INF5820
Distributional Semantics: Extracting Meaning from Data

Lecture 5
Kings and queens, men and women: semantic relations between word embeddings

Andrey Kutuzov
andreku@ifi.uio.no

23 November 2016

Contents

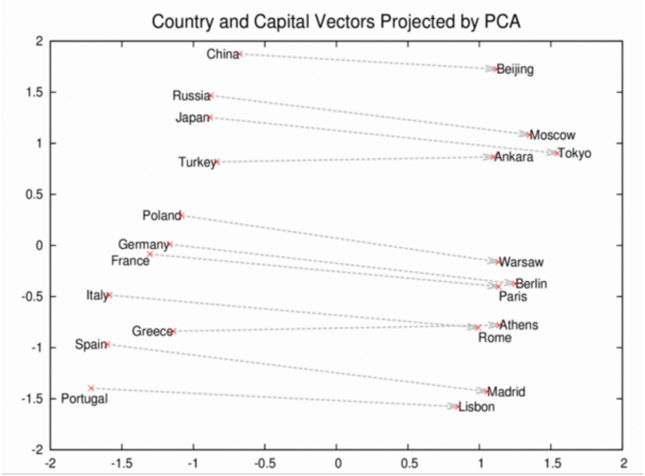
Semantic relations as geometrical directions

- Reducing high-dimensional representations to 2D or 3D can be used not only to create fancy pictures (or videos);

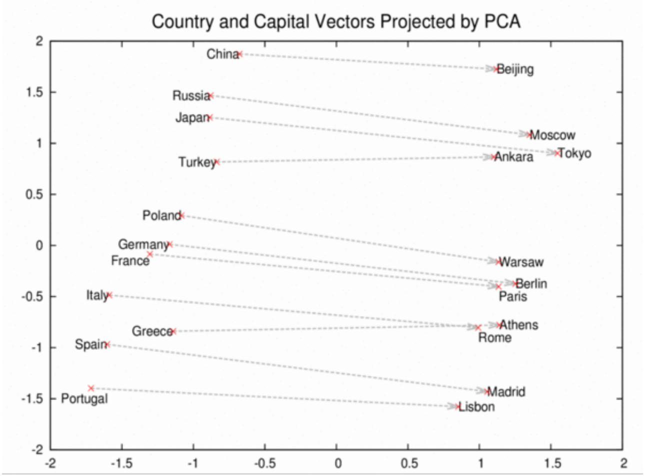
Semantic relations as geometrical directions

- Reducing high-dimensional representations to 2D or 3D can be used not only to create fancy pictures (or videos);
- Sometimes looking at the data in an understandable form can bring great insights...

Semantic relations as geometrical directions



Semantic relations as geometrical directions



'...ability of the model to automatically organize concepts and learn implicitly the relationships between them...' [Mikolov et al., 2013b]

Semantic relations as geometrical directions

A surprising discovery

- Algebraic operations on word embeddings reflect semantic relations

Semantic relations as geometrical directions

A surprising discovery

- Algebraic operations on word embeddings reflect semantic relations
- simple operations with word vectors, like addition, subtraction, finding average, etc, produce intuitively meaningful results.

Semantic relations as geometrical directions

A surprising discovery

- Algebraic operations on word embeddings reflect semantic relations
- simple operations with word vectors, like addition, subtraction, finding average, etc, produce intuitively meaningful results.

Examples

- **Element-wise adding** phone vector to mobile vector results in a vector very close to cellphone;

Semantic relations as geometrical directions

A surprising discovery

- Algebraic operations on word embeddings reflect semantic relations
- simple operations with word vectors, like addition, subtraction, finding average, etc, produce intuitively meaningful results.

Examples

- **Element-wise adding** phone vector to mobile vector results in a vector very close to cellphone;
- **Element-wise subtracting** France vector from Paris vector and adding Germany vector results in a vector very close to Berlin;

Semantic relations as geometrical directions

A surprising discovery

- Algebraic operations on word embeddings reflect semantic relations
- simple operations with word vectors, like addition, subtraction, finding average, etc, produce intuitively meaningful results.

