# Natural Language Processing for Law and Social Science

6. Word Embeddings

# Outline

# What is an Embedding?

- In a broad sense, **"embedding"** refers to a lower-dimensional dense vector representation of a higher-dimensional object.
    - in NLP, this higher-dimensional object will be a document.

# What is an Embedding?

- In a broad sense, **“embedding”** refers to a lower-dimensional dense vector representation of a higher-dimensional object.
  - in NLP, this higher-dimensional object will be a document.
- Not embeddings:
  - counts over LIWC dictionary categories.
  - sklearn CountVectorizer count vectors

# What is an Embedding?

- In a broad sense, **“embedding”** refers to a lower-dimensional dense vector representation of a higher-dimensional object.
  - in NLP, this higher-dimensional object will be a document.
- Not embeddings:
  - counts over LIWC dictionary categories.
  - sklearn CountVectorizer count vectors
- Embeddings:
  - PCA reductions of the word count vectors
  - LDA topic shares
  - compressed encodings from an autoencoder

## Categorical Embeddings = dense representations of categorical variables

Say we have a binary classification problem with outcome $Y$:

- we have a high-dimensonal categorical variable (e.g. area of law with 1000 categories)
- including dummy variables $A$ for each category in your ML model is computationally expensive.

# Categorical Embeddings = dense representations of categorical variables

Say we have a binary classification problem with outcome $Y$:

- we have a high-dimensonal categorical variable (e.g. area of law with 1000 categories)
- including dummy variables $A$ for each category in your ML model is computationally expensive.

Embedding approaches:

1. PCA applied to the dummy variables $A$ to get lower-dimensional $\tilde{A}$.

# Categorical Embeddings = dense representations of categorical variables

Say we have a binary classification problem with outcome $Y$:

- we have a high-dimensonal categorical variable (e.g. area of law with 1000 categories)
- including dummy variables $A$ for each category in your ML model is computationally expensive.

Embedding approaches:

1. PCA applied to the dummy variables $A$ to get lower-dimensional $\tilde{A}$.
2. Regress $Y$ on $A$, predict $\hat{Y}(A_i)$, add that as a predictor in your model instead.

# Categorical Embeddings = dense representations of categorical variables

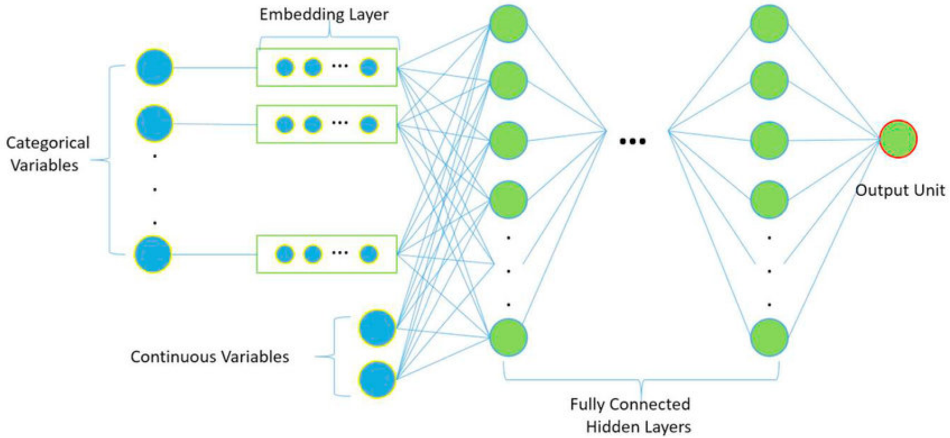
Say we have a binary classification problem with outcome $Y$:

- we have a high-dimensonal categorical variable (e.g. area of law with 1000 categories)
- including dummy variables $A$ for each category in your ML model is computationally expensive.

Embedding approaches:

1. PCA applied to the dummy variables $A$ to get lower-dimensional $\tilde{A}$.
2. Regress $Y$ on $A$, predict $\hat{Y}(A_i)$, add that as a predictor in your model instead.

**(2) is quite close to what embedding layers do in neural nets.**




An embedding layer is matrix multiplication:

$$\underbrace{h_1}_{n_E \times 1} = \underbrace{\omega_E}_{n_E \times n_w} \cdot \underbrace{x}_{n_x \times 1}$$

- $x$ = a categorical variable (e.g., representing a word)
  - one-hot vector with a single item equaling one. Input to the embedding layer.
- $h_1$ = the first hidden layer of the neural net
  - The output of the embedding layer.

## The embedding matrix $\omega_E$ encodes predictive information about the categories, has a spatial interpretation when projected to two dimensions.

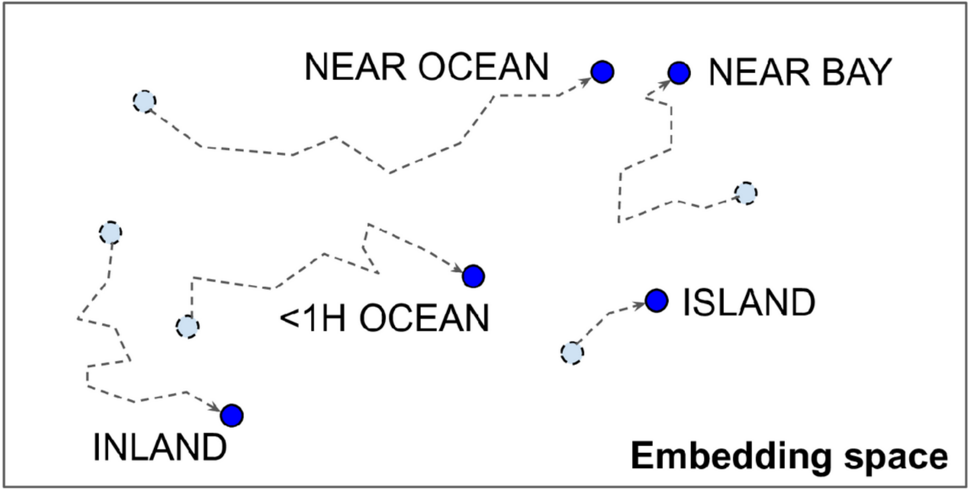



*Figure 13-4. Embeddings will gradually improve during training*

# Embedding Layers versus Dense Layers

- An embedding layer is statistically equivalent to a fully-connected dense layer with one-hot vectors as input and linear activation.
  - embedding layers are much faster for many categories ($>\sim 50$)

# Outline

## Word Embeddings = NN layers that map word indexes to dense vectors.

# Word Embeddings = NN layers that map word indexes to dense vectors.

- Documents are lists of word indexes $\{w_1, w_2, ..., w_{n_i}\}$.
  - equivalently, let $w_i$ be a one-hot vector (dimensionality $n_w$ = vocab size) where the associated word's index equals one.

# Word Embeddings = NN layers that map word indexes to dense vectors.

- Documents are lists of word indexes $\{w_1, w_2, ..., w_{n_i}\}$.
  - equivalently, let $w_i$ be a one-hot vector (dimensionality $n_w$ = vocab size) where the associated word's index equals one.
  - Normalize all documents to the same length $L$; shorter documents can be padded with a null token. This requirement can be relaxed with recurrent neural networks.

# Word Embeddings = NN layers that map word indexes to dense vectors.

- Documents are lists of word indexes $\{w_1, w_2, ..., w_{n_i}\}$.
  - equivalently, let $w_i$ be a one-hot vector (dimensionality $n_w$ = vocab size) where the associated word's index equals one.
  - Normalize all documents to the same length $L$; shorter documents can be padded with a null token. This requirement can be relaxed with recurrent neural networks.
- The embedding layer replaces the list of sparse one-hot vectors with a list of $n_E$-dimensional ($n_E << n_w$) dense vectors

$$\boldsymbol{X} = \begin{bmatrix} x_1 & \dots & x_L \end{bmatrix}$$

  where

$$\underbrace{x_j}_{n_E \times 1} = \underbrace{\boldsymbol{E}}_{n_E \times n_w} \cdot \underbrace{w_j}_{n_w \times 1}$$

  - $\boldsymbol{E}$ is a matrix of word vectors. The column associated with the word at $j$ is selected by the dot-product with one-hot vector $w_j$.

# Word Embeddings = NN layers that map word indexes to dense vectors.

- Documents are lists of word indexes $\{w_1, w_2, ..., w_{n_i}\}$.
  - equivalently, let $w_i$ be a one-hot vector (dimensionality $n_w$ = vocab size) where the associated word's index equals one.
  - Normalize all documents to the same length $L$; shorter documents can be padded with a null token. This requirement can be relaxed with recurrent neural networks.
- The embedding layer replaces the list of sparse one-hot vectors with a list of $n_E$-dimensional ($n_E << n_w$) dense vectors

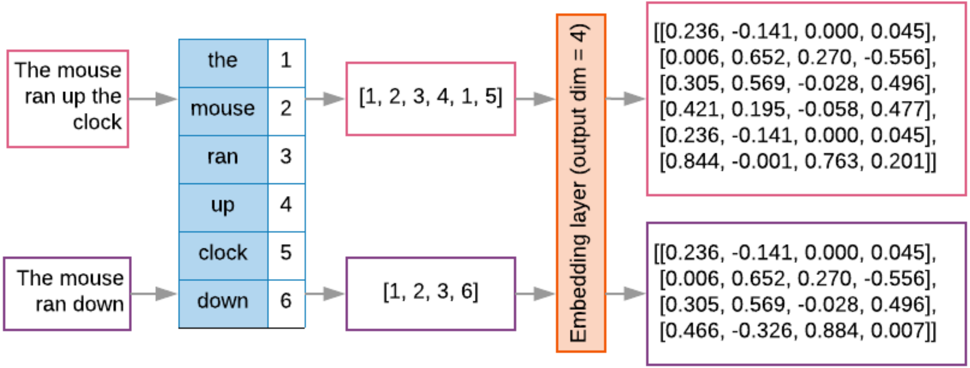
$$\boldsymbol{X} = \begin{bmatrix} x_1 & \ldots & x_L \end{bmatrix}$$

where

$$\underbrace{x_j}_{n_E \times 1} = \underbrace{\boldsymbol{E}}_{n_E \times n_w} \cdot \underbrace{w_j}_{n_w \times 1}$$

  - $\boldsymbol{E}$ is a matrix of word vectors. The column associated with the word at $j$ is selected by the dot-product with one-hot vector $w_j$.
- $\boldsymbol{X}$ is flattened into an $L * n_E$ vector for input to the next layer.

