Distributional Semantics

Lecture 10. Recent Trends in Distributional Semantics

Daniil Vodolazsky

April 27th, 2019

Lecture Plan

Lecture Plan

Introduction

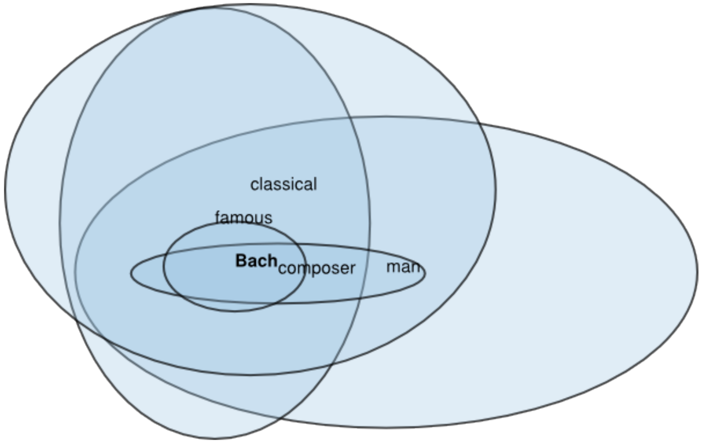
Introduction: What do you think?

What properties of word embeddings are still not studied?

Gaussian Word Embeddings

Gaussian Word Embeddings (2015)

Moving beyond vector **point** representations to **potential functions**, or continuous densities in latent space.



Loss Function

An **energy function** is a function $E_\theta(x, y)$ that scores pairs of inputs x and outputs y, parametrized by θ. The goal of **energy-based learning** is to train the parameters of the energy function to score observed positive input-output pairs higher (or lower, depending on sign conventions) than negative pairs. This is accomplished by means of a loss function L which defines which pairs are positive and negative according to some supervision, and provides gradients on the parameters given the predictions of the energy function. A max-margin ranking objective pushes scores of positive pairs above negatives by a margin:

$$L_m(w, c_p, c_n) = \max(0, m - E(w, c_p) + E(w, c_n))$$

Symmetric Similarity: Expected Likelihood or Probability Product Kernel

While the dot product between two means of independent Gaussians is a perfectly valid measure of similarity (it is the expected dot product), it does not incorporate the covariances and would not enable us to gain any benefit from our probabilistic model.
The most logical next choice for a symmetric similarity function would be to take the inner product between the distributions themselves. For Gaussians, the inner product is defined as

$$E(P_i, P_j) = \int \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) d\mathbf{x} = \mathcal{N}(\mathbf{0}; \boldsymbol{\mu}_i - \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j).$$

Since we aim to discriminatively train the weights of the energy function, and it is always positive, we work not with this quantity directly, but with its logarithm.
The logarithm of the energy (in d dimensions) is

$$\log \mathcal{N}(\mathbf{0}; \boldsymbol{\mu}_i - \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j) = -\frac{1}{2}\log\det(\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j) - \frac{1}{2}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^\top(\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j)^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) - \frac{d}{2}\log(2\pi).$$

Asymmetric Similarity: KL Divergence

We optimize the following energy function (which has a similarly tractable closed form solution for Gaussians):

$$\begin{aligned} -E(P_i, P_j) &= D_{\mathrm{KL}}(\mathcal{N}_j || \mathcal{N}_i) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \log \frac{\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} d\boldsymbol{x} \\ &= \frac{1}{2}((\boldsymbol{\Sigma}_i^{-1}\boldsymbol{\Sigma}_j) + (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) - d - \log \frac{\det(\boldsymbol{\Sigma}_j)}{\det(\boldsymbol{\Sigma}_i)}). \end{aligned}$$

Note the leading negative sign (we define the negative energy), since KL is a distance function and not a similarity. KL divergence is a natural energy function for representing entailment between concepts — a low KL divergence from $\boldsymbol{x}$ to $\boldsymbol{y}$ indicates that we can encode $\boldsymbol{y}$ easily as $\boldsymbol{x}$, implying that $\boldsymbol{y}$ entails $\boldsymbol{x}$.

Poincare Word Embeddings

Poincare Word Embeddings (2017)

Although embedding methods have proven successful in numerous applications, they suffer from a fundamental limitation: their ability to model complex patterns is inherently bounded by the dimensionality of the embedding space. As a consequence, no method yet exists that is able to compute embeddings of large graph-structured data – such as social networks, knowledge graphs or taxonomies – without loss of information.

Introduction to Hyperbolic Geometry

Hyperbolic geometry is a non-Euclidean geometry which studies spaces of constant negative curvature. In network science, hyperbolic spaces have started to receive attention as they are well-suited to model hierarchical data.