Examples

- **Element-wise adding** phone vector to mobile vector results in a vector very close to cellphone;
- **Element-wise subtracting** France vector from Paris vector and adding Germany vector results in a vector very close to Berlin;

$$\vec{P} - \vec{F} + \vec{G} \approx \vec{B} \tag{1}$$

Semantic relations as geometrical directions

A surprising discovery

- Algebraic operations on word embeddings reflect semantic relations
- simple operations with word vectors, like addition, subtraction, finding average, etc, produce intuitively meaningful results.

Examples

- **Element-wise adding** phone vector to mobile vector results in a vector very close to cellphone;
- **Element-wise subtracting** France vector from Paris vector and adding Germany vector results in a vector very close to Berlin;

$$\vec{P} - \vec{F} + \vec{G} \approx \vec{B} \tag{1}$$

or

$$\vec{G} + \vec{P} - \vec{F} \approx \vec{B} \tag{2}$$

Semantic relations as geometrical directions

- The latter operation can be expressed as solving proportions or performing analogical inference:

Semantic relations as geometrical directions

- The latter operation can be expressed as solving proportions or performing analogical inference:
- '*France is to Paris as Germany is to what?*'

Semantic relations as geometrical directions

- The latter operation can be expressed as solving proportions or performing analogical inference:
- '*France is to Paris as Germany is to what?*'
- this is what Google Analogy dataset tries to evaluate.

Semantic relations as geometrical directions

- The latter operation can be expressed as solving proportions or performing analogical inference:
- '*France is to Paris as Germany is to what?*'
- this is what Google Analogy dataset tries to evaluate.

Stanford folks call that 'directions of meaning' [Pennington et al., 2014].

Semantic relations as geometrical directions

- The latter operation can be expressed as solving proportions or performing analogical inference:
- '*France is to Paris as Germany is to what?*'
- this is what Google Analogy dataset tries to evaluate.

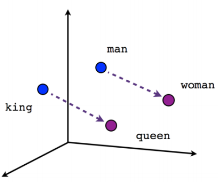
Stanford folks call that 'directions of meaning' [Pennington et al., 2014]. The GloVe model to some extent was designed with this particular task in mind.

Semantic relations as geometrical directions

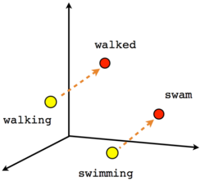
But isn't it simply finding 'geometrical' directions in the high-dimensional semantic space – from one word to another?

Semantic relations as geometrical directions

But isn't it simply finding 'geometrical' directions in the high-dimensional semantic space – from one word to another?



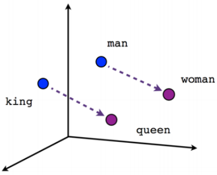
Male-Female



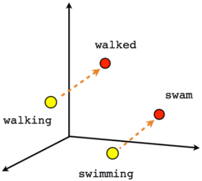
Verb tense

Semantic relations as geometrical directions

But isn’t it simply finding ‘geometrical’ directions in the high-dimensional semantic space – from one word to another?



Male-Female

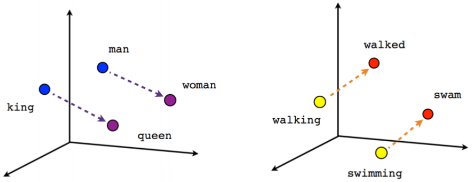


Verb tense

Can be any inference relation: plurality (*dog*→*dogs*),

Semantic relations as geometrical directions

But isn't it simply finding 'geometrical' directions in the high-dimensional semantic space – from one word to another?