# Illustration



# Word2Vec & GloVe

- "Word embeddings" often refer to Word2Vec or GloVe – these are particular (popular) models for producing word embeddings.
  - the goal: represent the meaning of words by the neighboring words – their **contexts**.

# Word2Vec & GloVe

- “Word embeddings” often refer to Word2Vec or GloVe – these are particular (popular) models for producing word embeddings.
  - the goal: represent the meaning of words by the neighboring words – their **contexts**.
  - rather than predicting some metadata (such as classifying topic labels) they predict the co-occurence of neighboring words.

# Word2Vec & GloVe

- "Word embeddings" often refer to Word2Vec or GloVe – these are particular (popular) models for producing word embeddings.
  - the goal: represent the meaning of words by the neighboring words – their **contexts**.
  - rather than predicting some metadata (such as classifying topic labels) they predict the co-occurence of neighboring words.
- "You shall know a word by the company it keeps":
  - "He filled the **wampimuk**, passed it around and we all drunk some."
  - "We found a little, hairy **wampimuk** sleeping behind the tree."

# Words and Contexts

A long line of NLP research aims to capture the distributional properties of words using a **word-context matrix** $\boldsymbol{M}$:

- each row $w$ represents a **word** (e.g. "income"), each column $c$ represents a linguistic **context** in which words can occur (e.g. "corporate ___ tax").
  - A matrix entry $\boldsymbol{M}_{[w,c]}$ quantifies the strength of association between a word and a context in a large corpus.

# Words and Contexts

A long line of NLP research aims to capture the distributional properties of words using a **word-context matrix** $\boldsymbol{M}$:

- each row $w$ represents a **word** (e.g. "income"), each column $c$ represents a linguistic **context** in which words can occur (e.g. "corporate ___ tax").
  - A matrix entry $\boldsymbol{M}_{[w,c]}$ quantifies the strength of association between a word and a context in a large corpus.
- each word (row) $\boldsymbol{M}_{[w,:]}$ gives a distribution over contexts.
  - different definitions of contexts and different measures of association → different types of **word vectors**.
  - these vectors often have a **spatial interpretation** → geometric distances between word vectors reflect semantic distances between words.

# Defining the context

- The simplest definition of context is neighboring words:
  - for "the tabby **cat**": we get $(w = \text{"cat"}, c = \text{"tabby"})$

# Defining the context

- The simplest definition of context is neighboring words:
  - for "the tabby **cat**": we get $(w = \text{"cat"}, c = \text{"tabby"})$
- Could extend this to words within a window of two:
  - add $(w = \text{"cat"}, c = \text{"the"})$
  - etc.

# Defining the context

- The simplest definition of context is neighboring words:
  - for "the tabby **cat**": we get $(w = \text{"cat"}, c = \text{"tabby"})$
- Could extend this to words within a window of two:
  - add $(w = \text{"cat"}, c = \text{"the"})$
  - etc.
- Popular embeddings (word2vec and glove) generally use 5- or 10-word windows.

# Defining the context

- The simplest definition of context is neighboring words:
  - for "the tabby **cat**": we get $(w = \text{"cat"}, c = \text{"tabby"})$
- Could extend this to words within a window of two:
  - add $(w = \text{"cat"}, c = \text{"the"})$
  - etc.
- Popular embeddings (word2vec and glove) generally use 5- or 10-word windows.
- Context doesn't have to be single words:
  - could be the tuple of the preceding and the subsequent word.
  - Could include all words in the same sentence.

# Defining the context

- The simplest definition of context is neighboring words:
  - for "the tabby **cat**": we get ($w =$ "cat", $c =$ "tabby")
- Could extend this to words within a window of two:
  - add ($w =$ "cat", $c =$ "the")
  - etc.
- Popular embeddings (word2vec and glove) generally use 5- or 10-word windows.
- Context doesn't have to be single words:
  - could be the tuple of the preceding and the subsequent word.
  - Could include all words in the same sentence.
  - or same paragraph
  - or nouns in the same sentence
  - or syntactically connected words (from the parse tree)

# Defining the context

- The simplest definition of context is neighboring words:
  - for “the tabby **cat**”: we get $(w = \text{"cat''}, c = \text{"tabby"})$
- Could extend this to words within a window of two:
  - add $(w = \text{"cat''}, c = \text{"the"})$
  - etc.
- Popular embeddings (word2vec and glove) generally use 5- or 10-word windows.
- Context doesn’t have to be single words:
  - could be the tuple of the preceding and the subsequent word.
  - Could include all words in the same sentence.
  - or same paragraph
  - or nouns in the same sentence
  - or syntactically connected words (from the parse tree)
  - ...
- Etc.

# Defining an Association Measure

- Let $\boldsymbol{M}_{[w,c]} = f_M(w, c)$ where $w$ and $c$ are lookups to words in the $w$ vocabulary and $c$ vocabulary.
  - "word" could also mean phrases or more complicated objects.

# Defining an Association Measure

- Let $\boldsymbol{M}_{[w,c]} = f_M(w,c)$ where $w$ and $c$ are lookups to words in the $w$ vocabulary and $c$ vocabulary.
  - "word" could also mean phrases or more complicated objects.
- e.g. **counts**: $f_M(w,c) = \#(w,c)$, the number of times $w$ appeared along with context $c$, or **document frequencies**: $f_M(w,c) = \frac{\#(w,c)}{n_D}$
  - puts high weight on common contexts shared across many words (e.g., "the cat" will be weighted higher than "tabby cat")

# Defining an Association Measure

- Let $\boldsymbol{M}_{[w,c]} = f_M(w,c)$ where $w$ and $c$ are lookups to words in the $w$ vocabulary and $c$ vocabulary.
  - "word" could also mean phrases or more complicated objects.
- e.g. **counts**: $f_M(w,c) = \#(w,c)$, the number of times $w$ appeared along with context $c$, or **document frequencies**: $f_M(w,c) = \frac{\#(w,c)}{n_D}$
  - puts high weight on common contexts shared across many words (e.g., "the cat" will be weighted higher than "tabby cat")
- Better: **Point-wise mutual information** (PMI):

$$f_M(w,c) = \frac{\Pr(w,c)}{\Pr(w)\Pr(c)} = \frac{\frac{\#(w,c)}{n_D}}{\frac{\#(w)}{n_D}\frac{\#(c)}{n_D}} = \frac{n_D\#(w,c)}{\#(w)\#(c)}$$

  where $\#(w)$ and $\#(c)$ are the corpus counts for $w$ and $c$, respectively.
  - as noted in Week 2, PMI assigns high value to rare word-context pairs $\rightarrow$ impose a minimum count threshold on $(w,c)$ pairs; below the threshold, set to zero.

# $\boldsymbol{M}$ is too high-dimensional

- $\boldsymbol{M}$ is $n_w \times n_c$
  - if $c$ is drawn from from the vocabulary of a reasonably large corpus, the associated word vectors $\{v_1 = \boldsymbol{M}_{[w_1,:]}, v_2 = \boldsymbol{M}_{[w_2,:]}, \ldots\}$ are too high-dimensional to be useful.

# $\boldsymbol{M}$ is too high-dimensional

- $\boldsymbol{M}$ is $n_w \times n_c$
  - if $c$ is drawn from from the vocabulary of a reasonably large corpus, the associated word vectors $\{v_1 = \boldsymbol{M}_{[w_1,:]}, v_2 = \boldsymbol{M}_{[w_2,:]}, \ldots\}$ are too high-dimensional to be useful.
- Going back to dimension reduction: can use singular value decomposition (SVD):
  - factorize $\boldsymbol{M} \in \mathbb{R}^{n_w \times n_c}$ into a word matrix $\boldsymbol{W} \in \mathbb{R}^{n_w \times n_E}$ and context matrix $\boldsymbol{C} \in \mathbb{R}^{n_c \times n_E}$
  - such that $\tilde{\boldsymbol{M}} = \boldsymbol{W}\boldsymbol{C}'$ is the best rank-$n_E$ approximation of $\boldsymbol{M}$.

# $\boldsymbol{M}$ is too high-dimensional

- $\boldsymbol{M}$ is $n_w \times n_c$
  - if $c$ is drawn from from the vocabulary of a reasonably large corpus, the associated word vectors $\{v_1 = \boldsymbol{M}_{[w_1,:]}, v_2 = \boldsymbol{M}_{[w_2,:]}, \ldots\}$ are too high-dimensional to be useful.
- Going back to dimension reduction: can use singular value decomposition (SVD):
  - factorize $\boldsymbol{M} \in \mathbb{R}^{n_w \times n_c}$ into a word matrix $\boldsymbol{W} \in \mathbb{R}^{n_w \times n_E}$ and context matrix $\boldsymbol{C} \in \mathbb{R}^{n_c \times n_E}$
  - such that $\tilde{\boldsymbol{M}} = \boldsymbol{W}\boldsymbol{C}'$ is the best rank-$n_E$ approximation of $\boldsymbol{M}$.
  - $\tilde{\boldsymbol{M}}$ can be seen as a "smoothed" version of $\boldsymbol{M}$; "missing" values are filled in, etc.