In hyperbolic geometry this kind of tree structure can be modeled easily in two dimensions: nodes that are exactly l levels below the root are placed on a sphere in hyperbolic space with radius $r \propto l$ and nodes that are less than l levels below the root are located within this sphere. This type of construction is possible as hyperbolic disc area and circle length grow exponentially with their radius. [1]

[1] For instance, in a two dimensional hyperbolic space with constant curvature $K = -1$, the length of a circle is given as $2\pi \sinh r$ while the area of a disc is given as $2\pi(\cosh r - 1)$. Since $\sinh r = \frac{1}{2}(e^r - e^{-r})$ and $\cosh r = \frac{1}{2}(e^r + e^{-r})$, both disc area and circle length grow exponentially with r.

Poincaré Embeddings

In addition to the similarity of objects, we intend to also reflect this hierarchy in the embedding space to improve over existing methods in two ways:

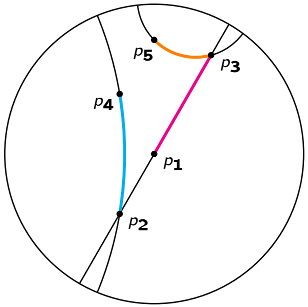
1. By inducing an appropriate bias on the structure of the embedding space, we aim at learning more parsimonious embeddings for superior generalization performance and decreased runtime and memory complexity.
2. By capturing the hierarchy explicitly in the embedding space, we aim at gaining additional insights about the relationships between symbols and the importance of individual symbols.

Let $\mathcal{B}^d = \{\boldsymbol{x} \in \mathbb{R}^d \mid \|\boldsymbol{x}\| < 1\}$ be the open d-dimensional unit ball. The Poincaré ball model of hyperbolic space corresponds then to the Riemannian manifold $(\mathcal{B}^d, g_{\boldsymbol{x}})$, i.e., the open unit ball equipped with the Riemannian metric tensor

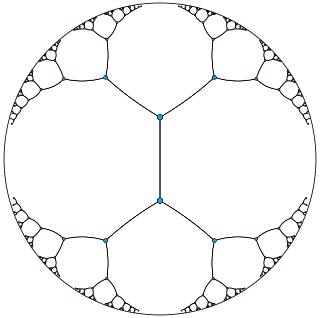
$$g_{\boldsymbol{x}} = \left(\frac{2}{1 - \|\boldsymbol{x}\|^2}\right)^2 g^E,$$

where $\boldsymbol{x} \in \mathcal{B}^d$ and g^E denotes the Euclidean metric tensor.

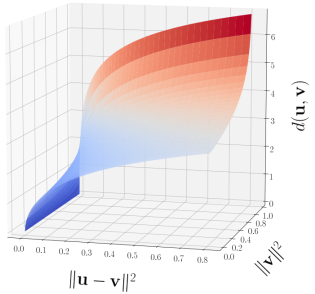
Poincaré Embeddings



Geodesics of the Poincaré disk



Embedding of a tree in $\mathcal{B}^2$



Growth of Poincaré distance

The distance between points $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{B}^d$ is given as

$$d(\boldsymbol{u}, \boldsymbol{v}) = \operatorname{arcosh}\left(1 + 2\frac{\|\boldsymbol{u}-\boldsymbol{v}\|^2}{(1-\|\boldsymbol{u}\|^2)(1-\|\boldsymbol{v}\|^2)}\right). \quad (1)$$

Multimodal Word Embeddings

Multimodal Word Embeddings (2013)

- Image captioning
- Image generation by a text description
- Information retrieval
- Neural machine translation
- etc.

Generating Descriptions of Images

Template-based methods involve filling in sentence templates, such as triplets, based on the results of object detections and spatial relationships. While these approaches can produce accurate descriptions, they are often more ‘robotic’ in nature and do not generalize to the fluidity and naturalness of captions written by humans.

Composition-based methods aim to harness existing image-caption databases by extracting components of related captions and composing them together to generate novel descriptions. The advantage of these approaches are that they allow for a much broader and more expressive class of captions that are more fluent and human-like then template-based approaches.

Neural network methods aim to generate descriptions by sampling from conditional neural language models. The initial work in this area, based off of multimodal neural language models, generated captions by conditioning on feature vectors from the output of a deep convolutional network. These ideas were recently extended to multimodal recurrent networks with significant improvements.