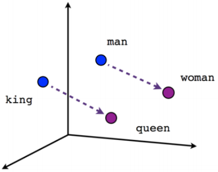
Male-Female

Verb tense

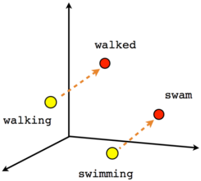
Can be any inference relation: plurality (*dog*$\rightarrow$*dogs*), causality (*life*$\rightarrow$*experience*),

Semantic relations as geometrical directions

But isn't it simply finding 'geometrical' directions in the high-dimensional semantic space – from one word to another?



Male-Female

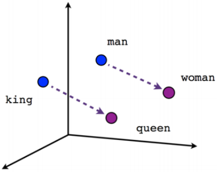


Verb tense

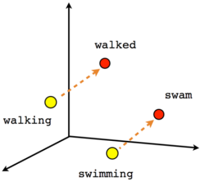
Can be any inference relation: plurality (*dog*→*dogs*), causality (*life*→*experience*), hypernymy (*falcon*→*bird*),

Semantic relations as geometrical directions

But isn't it simply finding 'geometrical' directions in the high-dimensional semantic space – from one word to another?



Male-Female

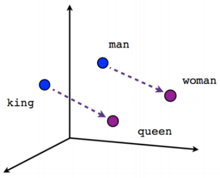


Verb tense

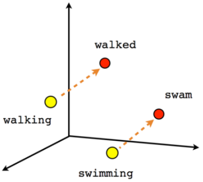
Can be any inference relation: plurality (*dog*→*dogs*), causality (*life*→*experience*), hypernymy (*falcon*→*bird*), meronymy (*leg*→*body*),

Semantic relations as geometrical directions

But isn't it simply finding 'geometrical' directions in the high-dimensional semantic space – from one word to another?



Male-Female



Verb tense

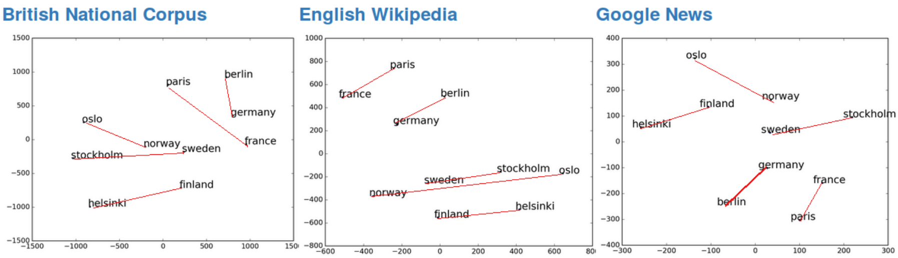
Can be any inference relation: plurality (*dog*→*dogs*), causality (*life*→*experience*), hypernymy (*falcon*→*bird*), meronymy (*leg*→*body*), antonymy (*hot*→*cold*), etc.

Semantic relations as geometrical directions

Semantic direction 'a capital of' in different models:

Semantic relations as geometrical directions

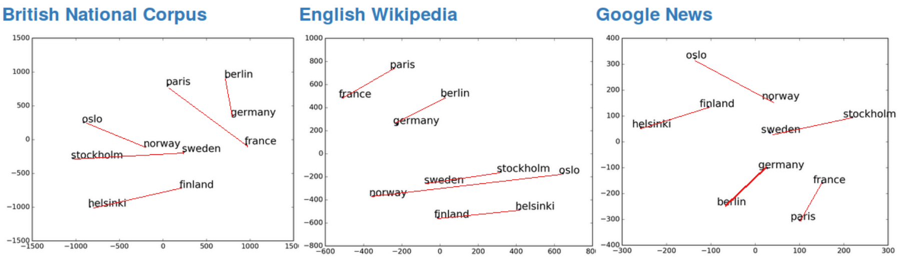
Semantic direction ‘a capital of’ in different models:



Visualized using t-SNE [Van der Maaten and Hinton, 2008]