# $\boldsymbol{M}$ is too high-dimensional

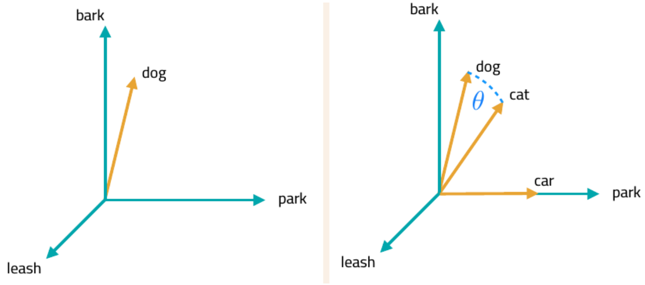
- $\boldsymbol{M}$ is $n_w \times n_c$
  - if $c$ is drawn from from the vocabulary of a reasonably large corpus, the associated word vectors $\{v_1 = \boldsymbol{M}_{[w_1,:]}, v_2 = \boldsymbol{M}_{[w_2,:]}, \ldots\}$ are too high-dimensional to be useful.
- Going back to dimension reduction: can use singular value decomposition (SVD):
  - factorize $\boldsymbol{M} \in \mathbb{R}^{n_w \times n_c}$ into a word matrix $\boldsymbol{W} \in \mathbb{R}^{n_w \times n_E}$ and context matrix $\boldsymbol{C} \in \mathbb{R}^{n_c \times n_E}$
  - such that $\tilde{\boldsymbol{M}} = \boldsymbol{W}\boldsymbol{C}'$ is the best rank-$n_E$ approximation of $\boldsymbol{M}$.
  - $\tilde{\boldsymbol{M}}$ can be seen as a "smoothed" version of $\boldsymbol{M}$; "missing" values are filled in, etc.
- $\boldsymbol{W}$ is the matrix of word vectors (word embeddings):
  - relatively low-dimensional ($n_E << n_w$, typically between 50 and 300)
  - dense, rather than sparse.

# $\boldsymbol{M}$ is too high-dimensional

- $\boldsymbol{M}$ is $n_w \times n_c$
  - if $c$ is drawn from from the vocabulary of a reasonably large corpus, the associated word vectors $\{v_1 = \boldsymbol{M}_{[w_1,:]}, v_2 = \boldsymbol{M}_{[w_2,:]}, \ldots\}$ are too high-dimensional to be useful.
- Going back to dimension reduction: can use singular value decomposition (SVD):
  - factorize $\boldsymbol{M} \in \mathbb{R}^{n_w \times n_c}$ into a word matrix $\boldsymbol{W} \in \mathbb{R}^{n_w \times n_E}$ and context matrix $\boldsymbol{C} \in \mathbb{R}^{n_c \times n_E}$
  - such that $\tilde{\boldsymbol{M}} = \boldsymbol{W}\boldsymbol{C}'$ is the best rank-$n_E$ approximation of $\boldsymbol{M}$.
  - $\tilde{\boldsymbol{M}}$ can be seen as a "smoothed" version of $\boldsymbol{M}$; "missing" values are filled in, etc.
- $\boldsymbol{W}$ is the matrix of word vectors (word embeddings):
  - relatively low-dimensional ($n_E << n_w$, typically between 50 and 300)
  - dense, rather than sparse.
- **similarity measures between rows of $\boldsymbol{W}$ approximate similarity measures between rows of $\boldsymbol{M}$**

# Word Similarity

- Once words are represented as vectors $\{v_1 = \boldsymbol{M}_{[w_1,:]},\ v_2 = \boldsymbol{M}_{[w_2,:]}, \ldots\}$, we can use linear algebra to understand the relationships between words:
  - Words that are geometrically close to each other are similar: e.g. "dog" and "cat":



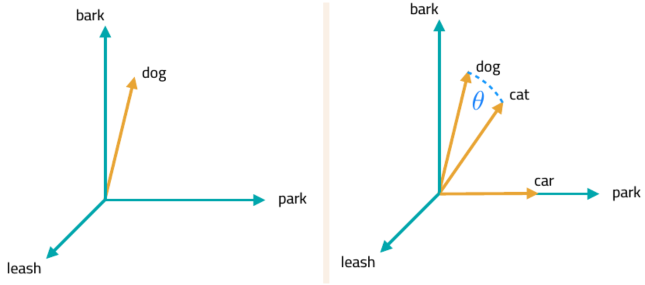

- The standard metric for comparing vectors is cosine similarity:

$$\cos\theta = \frac{v_1 \cdot v_2}{||v_1||||v_2||}$$

  - alternatives include e.g. Jaccard similarity (Goldberg 2017)

# Word Similarity

- Once words are represented as vectors $\{v_1 = \boldsymbol{M}_{[w_1,:]},\ v_2 = \boldsymbol{M}_{[w_2,:]}, \ldots\}$, we can use linear algebra to understand the relationships between words:
  - Words that are geometrically close to each other are similar: e.g. "dog" and "cat":




- The standard metric for comparing vectors is cosine similarity:

$$\cos\theta = \frac{v_1 \cdot v_2}{||v_1||\,||v_2||}$$

  - alternatives include e.g. Jaccard similarity (Goldberg 2017)
- Thanks to linearity, can compute similarities between groups of words by averaging the groups.

# Word2Vec

- When people mention “word2vec”, they are usually talking about a particular word-embedding model with good performance on a range of analogy and prediction tasks.

# Word2Vec

- When people mention "word2vec", they are usually talking about a particular word-embedding model with good performance on a range of analogy and prediction tasks.
- How does it learn the meaning of the word "fox"?
  - By comparing true instances of the word fox ("The quick brown **fox** jumps over the lazy dog")
  - to fake (randomly sampled) ones ("The prescription of **fox** is advised for this diagnosis")

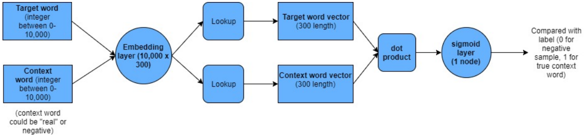
# Word2Vec

- When people mention "word2vec", they are usually talking about a particular word-embedding model with good performance on a range of analogy and prediction tasks.
- How does it learn the meaning of the word "fox"?
  - By comparing true instances of the word fox ("The quick brown **fox** jumps over the lazy dog")
  - to fake (randomly sampled) ones ("The prescription of **fox** is advised for this diagnosis")
- Word2Vec learns embedding vectors for the target word ("fox") and context words (neighbors of "fox") to distinguish true from false samples.

# Word2Vec Negative Sampling Objective

The dataset is a collection of context pairs indexed by $i$:

- $y_i = 1$ means correct (it appeared in the corpus)
- $y_i = 0$ means incorrect (it was randomly drawn $\rightarrow$ **negative sample**).




- Both words are looked up in the same embedding matrix.
- The concatened embeddings $[\boldsymbol{w}; \boldsymbol{c}]$ are input to a dense layer (no activation) then to sigmoid output:

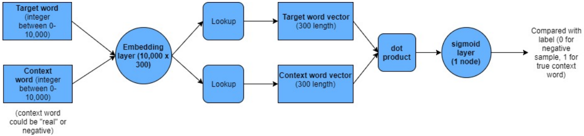
$$\hat{y}(w, c) = \text{sigmoid}(([\boldsymbol{w}; \boldsymbol{c}] \cdot \omega_0) \cdot \omega_1)$$

which gives the predicted probability of a correct rather than random pair.

# Word2Vec Negative Sampling Objective

The dataset is a collection of context pairs indexed by $i$:

- $y_i = 1$ means correct (it appeared in the corpus)
- $y_i = 0$ means incorrect (it was randomly drawn $\rightarrow$ **negative sample**).




- Both words are looked up in the same embedding matrix.
- The concatened embeddings $[\boldsymbol{w}; \boldsymbol{c}]$ are input to a dense layer (no activation) then to sigmoid output:

$$\hat{y}(w, c) = \text{sigmoid}(([\boldsymbol{w}; \boldsymbol{c}] \cdot \omega_0) \cdot \omega_1)$$

which gives the predicted probability of a correct rather than random pair.

- Word2Vec minimizes the binary cross-entropy

$$L(\theta) = -\sum_{i=1}^{n_D} [y_i \log \hat{y}_i(w, c; \theta) + [1 - y_i] \log(1 - \hat{y}_i(w, c; \theta))]$$

# How does Word2Vec relate to the $\boldsymbol{M}$ matrix?

- Word2Vec produces embedding matrices $\boldsymbol{W}$ and $\boldsymbol{C}$.
  - generally, context embeddings are discarded after training.
- Levy and Goldberg (2014):
  - If we take $\tilde{\boldsymbol{M}} = \boldsymbol{W}\boldsymbol{C}'$, word2vec is equivalent to factorizing a matrix $\boldsymbol{M}$ with items

    $$\boldsymbol{M}_{[w,c]} = \mathrm{PMI}(w,c) - \log a$$

    where $a$ is a constant calibrating the amount of negative sampling.

# GloVe Embeddings

- Pennington et al (2014) (GloVe = Global Vectors) take a different approach:
  - that does not require a neural net
- Input: $C_{ij}$ = local co-occurrence counts between words $i, j \in \{1, ..., n_w\}$ within some co-occurence window, e.g. ten words.

# GloVe Embeddings

- Pennington et al (2014) (GloVe = Global Vectors) take a different approach:
  - that does not require a neural net
- Input: $C_{ij}$ = local co-occurrence counts between words $i, j \in \{1, ..., n_w\}$ within some co-occurence window, e.g. ten words.

Learn word vectors $\boldsymbol{w} = (w_1, ..., w_i, ..., w_{n_w})$, where $w_i \in (-1, 1)^{n_E}$, to solve

$$\min_{\boldsymbol{w}} \sum_{i,j} f\left(C_{ij}\right)\left(w_i^T w_j - \log\left(C_{ij}\right)\right)^2$$

where $f(\cdot)$ is weighting function to down-weight frequent words.

