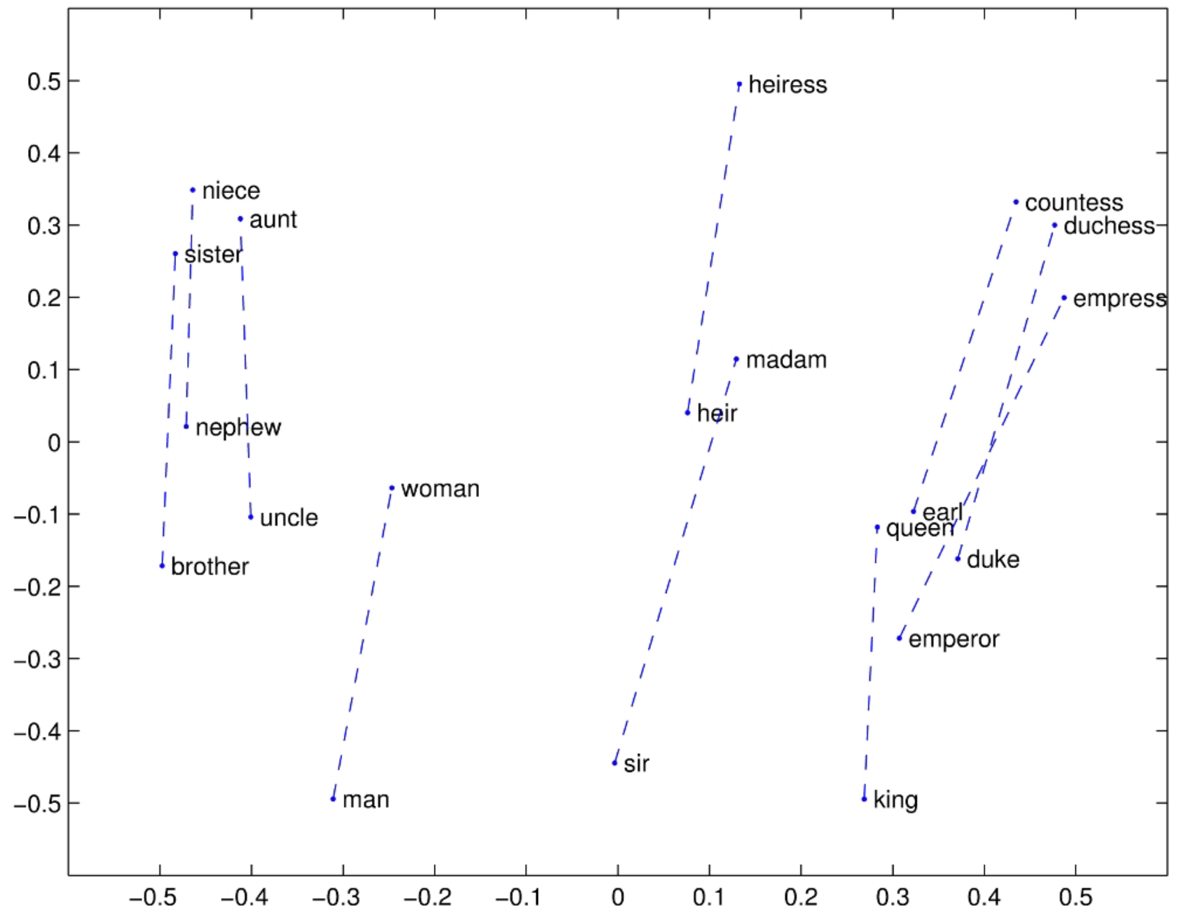
Dense vectors

- Each vector’s dimension is around 50-1000
- Most elements are non-zero
- Examples: word2vec, fasttext, GloVe, etc
- Obtained through complex training algorithms

Ideal Properties of Word Embeddings

- Word with similar meanings have similar vectors
  - Identify similar vectors using `cosine similarity’
- Capture word relatedness
  - Identify word relationships using the parallelogram model, i.e., answering questions such as “apple is to tree as grape is to?”