Semantic relations as geometrical directions

Semantic direction ‘a capital of’ in different models:



Visualized using t-SNE [Van der Maaten and Hinton, 2008]

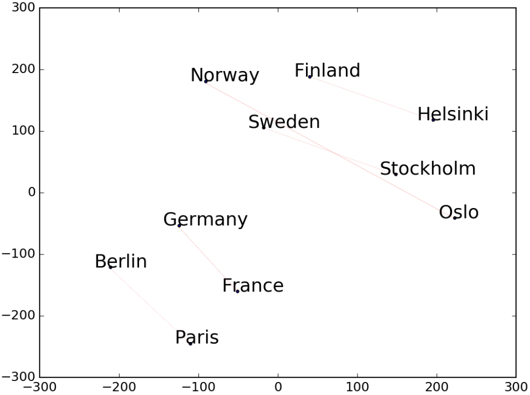
But why the *Google News* model looks so bad? Any ideas?

Semantic relations as geometrical directions

...we used lower-case lemmas, and this model has separate embeddings for title-case and lower-case words (quite annoying). Let's try title-case countries and cities...

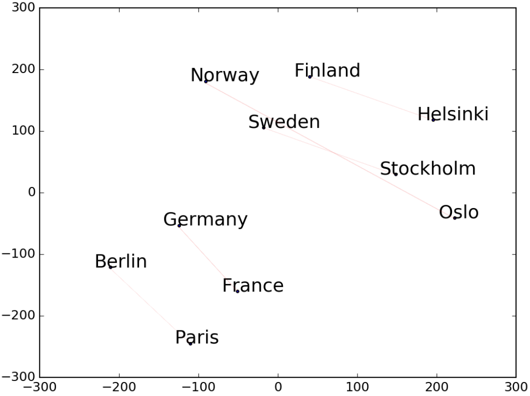
Semantic relations as geometrical directions

...we used lower-case lemmas, and this model has separate embeddings for title-case and lower-case words (quite annoying). Let's try title-case countries and cities...



Semantic relations as geometrical directions

...we used lower-case lemmas, and this model has separate embeddings for title-case and lower-case words (quite annoying). Let's try title-case countries and cities...



Looks better, doesn't it?

Contents

Why vector algebra works?

Why sum (element-wise addition) works?

Why vector algebra works?

Why sum (element-wise addition) works?

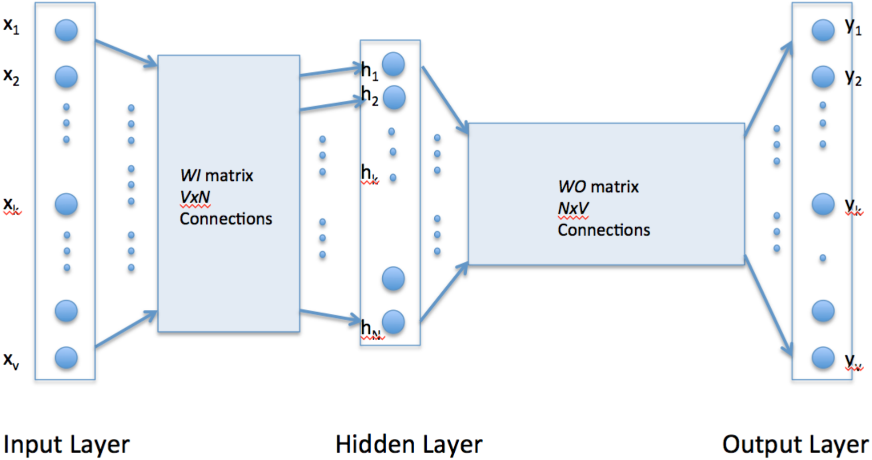
phone + mobile = cellphone

Why vector algebra works?

Let's recall the prediction model architecture:

Why vector algebra works?

Let's recall the prediction model architecture:



Why vector algebra works?

Why sum (element-wise addition) works?