Generating Descriptions of Images

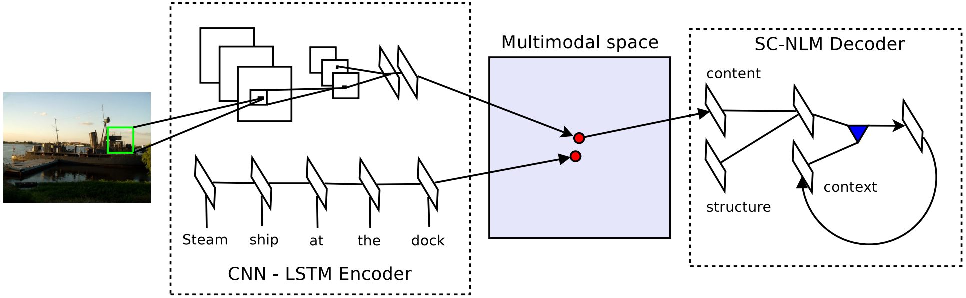
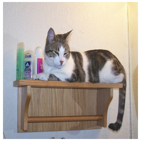
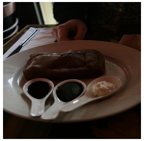
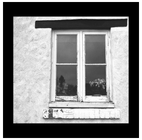
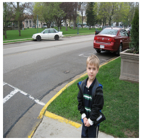


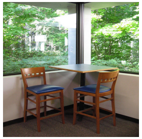
Рис.: **Encoder:** A deep convolutional network (CNN) and long short-term memory recurrent network (LSTM) for learning a joint image-sentence embedding. **Decoder:** A new neural language model that combines structure and content vectors for generating words one at a time in sequence.

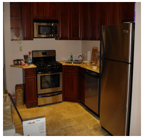

there is a cat sitting on a shelf .

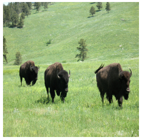

a plate with a fork and a piece of cake .

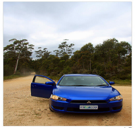

a black and white photo of a window .

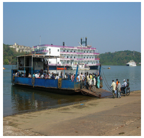

a young boy standing on a parking lot next to cars .

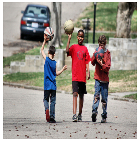

a wooden table and chairs arranged in a room .

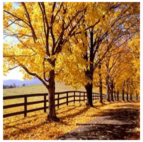

a kitchen with stainless steel appliances .

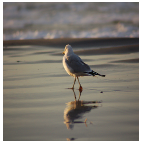

this is a herd of cattle out in the field .

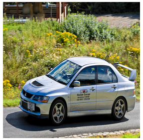

a car is parked in the middle of nowhere .

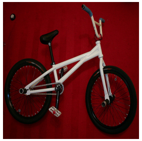

a ferry boat on a marina with a group of people .

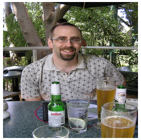

a little boy with a bunch of friends on the street .


a giraffe is standing next to a fence in a field .
(hallucination)


the two birds are trying to be seen in the water .
(counting)


a parked car while driving down the road .
(contradiction)


the handlebars are trying to ride a bike rack .
(nonsensical)


a woman and a bottle of wine in a garden .
(gender)

Multimodal Distributed Representations

Suppose we are given image-description pairs each corresponding to an image and a description that correctly describes the image.

Let D be the dimensionality of an image feature vector, K the dimensionality of the embedding space and let $V = |\mathcal{W}|$. Let $\boldsymbol{W}_I \in \mathbb{R}^{K \times D}$ and $\boldsymbol{W}_W \in \mathbb{R}^{K \times V}$ be the image and word embedding matrices.

Given an image description $S = \{w_1, \dots, w_N\}$ let $\{\boldsymbol{w}_{w_1}, \dots, \boldsymbol{w}_{w_N}\}$ denote the corresponding K-dimentional word representations to words $w_1, \dots, w_N$. The representation of a sentence $\boldsymbol{w}_S$ is the hidden state of the LSTM at time step N.

Let $\boldsymbol{q} \in \mathbb{R}^D$ denote an image feature vector and let $\boldsymbol{x} = \boldsymbol{W}_I \cdot \boldsymbol{q} \in \mathbb{R}^K$ be the image embedding. We minimize w.r.t all trainable parameters the following pairwise ranking loss:

$$\sum_{\boldsymbol{x}} \sum_{n} \max(0, \alpha - \cos(\boldsymbol{x}, \boldsymbol{w}_S) + \cos(\boldsymbol{x}, \boldsymbol{w}_n) + \sum_{\boldsymbol{w}} \sum_{n} \max(0, \alpha - \cos(\boldsymbol{x}, \boldsymbol{w}_S) + \cos(\boldsymbol{x}_n, \boldsymbol{w}_S))$$

where $\boldsymbol{w}_n$ is a contrastive (non-descriptive) sentence for image embedding $\boldsymbol{x}$, and vice-versa with $\boldsymbol{x}_n$.