Example of a Word Embedding:



Why do we need word embeddings?

An effective word embedding can be applied to many NLP tasks downstream, some applications of word embeddings include:

- Analyzing trends and biases in texts
- Domain specific textual analysis
- Identify similar documents
- Query expansion for search

Looking Ahead: Contextual Embedding

- In this class, we will focus on static word embeddings, i.e., represent a word/token with a fixed dense vector.
- Later, we will look at contextual word embeddings, i.e., represent the meaning of a word/token with a dense vector within the context
  - Under contextual word embedding, the same word, under different contexts, would be represented differently.
  - Contextual word embedding is more effective in many NLP tasks where the context is often important. However, it is harder to train and more difficult to interpret the vectors.
  - As we shall see, contextual word embedding also represent each sentence as a vector. If we are representing a sentence using a static embedding, we would resort to basic methods such as summing all word vectors in the sentence.

Pre-trained NLP Models

- A pre-trained NLP model is model, typically trained on large datasets to accomplish a specific task, then applied downstream for other tasks.
- Example 1: word2vec (Skipgram) is a pre-trained model that predicts the surrounding words given a center word.
  - Example application: identify similar words/tokens
- Example 2: BERT is a pre-trained model that predicts the masked words in given documents.
  - Example application: detect the sentiments of a document

Word Embedding Models We Will Use

- Word2vec
  - Treats each word in a corpus like an atomic entity
  - Returns an error when handle out-of-vocabulary tokens
- Fasttext
  - Treats each word as composed of character n-grams
  - Can handle out-of-vocabulary tokens or misspelled words

word2vec

A Foundational Algorithm for Word Embeddings

Two Paradigms of Machine Learning in NLP

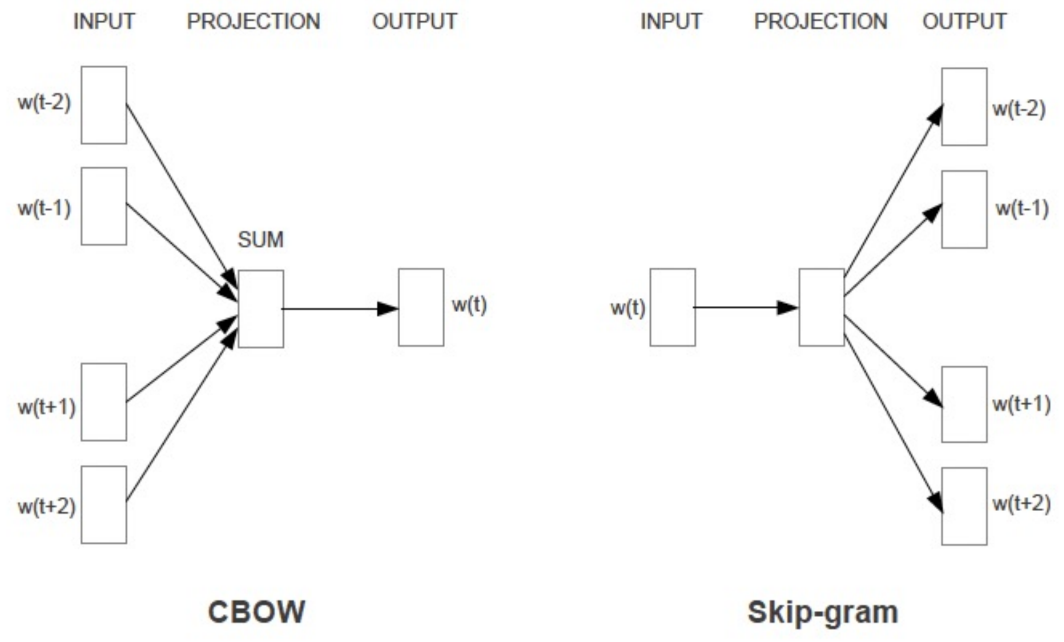
- Supervised Learning
  - Training a model on a labeled dataset with the correct output
  - Example 1: Naïve-Bayes (Class 1)
  - Example 2: Feedforward Neural Networks (Class 2)
- Unsupervised Learning
  - Training a model on data without any labels
  - Example 1: **word2vec (this class)**
    - Word2Vec belongs to a subclass of unsupervised learning called **self-supervised learning**; which generates its own labels from an unlabeled dataset then apply supervised learning
  - Example 2: Topic Modeling (next class)
  - Example 3: Language Models (class 5 and 6)