Why vector algebra works?

Why sum (element-wise addition) works?

phone + mobile = cellphone

Why vector algebra works?

Why sum (element-wise addition) works?

phone + mobile = cellphone

- input vectors represent context distribution;

Why vector algebra works?

Why sum (element-wise addition) works?

phone + mobile = cellphone

- input vectors represent context distribution;
- they are related logarithmically to output probabilities;

Why vector algebra works?

Why sum (element-wise addition) works?

phone + mobile = cellphone

- input vectors represent context distribution;
- they are related logarithmically to output probabilities;
- the sum of 2 input vectors $\vec{x1}$ and $\vec{x2}$ is related to the product of 2 probability context distributions;

Why vector algebra works?

Why sum (element-wise addition) works?

phone + mobile = cellphone

- input vectors represent context distribution;
- they are related logarithmically to output probabilities;
- the sum of 2 input vectors $\vec{x}1$ and $\vec{x}2$ is related to the product of 2 probability context distributions;
- when multiplying 2 probability distributions, only words with high probabilities for both $\vec{x}1$ and $\vec{x}2$ will rank higher;

Why vector algebra works?

Why sum (element-wise addition) works?

phone + mobile = cellphone

- input vectors represent context distribution;
- they are related logarithmically to output probabilities;
- the sum of 2 input vectors $\vec{x}1$ and $\vec{x}2$ is related to the product of 2 probability context distributions;
- when multiplying 2 probability distributions, only words with high probabilities for both $\vec{x}1$ and $\vec{x}2$ will rank higher;
- ...sort of AND function;

Why vector algebra works?

Why sum (element-wise addition) works?

phone + mobile = cellphone

- input vectors represent context distribution;
- they are related logarithmically to output probabilities;
- the sum of 2 input vectors $\vec{x}1$ and $\vec{x}2$ is related to the product of 2 probability context distributions;
- when multiplying 2 probability distributions, only words with high probabilities for both $\vec{x}1$ and $\vec{x}2$ will rank higher;
- ...sort of AND function;
- if cellphone appears frequently as a context word for phone and mobile, it will become the answer.

Why vector algebra works?

Why analogical inference works?

Germany + Paris - France = Berlin

Why vector algebra works?

Why analogical inference works?

Germany + Paris - France = Berlin

- That's more difficult to explain.

Why vector algebra works?

Why analogical inference works?

Germany + Paris - France = Berlin

- That's more difficult to explain.
- [Levy et al., 2015] are not sure that relational information exists in contextual features at all.

Why vector algebra works?

Why analogical inference works?

Germany + Paris - France = Berlin

- That's more difficult to explain.
- [Levy et al., 2015] are not sure that relational information exists in contextual features at all.
- In fact, Germany + Paris produces Berlin as well.

Why vector algebra works?

Why analogical inference works?

Germany + Paris - France = Berlin

- That's more difficult to explain.
- [Levy et al., 2015] are not sure that relational information exists in contextual features at all.
- In fact, Germany + Paris produces Berlin as well.
- So it might be that in many cases there are no 'relations' at all, only finding nearest associates of the positive input words.

Why vector algebra works?

Why analogical inference works?

Germany + Paris - France = Berlin

- That's more difficult to explain.
- [Levy et al., 2015] are not sure that relational information exists in contextual features at all.
- In fact, Germany + Paris produces Berlin as well.
- So it might be that in many cases there are no 'relations' at all, only finding nearest associates of the positive input words.

But the idea is still attractive and produces empirical results.

Contents

Possible applications

Geometric directions between word embeddings are used in many sense-related applications.

Possible applications

Geometric directions between word embeddings are used in many sense-related applications.

Information extraction through semantic relations

- It is possible to find semantic relations like **A** is a capital of **B**, or **A** is located in **B**.

Possible applications

Geometric directions between word embeddings are used in many sense-related applications.