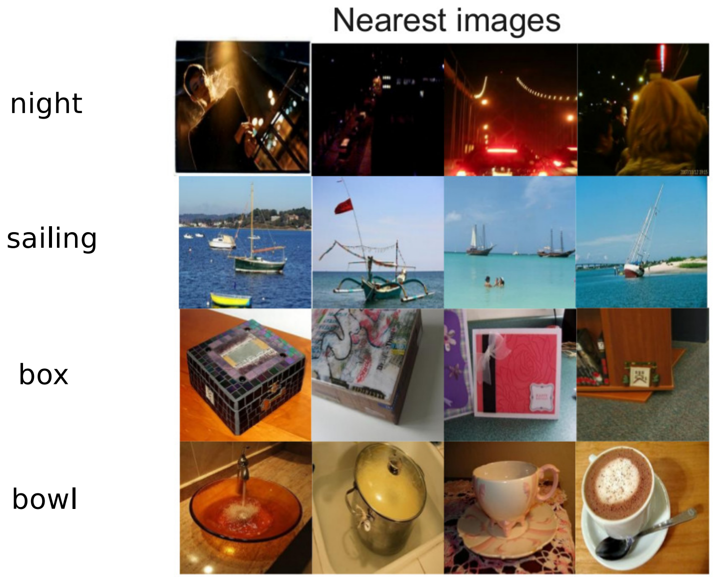
Multimodal Linguistic Regularities

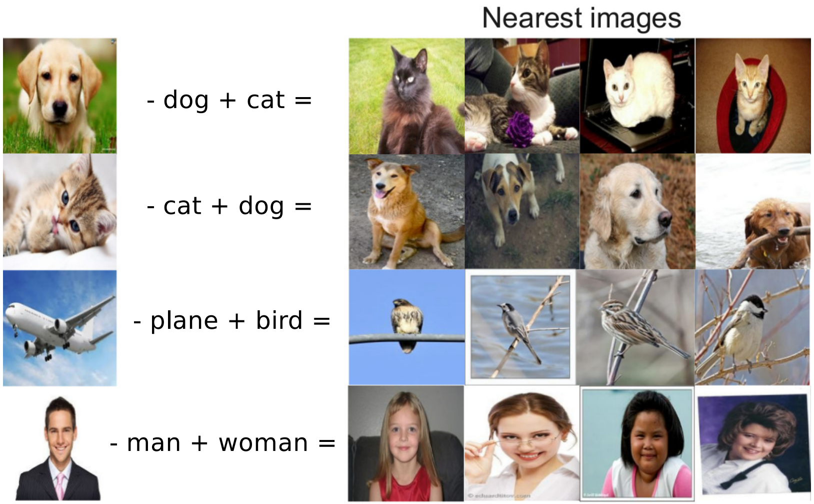
Word embeddings learned with Skip-gram or neural language models were shown to exhibit linguistic regularities that allow these models to perform analogical reasoning. For instance, "man"is to "woman"as "king"is to ? can be answered by finding the closest vector to "king "man"+ "woman". A natural question we ask is whether multimodal vector spaces exhibit the same phenomenon. Would *image of a blue car* - "blue"+ "red"be near images of red cars?

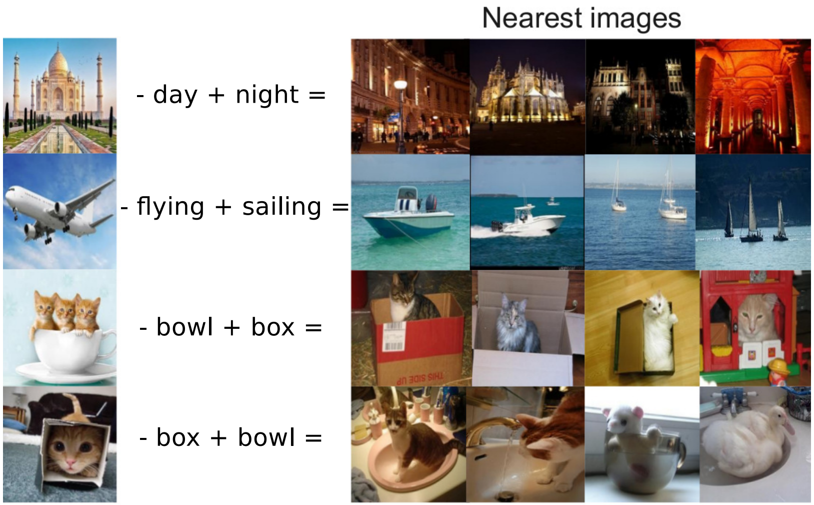
Given a query image $\boldsymbol{q}$, a negative word w_n and a positive word w_p, we seek an image $\boldsymbol{x}^*$ such that:

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmax}} \frac{(\boldsymbol{q} - \bar{\boldsymbol{w}}_{w_n} + \bar{\boldsymbol{w}}_{w_p})^\top \boldsymbol{x}}{\|\boldsymbol{q} - \bar{\boldsymbol{w}}_{w_n} + \bar{\boldsymbol{w}}_{w_p}\|}, \tag{2}$$

where $\bar{\boldsymbol{w}}_w = \boldsymbol{w}_w / ||\boldsymbol{w}_w||$.