Key Ideas of Word2vec (Skip-gram) Model

- We train a model to perform the following prediction tasks:
  - Given a center word, say, ‘throw’, what is the probability another word, say, ‘basketball’, shows up nearby?
  - The parameters in the model are exactly the word embeddings
  - The training data is an unlabeled corpus
- The model’s performance on predicting the nearby words is not important, it is more about the useful of the word embedding obtained from the trained model for subsequent tasks

Two Models in Word2vec

The two models in word2vec that differ in the prediction tasks:



Continuous Bag of Words (CBOW)

- Predicts the center word
- Faster to train, slightly better embedding for frequent words in a large corpus

Skip-gram

- Predicts the nearby words
- Works better with a small corpus, better embedding with rare words

Applying the Word2vec Model

Training a Word2vec Model in Python (via genism)

```
w2v_sg_ABC = Word2Vec(sentences=abc_text, vector_size=100, window=5, min_count=5, epochs=10, workers=4, sg=1)
```

- \`sentences’: Corpus used to train the word2vec mode
- \`vector_size\`: Dimensionality of the word vectors.
- \`window\`: Maximum distance between the current and predicted word within a sentence.
- \`min_count\`: Ignores all words with total frequency lower than this.
- \`epochs\`: Number of training iterations (epochs) over the corpus.
- \`workers\`: Use these many worker threads to train the model (=faster training with multicore machines).
- \`sg\`: Training algorithm: 1 for skip-gram; 0 for CBOW.

Finding Similar Words in a Corpus

Once we trained a word2vec model, we can identify similar words in the corpus using cosine similarity

Code: `w2v_sg_ABC.wv.most_similar(positive="sydney")`

Output:

```
[('melbourne', 0.7940919995307922),
 ('prince', 0.7017743587493896),
 ('monash', 0.6924399137496948),
 ('macquarie', 0.6855758428573608),
 ('symposium', 0.6843878626823425),
 ('adelaide', 0.684113621711731),
 ('fremantle', 0.683067262172699),
 ('chicago', 0.6760169267654419),
 ('kate', 0.6701173782348633),
 ('astrobiology', 0.6657502055168152)]
```

Detecting Cultural Bias Using Word Embeddings

Example:

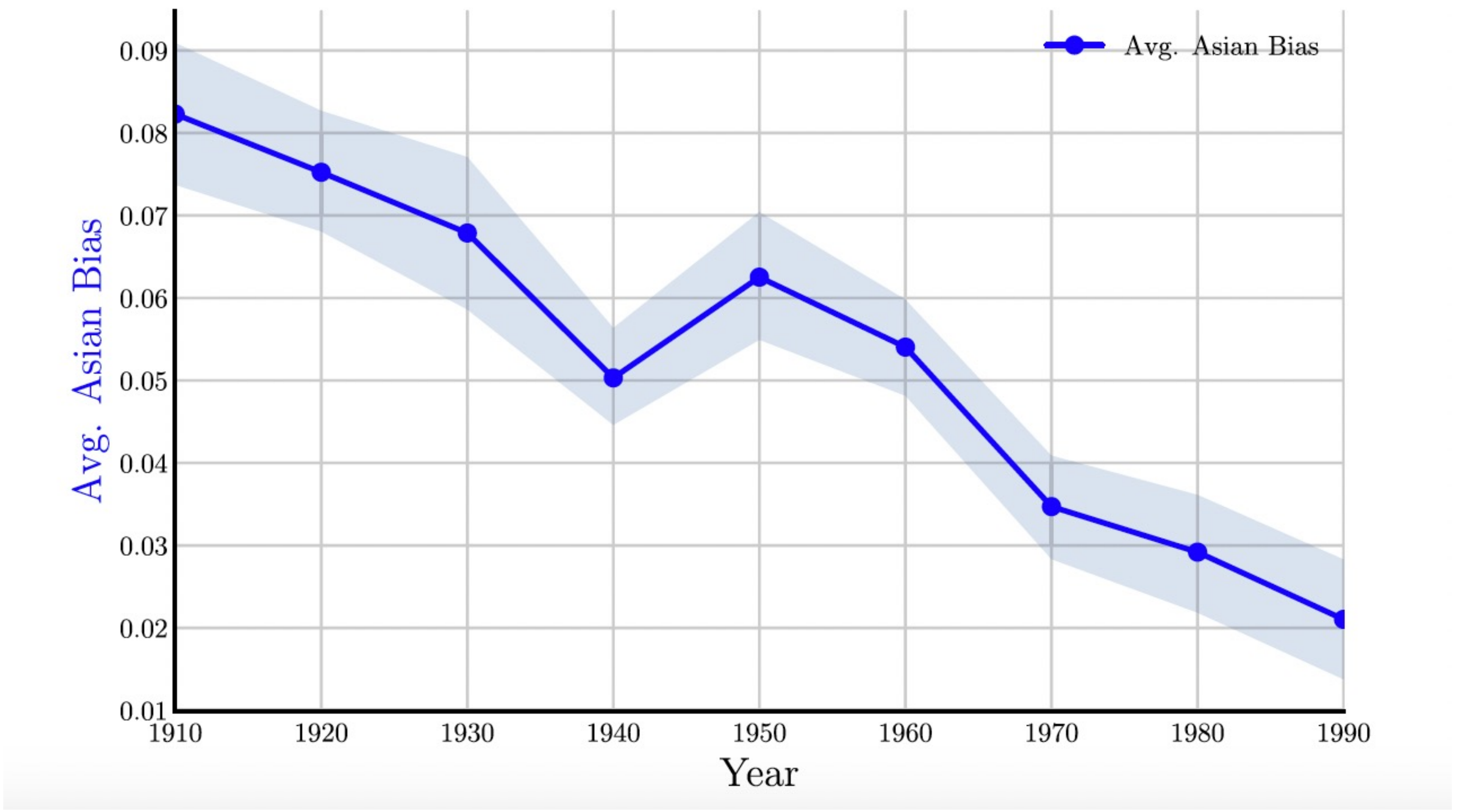
- Ask “Paris:France :: Tokyo:??”
    - Returns ?? = Japan

Findings:

- Ask “father:doctor :: mother:??”
    - ?? = nurse
- Ask “man:computer programmer :: woman:??”
    - ?? = homemaker

Source:
Bolukbasi, Tolga, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam T. Kalai. "Man is to computer programmer as woman is to homemaker? debiasing word embeddings." In Advances in Neural Information Processing Systems, pp. 4349-4357. 2016.

Word Embeddings Can Reflect Ethnic Stereotypes Over Time.



Change in association of Chinese names with adjectives framed as "othering" (barbaric, monstrous, bizarre)

Source:
Garg, Nikhil, Schiebinger, Londa, Jurafsky, Dan, and Zou, James (2018). Word embeddings quantify 100 years of gender and ethnic stereotypes. Proceedings of the National Academy of Sciences, 115(16), E3635–E3644

Homework 3

Train and apply Word2Vec dataset of tweets

- Q1: Text Pre-Processing. You will start by importing and preprocessing a dataset of movie reviews.
- Q2: Model Training. You need to train **Word2Vec** and **FastText** models under different settings on the processed dataset.
- Q3: Model Evaluation. Using these trained models, you are required to find the synonyms of the given tokens.
- Q4: Model Comparison. You will compare your models with some other models that are pre-trained on larger datasets.
- Q5: Non-Coding Questions. You can check your understandings about language models and word embeddings by answering these questions.