Information extraction through semantic relations

- It is possible to find semantic relations like **A** is a capital of **B**, or **A** is located in **B**.
- But it can be **A** is an object of **B**'s actions, for that matter.

Possible applications

Geometric directions between word embeddings are used in many sense-related applications.

Information extraction through semantic relations

- It is possible to find semantic relations like **A** is a capital of **B**, or **A** is located in **B**.
- But it can be **A** is an object of **B**'s actions, for that matter.
- We can even build full-fledged ontologies using this approach.

Possible applications

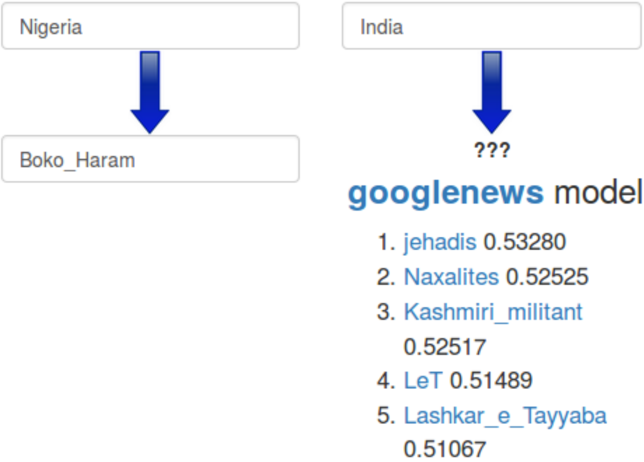
Geometric directions between word embeddings are used in many sense-related applications.

Information extraction through semantic relations

- It is possible to find semantic relations like **A** is a capital of **B**, or **A** is located in **B**.
- But it can be **A** is an object of **B**'s actions, for that matter.
- We can even build full-fledged ontologies using this approach.
- ...or find phenomena analogical to other phenomena in complex ways.

Possible applications

Unsupervised information extraction: militant groups across countries



Possible applications

Unsupervised information extraction: militant groups across countries

Possible applications

Unsupervised information extraction: militant groups across countries

Nigeria

Boko_Haram

Syria

???

googlenews model

1. Hezbollah 0.61445
2. Hizbullah 0.59423
3. Syrians 0.57959
4. Damascus 0.57835
5. Syrian 0.57829

Possible applications

Unsupervised information extraction: militant groups across countries

Try yourself at
`http://ltr.uio.no/semvec/calculator`

Possible applications

What next?

- We can cross the boundaries of one model and find links between languages (relation of ‘being a translation of’).

Possible applications

What next?

- We can cross the boundaries of one model and find links between languages (relation of ‘being a translation of’).
- ‘Machine translation’ for words, relying only on monolingual distributional models and small bilingual dictionaries.

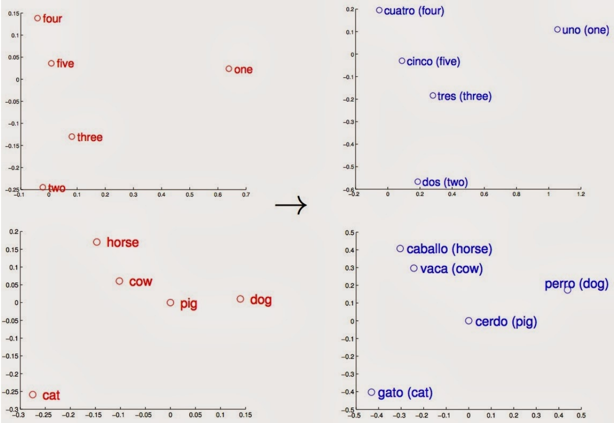
Possible applications

What next?