Nearest images
night
sailing
box
bowl


Nearest images
- dog + cat =
- cat + dog =
- plane + bird =
- man + woman =


Nearest images
- day + night =
- flying + sailing =
- bowl + box =
- box + bowl =

Diachronic Word Embeddings

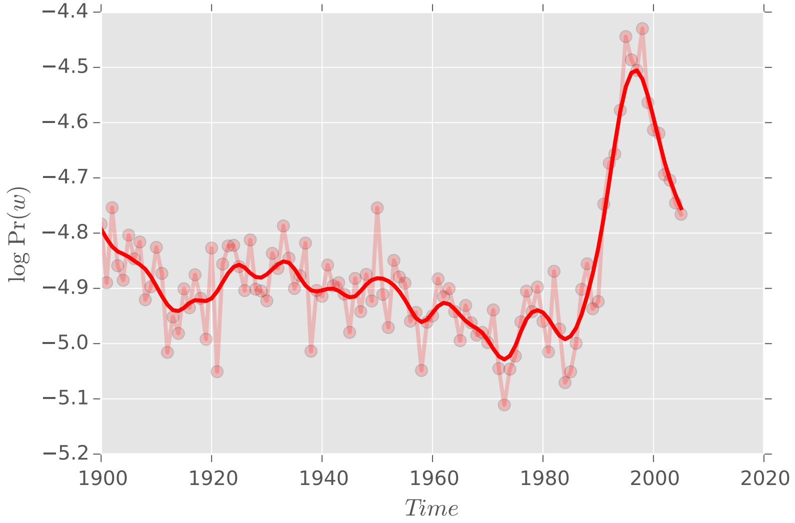
Frequency Method

Frequency based methods can capture linguistic shift, as changes in frequency can correspond to words acquiring or losing senses.
We calculate for each time snapshot corpus $\mathcal{D}^{(t)}$, a unigram language model. Specifically, we construct the time series for a word w as follows:

$$\mathcal{T}^{(t)}(w) = \log \frac{\#(w \in \mathcal{D}^{(t)})}{|\mathcal{D}^{(t)}|},$$

where $\#(w \in \mathcal{D}^{(t)})$ is the number of occurrences of the word w in corpus snapshot $\mathcal{D}^{(t)}$.

Frequency Method: time series for a word “gay”



Syntactic Method

A word could evolve a new syntactic functionality by acquiring a new part of speech category.

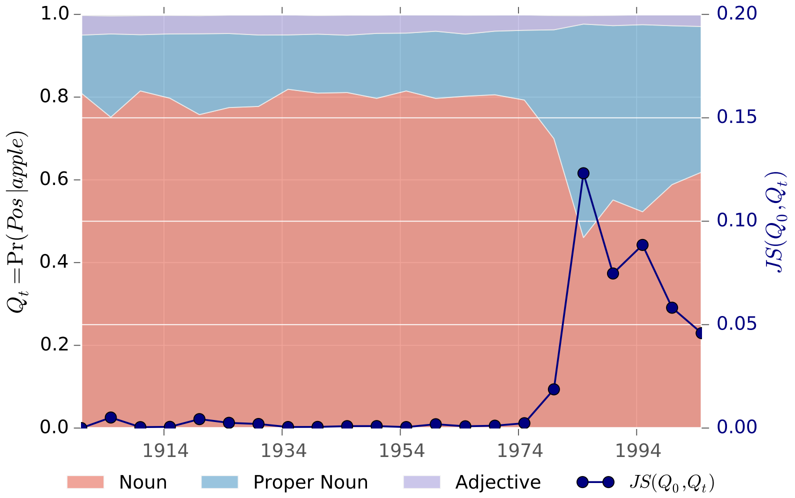
For example, `apple` used to be only a “Noun” describing a fruit, but over time it acquired the new part of speech “Proper Noun” to indicate the new sense describing a technical company.

After annotating a corpus with POS tags, we calculate the probability distribution of part of speech tags Q^t given the word w and time snapshot t as follows: $Q^t = P_{x \sim \text{POS Tags}}(x \mid w, \mathcal{D}^{(t)})$.
To quantify the temporal change between two time snapshots corpora, for a specific word w, we calculate the divergence between the POS distributions in both snapshots.

$$\mathcal{T}^{(t)}(w) = D_{JS}(Q^0, Q^t),$$

where D_{JS} is the Jenssen-Shannon divergence.