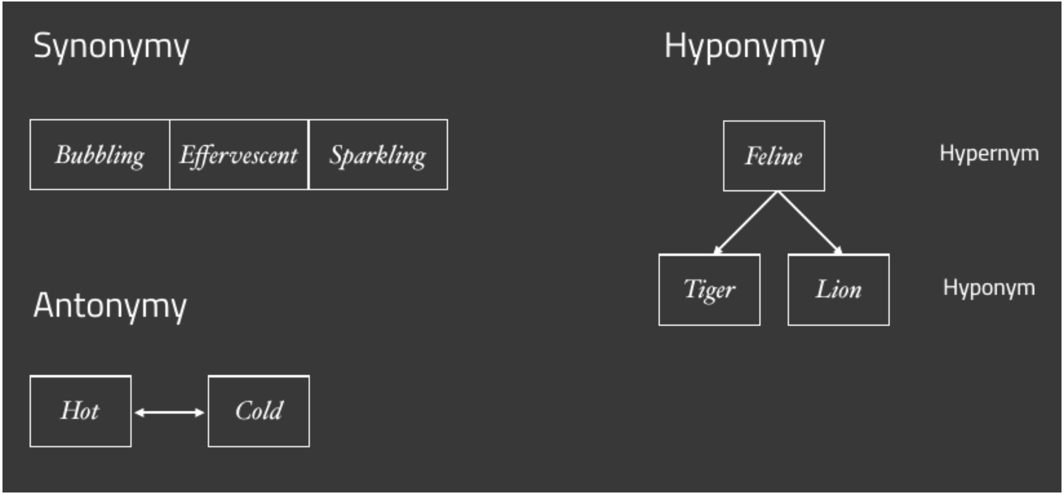
- Minimizes **squared difference** between**:**
  - **dot product of word vectors,** $w_i^T w_j$
  - **empirical co-occurrence,** $\log\left(C_{ij}\right)$
    [Arora et al (2016) put the PMI here instead of co-occurence counts]
- Intuitively: words that co-occur should have high correlation (dot product)

# Check for Understanding

1. What is the difference/connection between an embedding layer and a word embedding?
2. Why use PMI instead of co-occurence frequencies when constructing the word association matrix?
3. What does negative sampling mean in general, and in the case of Word2Vec?
4. What are the main differences between Word2Vec and GloVe?

# Word Embeddings Encode Linguistic Relations

# Word Embeddings Encode Linguistic Relations



# Similarity vs. Relatedness (Budansky and Hirst, 2006)

- Semantic **similarity**: words sharing salient attributes / features
  - synonymy (car / automobile)
  - hypernymy (car / vehicle)
  - co-hyponymy (car / van / truck)

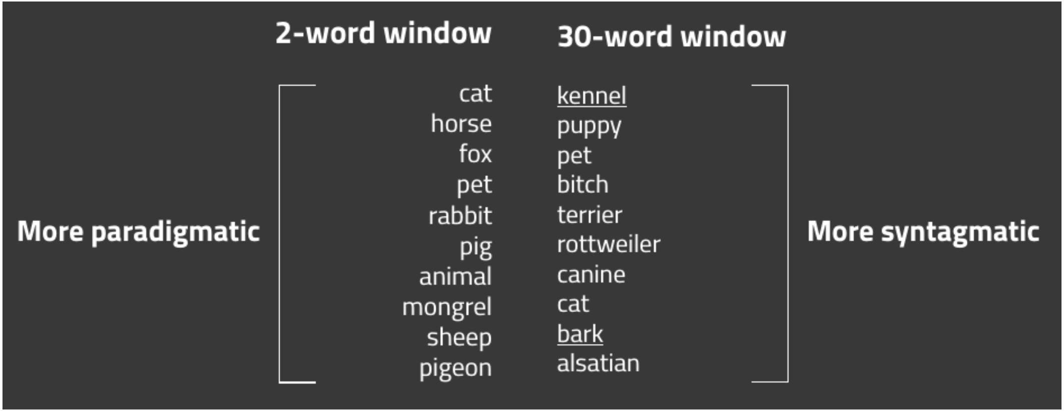
# Similarity vs. Relatedness (Budansky and Hirst, 2006)

- Semantic **similarity**: words sharing salient attributes / features
  - synonymy (car / automobile)
  - hypernymy (car / vehicle)
  - co-hyponymy (car / van / truck)
- Semantic **relatedness**: words semantically associated without necessarily being similar
  - function (car / drive)
  - meronymy (car / tire)
  - location (car / road)
  - attribute (car / fast)

# Similarity vs. Relatedness (Budansky and Hirst, 2006)

- Semantic **similarity**: words sharing salient attributes / features
  - synonymy (car / automobile)
  - hypernymy (car / vehicle)
  - co-hyponymy (car / van / truck)
- Semantic **relatedness**: words semantically associated without necessarily being similar
  - function (car / drive)
  - meronymy (car / tire)
  - location (car / road)
  - attribute (car / fast)
- Word embeddings will recover one or both of these relations, depending on how contexts and associated are constructed.

# Most similar words to dog, depending on context window size




- Small windows pick up substitutable words; large windows pick up topics.

# Parts of Speech and Phrases

- In the default model multiple senses of a word are merged.
  - e.g. “I like a bird” (verb) and “I am like a bird” (preposition).

# Parts of Speech and Phrases

- In the default model multiple senses of a word are merged.
  - e.g. "I like a bird" (verb) and "I am like a bird" (preposition).
- Can improve the quality of embeddings in these cases by attaching the POS to the word (e.g. "like:verb", "like:prep") before training.

# Parts of Speech and Phrases

- In the default model multiple senses of a word are merged.
  - e.g. "I like a bird" (verb) and "I am like a bird" (preposition).
- Can improve the quality of embeddings in these cases by attaching the POS to the word (e.g. "like:verb", "like:prep") before training.
- The default model only works by word, but "new york $\neq$ "new" + "york"
  - can tokenize phrases together (see Week 2 lecture) before training.

# The black sheep problem

- The trivial or obvious features of a word are not mentioned in standard corpora.

# The black sheep problem

- The trivial or obvious features of a word are not mentioned in standard corpora.
- For example, although most sheep are white, you rarely see the phrase "white sheep".
  - so word2vec tells you sim(black,sheep) $>$ sim(white,sheep).

# The black sheep problem

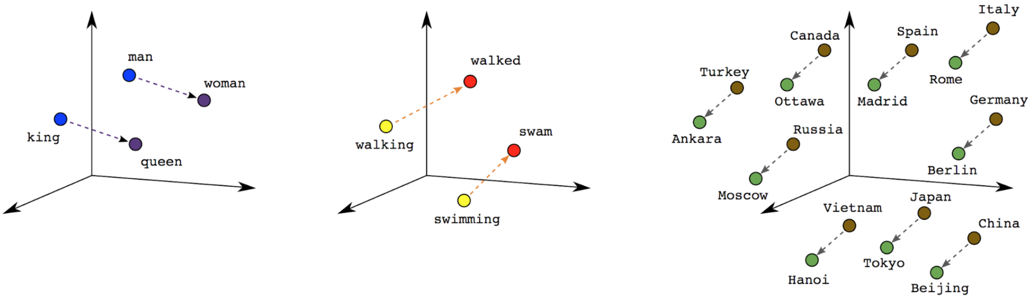
- The trivial or obvious features of a word are not mentioned in standard corpora.
- For example, although most sheep are white, you rarely see the phrase "white sheep".
  - so word2vec tells you sim(black,sheep) $>$ sim(white,sheep).
- This is really important when we will use embeddings to anayze beliefs/attitudes.
  - And I don't see a solution to it.

# The black sheep problem

- The trivial or obvious features of a word are not mentioned in standard corpora.
- For example, although most sheep are white, you rarely see the phrase "white sheep".
  - so word2vec tells you sim(black,sheep) $>$ sim(white,sheep).
- This is really important when we will use embeddings to anayze beliefs/attitudes.
  - And I don't see a solution to it.
- Relatedly, antonyms are often rated similarly, have to be careful with that.

# Vector Directions $\leftrightarrow$ Meaning

- Intriguingly, word2vec algebra can depict conceptual, analogical relationships between words:




Male-Female    Verb Tense    Country-Capital

# Word Embeddings for Analogies

$$vec(king) - vec(man) + vec(woman) \approx vec(queen)$$

# Word Embeddings for Analogies

$$vec(king) - vec(man) + vec(woman) \approx vec(queen)$$

- More generally: The analogy $a_1 : b_1 :: a_2 : b_2$ can be solved (that is, find $b_2$ given $a_1, b_1, a_2$) by

$$\arg\max_{b_2 \in V} \cos(b_2, a_2 - a_1 + b_1)$$

  where $V$ excludes $(a_1, b_1, a_2)$.

# Word Embeddings for Analogies

$$vec(king) - vec(man) + vec(woman) \approx vec(queen)$$

- More generally: The analogy $a_1 : b_1 :: a_2 : b_2$ can be solved (that is, find $b_2$ given $a_1, b_1, a_2$) by

$$\arg\max_{b_2 \in V} \cos(b_2, a_2 - a_1 + b_1)$$

  where $V$ excludes $(a_1, b_1, a_2)$.
- Often works better with normalized vectors (so that one long vector doesn't wash out the others)

# Word Embeddings for Analogies

$$vec(king) - vec(man) + vec(woman) \approx vec(queen)$$

- More generally: The analogy $a_1 : b_1 :: a_2 : b_2$ can be solved (that is, find $b_2$ given $a_1, b_1, a_2$) by

$$\arg\max_{b_2 \in V} \cos(b_2, a_2 - a_1 + b_1)$$

  where $V$ excludes $(a_1, b_1, a_2)$.
- Often works better with normalized vectors (so that one long vector doesn't wash out the others)
- Levy and Goldberg (2014) recommend the following "CosMul" metric which tends to perform better:

$$\arg\max_{b_2 \in V} \frac{\cos(b_2, a_2)\cos(b_2, b_1)}{\cos(b_2, a_1) + \epsilon}$$

  - requires normalized, non-negative vectors (can transform using $(x + 1)/2$)
  - $\epsilon$ is a small smoothing parameter.