Syntactic Method: time series for a word "apple"

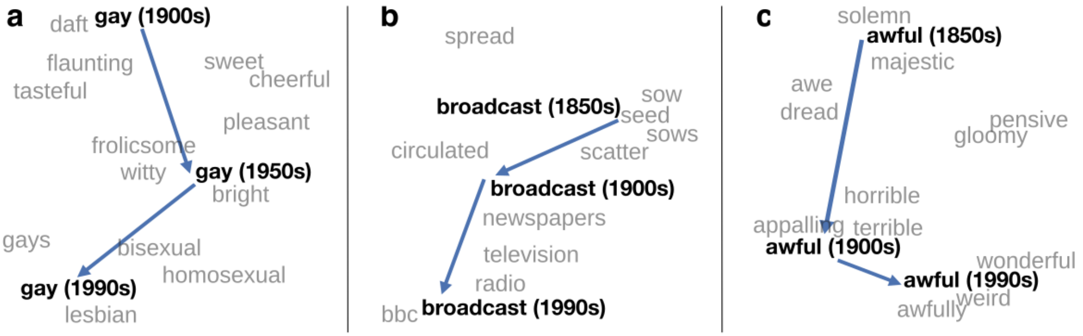


Distributional Method

A pipeline for distributional methods consists of several steps.

1. Obtaining word embedding matrices $\boldsymbol{W}^{(t)}$ for each time period t using, e.g., PPMI, SVD or SGNS;
2. Aligning embeddings;
3. Constructing time series.

Distributional Method: semantic changes in 20th century



Aligning historical embeddings

Explicit PPMI vectors are naturally aligned, as each column simply corresponds to a context word. Low-dimensional embeddings will not be naturally aligned due to the non-unique nature of the SVD and the stochastic nature of SGNS. In particular, both these methods may result in arbitrary orthogonal transformations, which do not affect pairwise cosine-similarities within-years but will preclude comparison of the same word across time.

We use orthogonal Procrustes to align the learned low-dimensional embeddings. Defining $\boldsymbol{W}^{(t)} \in \mathbb{R}^{d \times |\mathcal{W}|}$ as the matrix of word embeddings learned at year t, we align across time-periods while preserving cosine similarities by optimizing:

$$\boldsymbol{R}^{(t)} = \arg \min_{\boldsymbol{Q}^\top \boldsymbol{Q} = \boldsymbol{I}} \|\boldsymbol{Q}\boldsymbol{W}^{(t)} - \boldsymbol{W}^{(t+1)}\|_F, \tag{3}$$

with $\boldsymbol{R}^{(t)} \in \mathbb{R}^{d \times d}$. The solution corresponds to the best rotational alignment and can be obtained efficiently using an application of SVD.

Time-series from historical embeddings

Diachronic word embeddings can be used in two ways to quantify semantic change: (i) we can measure changes in pair-wise word similarities over time, or (ii) we can measure how an individual word's embedding shifts over time.

Pair-wise similarity time-series
Measuring how the cosine-similarity between pairs of words changes over time allows us to test hypotheses about specific linguistic or cultural shifts in a controlled manner. At first, we quantify shifts by computing the similarity time-series

$$s^{(t)}(w_i, w_j) = \cos(\boldsymbol{u}_{w_i}^{(t)}, \boldsymbol{u}_{w_j}^{(t)})$$

between two words w_i and w_j over a time-period $(t, \ldots, t+\Delta)$. We then measure the Spearman correlation of this series against time, which allows us to assess the magnitude and significance of pairwise similarity shifts.

Time-series from historical embeddings

Diachronic word embeddings can be used in two ways to quantify semantic change: (i) we can measure changes in pair-wise word similarities over time, or (ii) we can measure how an individual word's embedding shifts over time.

Measuring semantic displacement
After aligning the embeddings for individual time-periods, we can use the aligned word vectors to compute the semantic displacement that a word has undergone during a certain time-period. In particular, we can directly compute the cosine-distance between a word's representation for different time-periods, i.e. $\cos(\boldsymbol{u}_{w_i}^{(t)}, \boldsymbol{u}_{w_i}^{(t+\Delta)})$, as a measure of semantic change. We can also use this measure to quantify 'rates' of semantic change for different words by looking at the displacement between consecutive time-points.

Fair Word Embeddings

Fair Word Embeddings

Gender stereotype she-he analogies.

sewing-carpentry	register-nurse-physician	housewife-shopkeeper
nurse-surgeon	interior designer-architect	softball-baseball
blond-burly	feminism-conservatism	cosmetics-pharmaceuticals
giggle-chuckle	vocalist-guitarist	petite-lanky
sassy-snappy	diva-superstar	charming-affable
volleyball-football	cupcakes-pizzas	hairdresser-barber

Gender appropriate she-he analogies.

queen-king	sister-brother	mother-father
waitress-waiter	ovarian cancer-prostate cancer	convent-monastery

Рис.: Examples of automatically generated analogies for the pair she-he.

Fair Word Embeddings

Our goal is to **reduce gender biases** in the word embedding while preserving the useful properties of the embedding. The goals of debiasing are:

1. Reduce bias:
   1. Ensure that gender neutral words such as <u>nurse</u> are equidistant between gender pairs such as <u>he</u> and <u>she</u>.
   2. Reduce gender associations that pervade the embedding even among gender neutral words.
2. Maintain embedding utility:
   1. Maintain meaningful non-gender-related associations between gender neutral words, including associations within stereotypical categories of words such as fashion-related words or words associated with football.
   2. Correctly maintain definitional gender associations such as between <u>man</u> and <u>father</u>.

Identifying the gender subspace

Language use is "messy" and therefore individual word pairs do not always behave as expected. For instance, the word man has several different usages: it may be used as an exclamation as in oh man! or to refer to people of either gender or as a verb, e.g., man the station. To more robustly estimate bias, we shall aggregate across multiple paired comparisons. By combining several directions, such as *she* − *he* and *woman* − *man*, we identify a **gender direction** $g \in \mathbb{R}^d$ that largely captures gender in the embedding. This direction helps us to quantify direct and indirect biases in words and associations.
In English as in many languages, there are numerous gender pair terms, and for each we can consider the difference between their embeddings. Before looking at the data, one might imagine that they all had roughly the same vector differences, as in the following caricature:

$$\begin{aligned} grandmother &= wise + gal \\ grandfather &= wise + guy \\ grandmother - grandfather &= gal - guy = g \end{aligned}$$

However, gender pair differences are not parallel in practice, for multiple reasons.

Direct bias

To measure **direct bias**, we first identify words that should be gender-neutral for the application in question. Given the gender neutral words, denoted by $\mathcal{N}$, and the gender direction learned from above, g, we define the direct gender bias of an embedding to be

$$\mathrm{DirectBias}_c = \frac{1}{|\mathcal{N}|} \sum_{w \in \mathcal{N}} |\cos(\vec{w}, g)|^c$$

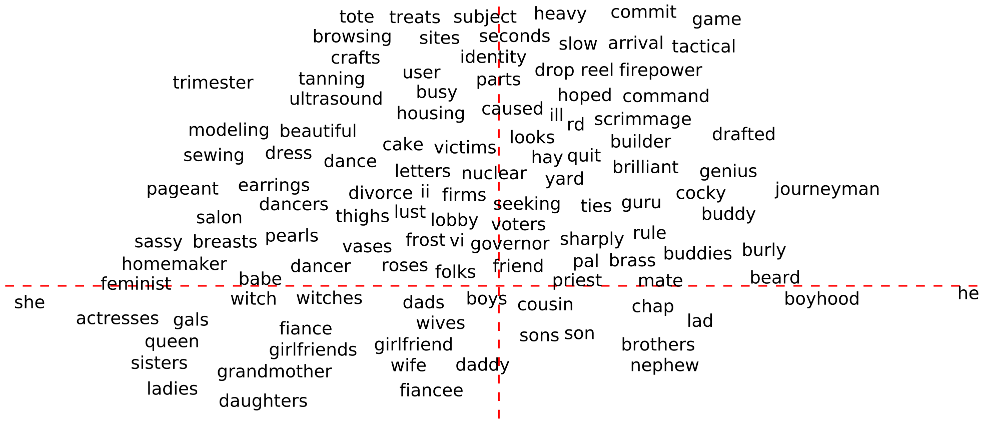
where c is a parameter that determines how strict do we want to in measuring bias. If c is 0, then $|\cos(\vec{w} - g)|^c = 0$ only if $\vec{w}$ has no overlap with g and otherwise it is 1. Such strict measurement of bias might be desirable in settings such as the college admissions example from the Introduction, where it would be unacceptable for the embedding to introduce a slight preference for one candidate over another by gender. A more gradual bias would be setting $c = 1$.

Indirect bias

Imagine completely removing from the embedding both words in gender pairs (as well as words such as beard or uterus that are arguably gender-specific but which cannot be paired). There would still be indirect gender association in that a word that should be gender neutral, such as receptionist, is closer to softball than football.
The gender subspace g that we have identified allows us to quantify the contribution of g to the similarities between any pair of words. We can decompose a given word vector $w \in \mathbb{R}^d$ as $w = w_g + w_\perp$, where $w_g = (w \cdot g)g$ and $w_\perp = w - w_g$. We define the gender component to the similarity between two word vectors w and v as

$$\beta(w, v) = \left(w \cdot v - \cos(w_\perp, v_\perp)\right) \Big/ w \cdot v.$$

The metric quantifies how much this inner product changes (as a fraction of the original inner product value) due to this operation of removing the gender subspace. Note that $\beta(w, w) = 0$, which is reasonable since the similarity of a word to itself should not depend on gender contribution.