# Tokenizing for Word Embeddings

- drop capitalization
- punctuation is optional
- don't drop stopwords/function-words
- add special tokens for start of sentence and end of sentence
- for out-of-vocab words, substitute a special token or replace with part-of-speech tag

# Can cluster word embeddings to produce topics

| Cluster # | Top 10 Words |
|---|---|
| 174 | complicate, depend, crucial, illustrate, elusive, focus, important, straightforward, elide, critical |
| 134 | implausible, problematic, exaggeration, skeptical, ascribe, discredit, contradictory, weak, exaggerate, supportable |
| 75 | reverse, AFFIRM, affirm, vacate, reversed, REMANDED, forego, foregoing, forgoing, remands |
| 70 | importation, import, ecstasy, marihuana, illicit, opium, distilled, export, phencyclidine, narcotic |
| 178 | perverse, sensible, tempt, unlikely, unwise, anomalous, would, easy, costly, attractive |
| 32 | phrase, meaning, word, synonymous, language, interpret, noun, wording, verb, adjective |
| 169 | circumscribe, endow, unfettered, vest, unlimited, boundless, broad, constrain, exercise, unbounded |
| 85 | hundred, thousand, many, million, huge, massive, large, enormous, most, dozen |
| 28 | emphasis, bracket, alteration, citation, footnote, italic, ellipsis, petcitation, idcitation, punctuation |
| 138 | logo, symbol, stylized, imprint, emblem, grille, prefix, lettering, suffix, crosshair |
| 181 | wilful, carelessness, recklessness, careless, intentional, willful, conscious, reckless, unintentional, wantonness |
| 158 | rigorous, demanding, heightened, reasonableness, rigid, heighten, objective, deferential, flexible, particular |
| 55 | agreement, contract, contractual, promise, novation, repudiate, guaranty, enforceable, novate, repurchase |
| 197 | summation, admonish, sidebar, prosecutor, admonishment, mistrial, curative, questioning, remark, recess |
| 120 | scrivener, typographical, reversible, plain, harmless, clerical, invited, clear, requiresthe, instructional |
| 15 | adjudicatory, adjudicative, adversarial, judicial, rulemaking, decisionmaking, administrative, meaningful, rulemake, agency |

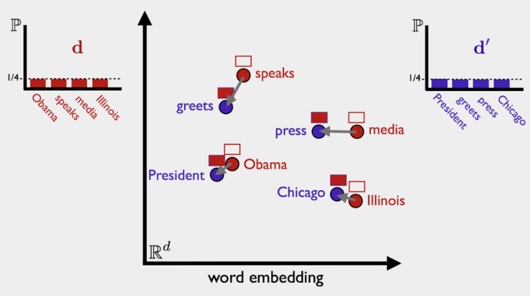
Clustered word embeddings in judicial opinions, from Ash and Nikolaus (2020)

# Word Mover Distance

- TF-IDF distance treats synonyms as just as close as totally unrelated words.

# Word Mover Distance

- TF-IDF distance treats synonyms as just as close as totally unrelated words.
- Word mover distance (Kusner, Sun, Kolkin, and Weinberger ICML 2015) between two texts is given by:
  - total amount of "mass" needed to move words from one side into the other
  - multiplied by the distance the words need to move
  - uses word embedding distance



# Pre-trained word embeddings

- In many settings (e.g. a small corpus), better to use pre-trained embeddings.
- e,g, spaCy's GloVe embeddings:
  - one million vocabulary entries
  - 300-dimensional vectors
  - trained on the Common Crawl corpus
- Can initialize models with pre-trained embeddings, can fine-tune as needed.

# “Enriching word vectors with subword information” (Bojanowski et al 2017)

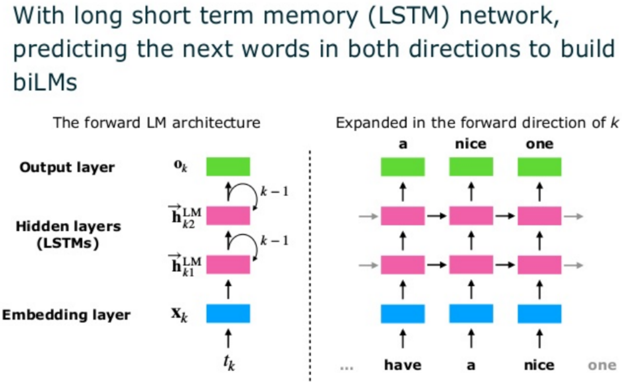
- each word is represented as a bag of (hashed) character n-grams. (e.g., spicy = (spi, pic, icy)).
- learn embeddings for the character segments, and construct word embedding by summing over the segment embeddings

# “Enriching word vectors with subword information” (Bojanowski et al 2017)

- each word is represented as a bag of (hashed) character n-grams. (e.g., spicy = (spi, pic, icy)).
- learn embeddings for the character segments, and construct word embedding by summing over the segment embeddings
- competitive with word2vec in standard tasks; better in some languages.
- produces good embeddings for unseen words.

# ELMo (Embedings from Language Models)

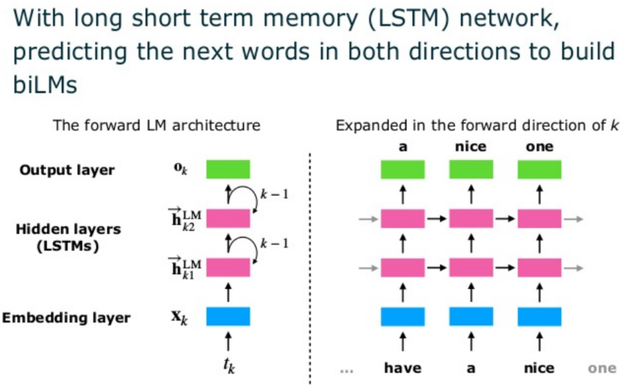
- ELMo is a context-sensitive word embedding model that uses the output of a bidirectional LSTM:




- The task:
  - predict previous and next words in a sentence using a birectional LSTM.

# ELMo (Embedings from Language Models)

- ELMo is a context-sensitive word embedding model that uses the output of a bidrectional LSTM:




- The task:
  - predict previous and next words in a sentence using a birectional LSTM.
- embeddings go through two hidden layers before the softmax output:
  - first layer learns syntax
  - second layer learns semantics

GloVe mostly learns *sport*-related context

| | Source | Nearest Neighbors |
|---|---|---|
| GloVe | play | playing, game, games, played, players, plays, player, Play, football, multiplayer |
| biLM | Chico Ruiz made a spectacular play on Alusik 's grounder {...} | Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play . |
| | Olivia De Havilland signed to do a Broadway play for Garson {...} | {...} they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement . |

Table 4: Nearest neighbors to "play" using GloVe and the context embeddings from a biLM.

ELMo can distinguish the word sense based on the context

- Pre-trained ELMo models are available from AllenNLP (allennlp.org/elmo)

# Check for Understanding

1. How would it affect my word embeddings to use co-occurence within paragraph, rather than within sentence?
2. How would it my embeddings to drop function words in a pre-processing step?
3. What is the black sheep problem in the context of word embeddings?
4. Think of a setting (and explain) where:
   - using pre-trained embeddings would not work.
   - using embeddings with subword information would help a lot
   - using elmo would work a lot better than glove.
   - you would care more about the first layer or the second layer from elmo.

Standard word embeddings (e.g. word2vec/glove) have a number of limitations:

- **polysemy**: you get one vector for multiple senses of a word
  (e.g. "**glass** of water" vs "window **glass**")

Standard word embeddings (e.g. word2vec/glove) have a number of limitations:

- **polysemy**: you get one vector for multiple senses of a word (e.g. "**glass** of water" vs "window **glass**")
- **rare words**: a word that shows up just once or twice won't be well-defined
- **n-grams**: does not produce embeddings for multi-word phrases

Scientists attending ACL work on **cutting edge** research in NLP

**Petrichor**: the earthy scent produce when rain falls on dry soil

Roger Federer won the first **set**$^{NN}$ of the match

Standard word embeddings (e.g. word2vec/glove) have a number of limitations:

- **polysemy**: you get one vector for multiple senses of a word (e.g. "**glass** of water" vs "window **glass**")
- **rare words**: a word that shows up just once or twice won't be well-defined
- **n-grams**: does not produce embeddings for multi-word phrases

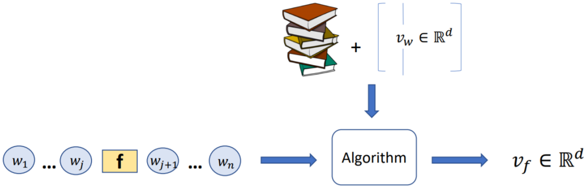
Scientists attending ACL work on **cutting edge** research in NLP

**Petrichor**: the earthy scent produce when rain falls on dry soil

Roger Federer won the first **set**$^{NN}$ of the match

- Goal of Khodak et al (2018): produce embeddings "a la carte" given a context:

Given: Text corpus and high quality word embeddings trained on it




Input: A feature in context(s)

Output: Good quality embedding for the feature

# A la carte embeddings

- Given a target word $f$ and its context $c$, define

$$v_f^{avg} = \frac{1}{|c|} \sum_{w \in c} v_w$$

  the average vector for the words in the context.

- Arora et al (2018) prove that for vectors produced by a generative language model, there exists a matrix $A$ such that

$$v_f \approx A v_f^{avg}$$

# A la carte embeddings

- Given a target word $f$ and its context $c$, define

$$v_f^{avg} = \frac{1}{|c|} \sum_{w \in c} v_w$$

the average vector for the words in the context.

- Arora et al (2018) prove that for vectors produced by a generative language model, there exists a matrix $A$ such that

$$v_f \approx A v_f^{avg}$$

- The "induction matrix" $A$ can be learned with a least-squares (linear regression) objective

$$A^* = \arg\min_A \sum_w |v_w - A v_w^{avg}|_2^2$$

where $w$ indexes over all the tokens in the corpus.

- empirically:

$$\text{cosine}(v_f, A^* v_f^{avg}) \geq 0.9$$

# Outline

# Outline

# Bias in NLP Systems

## Sentiment Analysis

```
text_to_sentiment("Let's go get Italian food")
2.0429166109
text_to_sentiment("Let's go get Chinese food")
1.4094033658
text_to_sentiment("Let's go get Mexican food")
0.3880198556
```

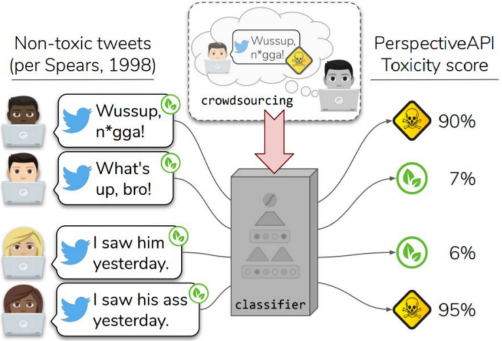
```
text_to_sentiment("My name is Emily")
2.2286179365
text_to_sentiment("My name is Heather")
1.3976291151
text_to_sentiment("My name is Yvette")
0.9846380213
text_to_sentiment("My name is Shaniqua")
-0.4704813178
```

**Is this sentiment model racist?**

Source: Kareem Carr slides.

# Bias in NLP Systems

## Toxicity Detection




Within dataset proportions

| Dataset | Group | Acc. | % false identification: None | % false identification: Offensive | % false identification: Hate |
|---|---|---|---|---|---|
| DWMW17 | AAE | 94.3 | 1.1 | **46.3** | 0.8 |
| | White | 87.5 | **7.9** | 9.0 | **3.8** |
| | Overall | 91.4 | 2.9 | 17.9 | 2.3 |

| Dataset | Group | Acc. | % false identification: None | % false identification: Abusive | % false identification: Hateful |
|---|---|---|---|---|---|
| FDCL18 | AAE | 81.4 | 4.2 | **26.0** | **1.7** |
| | White | 82.7 | **30.5** | 4.5 | 0.8 |
| | Overall | 81.4 | 20.9 | 6.6 | 0.8 |

**Is this toxicity detection model racist?**

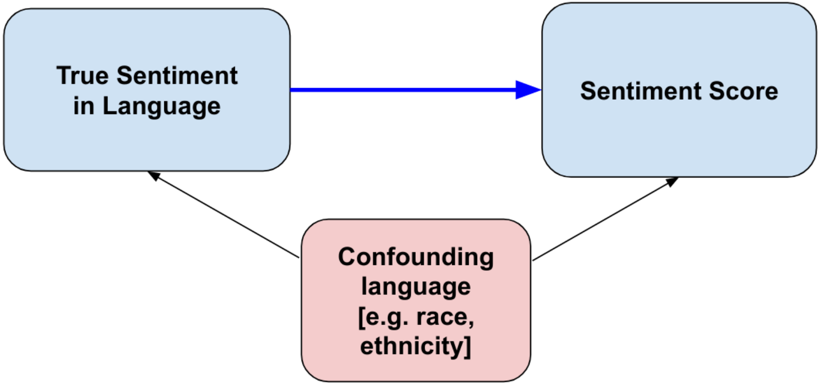
Source: Jacobs and Wallach slides.

# NLP "Bias" is statistical bias

- Sentiment scores that are trained on annotated datasets also learn from the correlated non-sentiment information.

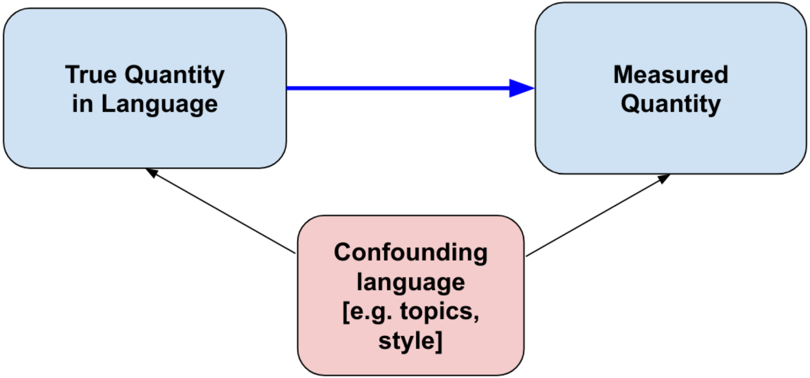
# NLP "Bias" is statistical bias

- Sentiment scores that are trained on annotated datasets also learn from the correlated non-sentiment information.



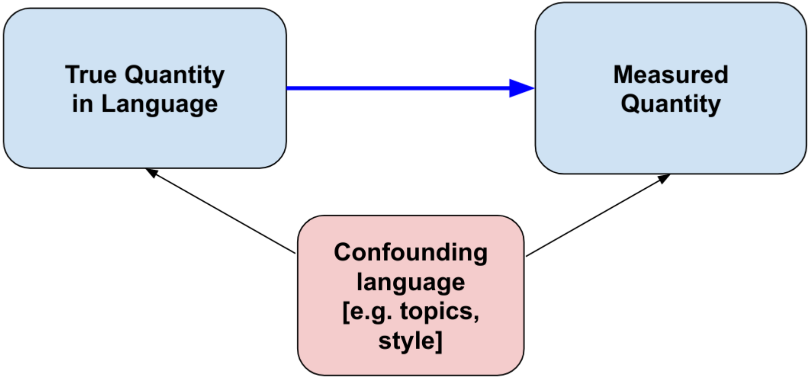

- Supervised sentiment models are confounded by correlated language factors.
  - e.g., in the training set maybe people complain about Mexican food more often than Italian food.

# This is a universal problem




- supervised models (classifiers, regressors) learn features that are correlated with the label being annotated.
- unsupervised models (topic models, word embeddings) learn correlations between topics / contexts.

# This is a universal problem




- supervised models (classifiers, regressors) learn features that are correlated with the label being annotated.
- unsupervised models (topic models, word embeddings) learn correlations between topics / contexts.
- An important exception: dictionary methods (perhaps explaining why they are often used by economists). But they have other serious limitations.

# Examples: Confounders in Measurement from Text

**What quantity do we care about? vs. What do we measure?**

# Examples: Confounders in Measurement from Text

**What quantity do we care about? vs. What do we measure?**

- Positive/negative sentiment $\rightarrow$ Count positive/negative words, or predict text annotations.
- Toxicity $\rightarrow$ Count toxic words, or predict text annotations.

**confounders?**

## Examples: Confounders in Measurement from Text

**What quantity do we care about? vs. What do we measure?**

- Positive/negative sentiment $\rightarrow$ Count positive/negative words, or predict text annotations.
- Toxicity $\rightarrow$ Count toxic words, or predict text annotations.

**confounders?**

- Student performance $\rightarrow$ predicted essay grade based on labeled essay documents.
- Credit worthiness $\rightarrow$ predicted probability of default based on loan application documents.

**confounders?**

# Examples: Confounders in Measurement from Text

**What quantity do we care about? vs. What do we measure?**

- Positive/negative sentiment $\rightarrow$ Count positive/negative words, or predict text annotations.
- Toxicity $\rightarrow$ Count toxic words, or predict text annotations.

**confounders?**

- Student performance $\rightarrow$ predicted essay grade based on labeled essay documents.
- Credit worthiness $\rightarrow$ predicted probability of default based on loan application documents.

**confounders?**

- Political partisanship $\rightarrow$ predicted probability being Democrat/Republican based on speeches.

**confounders?**

# Examples: Confounders in Measurement from Text

**What quantity do we care about? vs. What do we measure?**

- Positive/negative sentiment $\rightarrow$ Count positive/negative words, or predict text annotations.
- Toxicity $\rightarrow$ Count toxic words, or predict text annotations.

**confounders?**

- Student performance $\rightarrow$ predicted essay grade based on labeled essay documents.
- Credit worthiness $\rightarrow$ predicted probability of default based on loan application documents.

**confounders?**

- Political partisanship $\rightarrow$ predicted probability being Democrat/Republican based on speeches.

**confounders?**

- Policy priorities $\rightarrow$ predicted probability of speeches/laws being about a particular policy topic.

**confounders?**

# When is measurement confounding important?

- By itself, producing measurements that are biased by confounders might not be a problem.
- e.g.:
  - an NLP-based credit score that learns confounders $\rightarrow$ not a problem unless debtors learn about it and strategically alter their documents.
    - similarly with automated essay grading

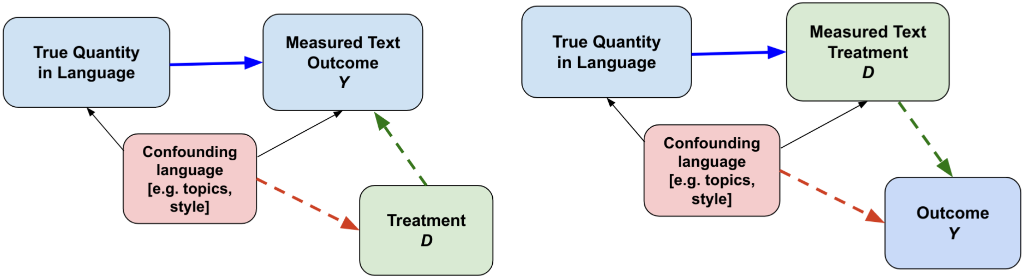
## When is measurement confounding important?