Selected words projected along two axes: x is a projection onto the difference between the embeddings of the words he and she, and y is a direction learned in the embedding that captures gender neutrality, with gender neutral words above the line and gender specific words below the line. Our hard debiasing algorithm removes the gender pair associations for gender neutral words. In this figure, the words above the horizontal line would all be collapsed to the vertical line.

Debiasing algorithms

The first step, called **Identify gender subspace**, is to identify a direction (or, more generally, a subspace) of the embedding that captures the bias. For the second step, we define two options: **Neutralize and Equalize** or **Soften**. **Neutralize** ensures that gender neutral words are zero in the gender subspace. **Equalize** perfectly equalizes sets of words outside the subspace and thereby enforces the property that any neutral word is equidistant to all words in each equality set.

The disadvantage of Equalize is that it removes certain distinctions that are valuable in certain applications. For instance, one may wish a language model to assign a higher probability to the phrase <u>to grandfather a regulation</u>) than <u>to grandmother a regulation</u> since <u>grandfather</u> has a meaning that <u>grandmother</u> does not – equalizing the two removes this distinction. The Soften algorithm reduces the differences between these sets while maintaining as much similarity to the original embedding as possible, with a parameter that controls this trade-off.

Debiasing algorithms. Step 1

A subspace B is defined by k orthogonal unit vectors $B = \{b_1, \dots, b_k\} \subset \mathbb{R}^d$. In the case $k = 1$, the subspace is simply a direction. We denote the projection of a vector v onto B by,

$$v_B = \sum_{j=1}^{k} (v \cdot b_j) b_j.$$

This also means that $v - v_B$ is the projection onto the orthogonal subspace.

Step 1: Identify gender subspace. Inputs: word sets W, defining sets $D_1, D_2, \dots, D_n \subset W$ as well as embedding $\left\{\vec{w} \in \mathbb{R}^d\right\}_{w \in W}$ and integer parameter $k \geq 1$. Let

$$\mu_i := \sum_{w \in D_i} \vec{w}/|D_i|$$

be the means of the defining sets. Let the bias subspace B be the first k rows of $\mathrm{SVD}(\mathbf{C})$ where

$$\mathbf{C} := \sum_{i=1}^{n} \sum_{w \in D_i} (\vec{w} - \mu_i)^T (\vec{w} - \mu_i)/|D_i|.$$

Debiasing algorithms. Step 2a

Step 2a: Hard de-biasing (neutralize and equalize). Additional inputs: words to neutralize $N \subseteq W$, family of equality sets $\mathcal{E} = \{E_1, E_2, \ldots, E_m\}$ where each $E_i \subseteq W$. For each word $w \in N$, let $\vec{w}$ be re-embedded to

$$\vec{w} := (\vec{w} - \vec{w}_B) \big/ \|\vec{w} - \vec{w}_B\|.$$

For each set $E \in \mathcal{E}$, let

$$\mu := \sum_{w \in E} w / |E|$$

$$\nu := \mu - \mu_B$$

$$\text{For each } w \in E, \quad \vec{w} := \nu + \sqrt{1 - \|\nu\|^2} \frac{\vec{w}_B - \mu_B}{\|\vec{w}_B - \mu_B\|}$$

Finally, output the subspace B and the new embedding $\left\{\vec{w} \in \mathbb{R}^d\right\}_{w \in W}$.

Debiasing algorithms. Step 2b

Step 2b: Soft bias correction. Overloading the notation, we let $W \in \mathbb{R}^{d \times |vocab|}$ denote the matrix of all embedding vectors and N denote the matrix of the embedding vectors corresponding to gender neutral words. W and N are learned from some corpus and are inputs to the algorithm. The desired debiasing transformation $T \in \mathbb{R}^{d \times d}$ is a linear transformation that seeks to preserve pairwise inner products between all the word vectors while minimizing the projection of the gender neutral words onto the gender subspace. This can be formalized as the following optimization problem

$$\min_T \|(TW)^T(TW) - W^T W\|_F^2 + \lambda \|(TN)^T(TB)\|_F^2$$

where B is the gender subspace learned in Step 1 and λ is a tuning parameter that balances the objective of preserving the original embedding inner products with the goal of reducing gender bias. For λ large, T would remove the projection onto B from all the vectors in N, which corresponds exactly to Step 2a. The output embedding is normalized to have unit length, $\hat{W} = \{Tw/\|Tw\|_2, w \in W\}$.

Questions?

Seminar

Overview

1. Word embeddings in 2017: Trends and future directions

http://ruder.io/word-embeddings-2017/

Link