- By itself, producing measurements that are biased by confounders might not be a problem.
- e.g.:
  - an NLP-based credit score that learns confounders $\rightarrow$ not a problem unless debtors learn about it and strategically alter their documents.
    - similarly with automated essay grading
- for measuring political divisiveness or policy priorities
  - probably won't matter for in-domain summary statistics
  - but would matter a lot for summary statistics in a new domain

# When is measurement confounding important?

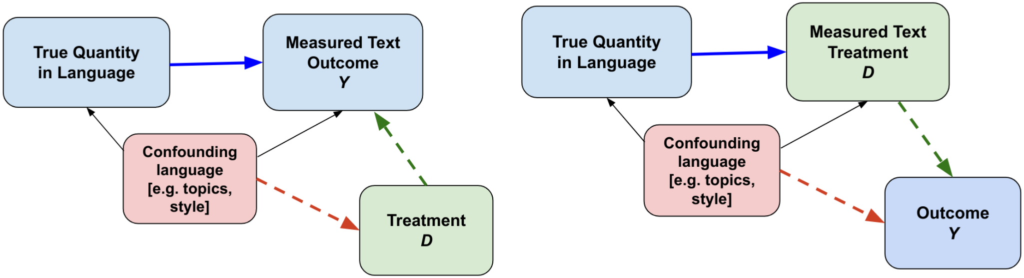
- By itself, producing measurements that are biased by confounders might not be a problem.
- e.g.:
  - an NLP-based credit score that learns confounders $\rightarrow$ not a problem unless debtors learn about it and strategically alter their documents.
    - similarly with automated essay grading
- for measuring political divisiveness or policy priorities
  - probably won't matter for in-domain summary statistics
  - but would matter a lot for summary statistics in a new domain
- even in domain, will matter for assessing the causal effect of a treatment, e.g. the electoral cycle:
  - elections might cause politicians to focus on social issues rather than economic issues,
  - if social/economic issues are confounded with partisanship, the resulting estimates are biased.

# When is measurement confounding important?




- When text is outcome, the confounders cannot be correlated with the treatment.
- When text is treatment, the confounders cannot be correlated with the outcome.

# When is measurement confounding important?




- When text is outcome, the confounders cannot be correlated with the treatment.
- When text is treatment, the confounders cannot be correlated with the outcome.
  - e.g.: estimating the effect of politician speech sentiment on his/her reelection chances?

# Steps for de-biasing

- Language features that are often confounded with an important quantity:
  - stopwords
  - person/organization/place names
- These can be dropped during pre-processing to reduce the influence of confounders in subsequent measurements.
- Can control for topic or style features or other potential confounders in regressions.

# Outline

# Research Objectives

1. **What is the research question?**
2. Corpus and Data.

# Research Objectives

1. **What is the research question?**
2. Corpus and Data.
3. Word Embeddings:
    - **What are we trying to measure?**

# Research Objectives

1. **What is the research question?**
2. Corpus and Data.
3. Word Embeddings:
   - **What are we trying to measure?**
   - Select a model and train it.
   - **Validate that the model is measuring what we want and that there are no clear confounders.**

# Research Objectives

1. **What is the research question?**
2. Corpus and Data.
3. Word Embeddings:
   - **What are we trying to measure?**
   - Select a model and train it.
   - **Validate that the model is measuring what we want and that there are no clear confounders.**
4. Empirical analysis
   - Produce statistics or predictions with the trained model.
   - **Answer the research question.**

# Implicit attitudes

"Attitudes that affect our understanding, actions, and decisions in an unconscious manner" (Kirnan institute, OSU)

# Implicit attitudes

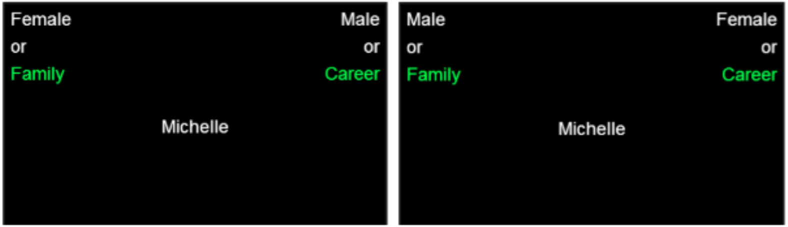
"Attitudes that affect our understanding, actions, and decisions in an unconscious manner" (Kirnan institute, OSU)

- Generally measured using Implicit Association Tests (IATs)

# Implicit attitudes

"Attitudes that affect our understanding, actions, and decisions in an unconscious manner" (Kirnan institute, OSU)

- Generally measured using Implicit Association Tests (IATs)
- Subjects asked to assign words to categories (Greenwald et al. 1998)

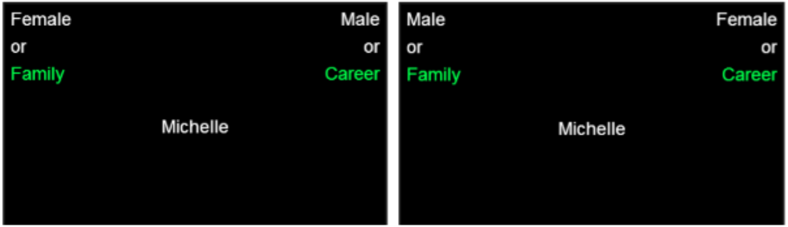



- Comparing reaction times across trials with different word pairs:
  - subjects tend to be slower and more error-prone in assignments against stereotype (e.g. "Michelle" goes to "Female or Career").

# Implicit attitudes

"Attitudes that affect our understanding, actions, and decisions in an unconscious manner" (Kirnan institute, OSU)

- Generally measured using Implicit Association Tests (IATs)
- Subjects asked to assign words to categories (Greenwald et al. 1998)



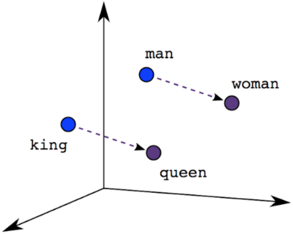

- Comparing reaction times across trials with different word pairs:
  - subjects tend to be slower and more error-prone in assignments against stereotype (e.g. “Michelle” goes to “Female or Career”).
  - IAT score = difference in reaction time between stereotype-consistent and stereotype-inconsistent rounds.

# Caliskan, Bryson, and Narayanan (*Science* 2017)

- “We replicated a spectrum of known biases, as measured by the Implicit Association Test, using a widely used, purely statistical machine-learning model trained on a standard corpus of text from the World Wide Web. . . “

# Caliskan, Bryson, and Narayanan (*Science* 2017)

- "We replicated a spectrum of known biases, as measured by the Implicit Association Test, using a widely used, purely statistical machine-learning model trained on a standard corpus of text from the World Wide Web. . . "

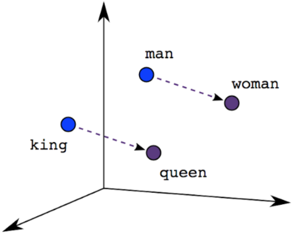



Analogies

- king : queen :: man : woman
- walked : walking :: swam : swimming

# Caliskan, Bryson, and Narayanan (*Science* 2017)

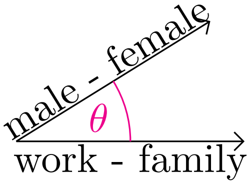
- "We replicated a spectrum of known biases, as measured by the Implicit Association Test, using a widely used, purely statistical machine-learning model trained on a standard corpus of text from the World Wide Web. . . "



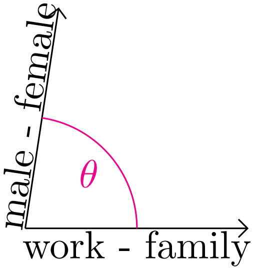

Analogies

- king : queen :: man : woman
- walked : walking :: swam : swimming
- **man : programmer :: woman : homemaker**
- **he : physician :: she : nurse**

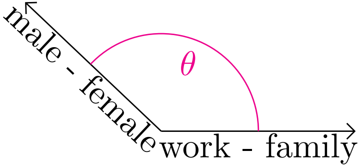
## Measuring Gender Stereotypes using Cosine Similarity




(a)




(b)




(c)

# Example Stimuli

- Targets:
  - **Flowers**: aster, clover, hyacinth, marigold, poppy, azalea, crocus, iris, orchid, rose, bluebell, daffodil, lilac, pansy, tulip, buttercup, daisy, lily, peony, violet, carnation, gladiola, magnolia, petunia, zinnia.
  - **Insects**: ant, caterpillar, flea, locust, spider, bedbug, centipede, fly, maggot, tarantula, bee, cockroach, gnat, mosquito, termite, beetle, cricket, hornet, moth, wasp, blackfly, dragonfly, horsefly, roach, weevil.

# Example Stimuli

- Targets:
  - **Flowers**: aster, clover, hyacinth, marigold, poppy, azalea, crocus, iris, orchid, rose, bluebell, daffodil, lilac, pansy, tulip, buttercup, daisy, lily, peony, violet, carnation, gladiola, magnolia, petunia, zinnia.
  - **Insects**: ant, caterpillar, flea, locust, spider, bedbug, centipede, fly, maggot, tarantula, bee, cockroach, gnat, mosquito, termite, beetle, cricket, hornet, moth, wasp, blackfly, dragonfly, horsefly, roach, weevil.
- Attributes:
  - **Pleasant**: caress, freedom, health, love, peace, cheer, friend, heaven, loyal, pleasure, diamond, gentle, honest, lucky, rainbow, diploma, gift, honor, miracle, sunrise, family, happy, laughter, paradise, vacation.
  - **Unpleasant**: abuse, crash, filth, murder, sickness, accident, death, grief, poison, stink, assault, disaster, hatred, pollute, tragedy, divorce, jail, poverty, ugly, cancer, kill, rotten, vomit, agony, prison.

# Results

- Pleasant vs. Unpleasant?
  - Flowers vs. Insects
  - Musical instruments vs. weapons.

# Results

- Pleasant vs. Unpleasant?
  - Flowers vs. Insects
  - Musical instruments vs. weapons.
  - European-American names vs. African-American names

# Results

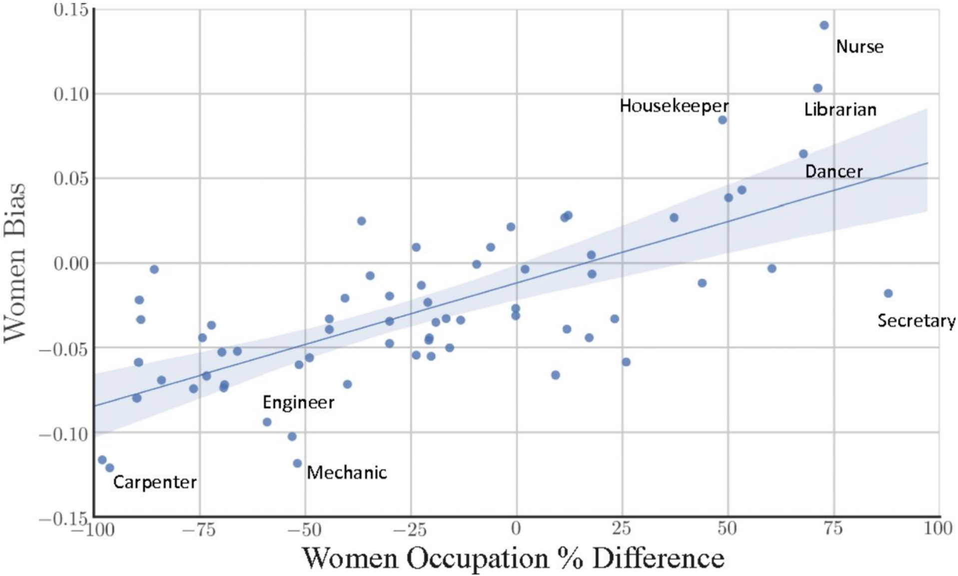
- Pleasant vs. Unpleasant?
  - Flowers vs. Insects
  - Musical instruments vs. weapons.
  - European-American names vs. African-American names
- Male names vs. Female names:

# Results

- Pleasant vs. Unpleasant?
  - Flowers vs. Insects
  - Musical instruments vs. weapons.
  - European-American names vs. African-American names
- Male names vs. Female names:
  - Career words (e.g. professional, corporation, ...) vs. family words (e.g. home, children, ...)
  - Math/science words vs arts words

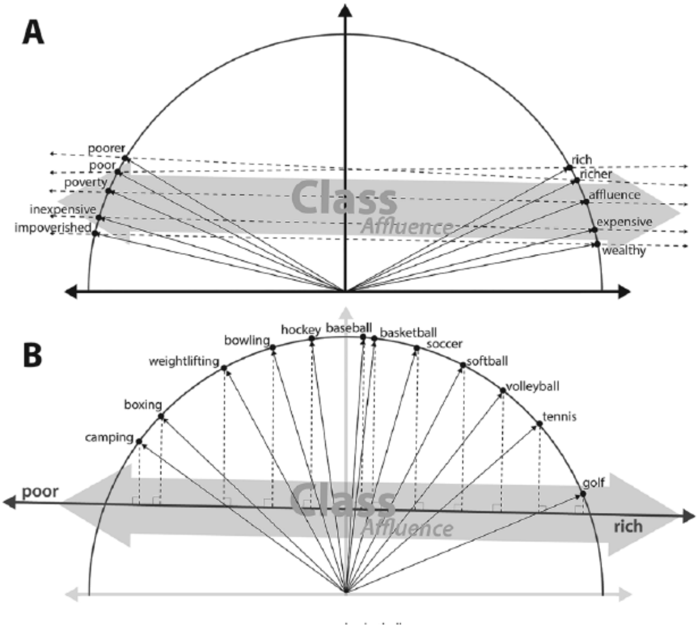
**What do we learn from this?**

# Garg, Schiebinger, Jurafsky, and Zou (PNAS 2018)




Women's occupation relative percentage vs. embedding bias in Google News vectors.

# Kozlowski, Evans, and Taddy (ASR 2019)



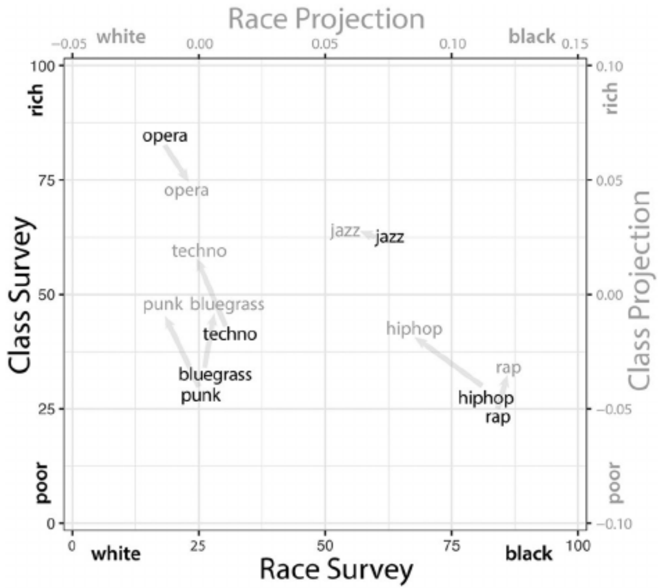



**Figure 3.** Projection of Music Genres onto Race and Class Dimensions of the Google News Word Embedding (Gray) and Average Survey Ratings for Race and Class Associations (Black)

# Time Series Analysis of Affluence

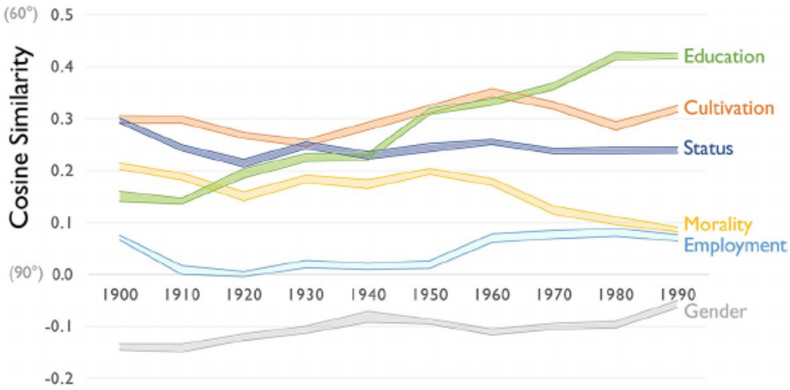



**Figure 5.** Cosine Similarity between the Affluence Dimension and Six Other Cultural Dimensions of Class by Decade; 1900 to 1999 Google Ngrams Corpus
*Note:* Bands represent 90 percent bootstrapped confidence intervals produced by subsampling.

"Among the 10 nouns most highly projecting on the affluence dimension in the first decade of the twentieth century are "fragrance," "perfume," "jewels," and "gems," ..."

# De-Biasing Word Embeddings

# De-Biasing Word Embeddings

- Bolukbasi et al (NIPS 2016):
  - “Geometrically, **gender bias is first shown to be captured by a direction in the word embedding**.“

# De-Biasing Word Embeddings

- Bolukbasi et al (NIPS 2016):
  - "Geometrically, **gender bias is first shown to be captured by a direction in the word embedding**."
  - "Second, gender neutral words are shown to be linearly separable from gender definition words in the word embedding."

# De-Biasing Word Embeddings

- Bolukbasi et al (NIPS 2016):
  - “Geometrically, **gender bias is first shown to be captured by a direction in the word embedding**.“
  - “Second, gender neutral words are shown to be linearly separable from gender definition words in the word embedding.”
  - “Using these properties, we provide a methodology for modifying an embedding to remove gender stereotypes, such as the association between the words receptionist and female, while maintaining desired associations such as between the words queen and female.”

# De-Biasing Word Embeddings

- Bolukbasi et al (NIPS 2016):
  - "Geometrically, **gender bias is first shown to be captured by a direction in the word embedding**."
  - "Second, gender neutral words are shown to be linearly separable from gender definition words in the word embedding."
  - "Using these properties, we provide a methodology for modifying an embedding to remove gender stereotypes, such as the association between the words receptionist and female, while maintaining desired associations such as between the words queen and female."
- But: Gonen and Goldberg (2019):
  - ***"... we argue that this removal is superficial. While the bias is indeed substantially reduced according to the provided bias definition, the actual effect is mostly hiding the bias, not removing it. The gender bias information is still reflected in the distances between 'gender-neutralized' words in the debiased embeddings, and can be recovered from them..."***
- Project idea: use double machine learning to de-bias word embeddings.

# Discussion

# Discussion

- What are we measuring?
  - e.g., how do measurements by person or by group correlate with actual attitudes, for example as measured by IAT scores?

# Discussion

- What are we measuring?
  - e.g., how do measurements by person or by group correlate with actual attitudes, for example as measured by IAT scores?
- What attitudes can be reliably measured?
  - e.g. racial sentiment, especially in light of black sheep problem.

# Discussion

- What are we measuring?
  - e.g., how do measurements by person or by group correlate with actual attitudes, for example as measured by IAT scores?
- What attitudes can be reliably measured?
  - e.g. racial sentiment, especially in light of black sheep problem.
- In what domains is this relevant?
  - social media, news media, politics, legal, scientific, ...

# Discussion

- What are we measuring?
  - e.g., how do measurements by person or by group correlate with actual attitudes, for example as measured by IAT scores?
- What attitudes can be reliably measured?
  - e.g. racial sentiment, especially in light of black sheep problem.
- In what domains is this relevant?
  - social media, news media, politics, legal, scientific, ...
- Does language matter?
  - Djourelova (2020): style change from "illegal" to "undocumented" immigrant softened attitudes toward immigration.