- We can cross the boundaries of one model and find links between languages (relation of 'being a translation of').
- 'Machine translation' for words, relying only on monolingual distributional models and small bilingual dictionaries.
- Let's see how it can be done

Possible applications

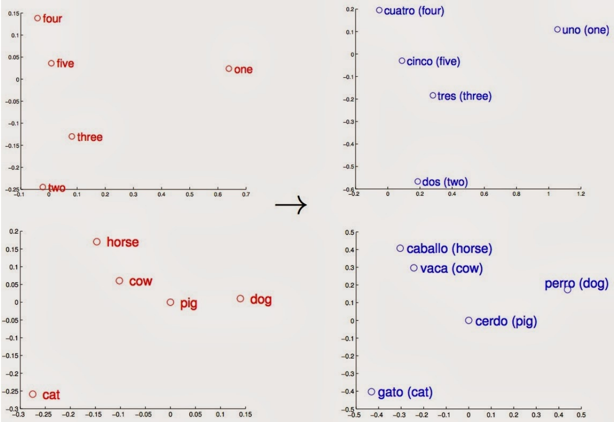
Inter-lingua?



[Mikolov et al., 2013a]

Possible applications

Inter-lingua?

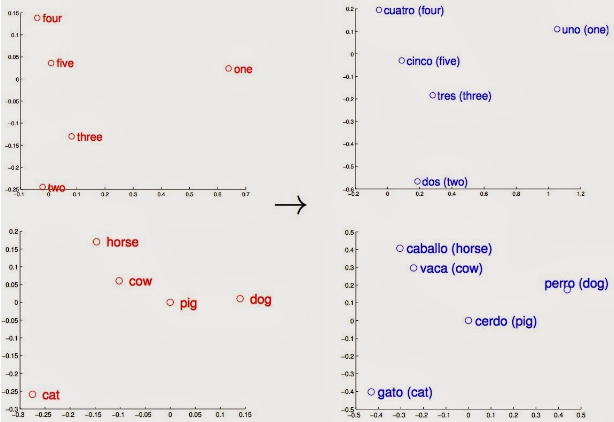


[Mikolov et al., 2013a]

Languages share concepts grounded in the real world

Possible applications

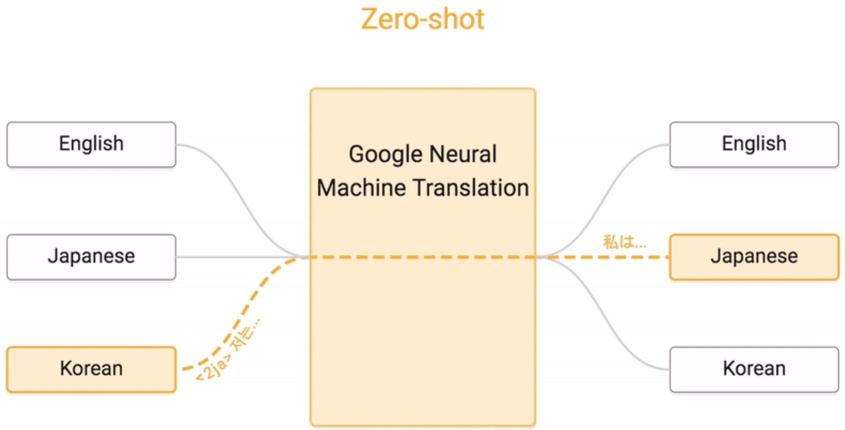
Inter-lingua?



[Mikolov et al., 2013a]

Languages share concepts grounded in the real world $\rightarrow$ after proper rotation and scaling, models (semantic spaces) trained on comparable corpora from different languages should 'map' into each other.

Possible applications



By the way, the recently announced Google's Zero-Shot neural machine translation model [Johnson et al., 2016] seems to rely on the same inter-lingua intuition.

Possible applications

The Mikolov's idea of 'distributional translation' is to find regular correspondences between vector spaces representing each language:

Possible applications

The Mikolov's idea of 'distributional translation' is to find regular correspondences between vector spaces representing each language:

1. train monolingual distributional models for languages L1 and L2;

Possible applications

The Mikolov's idea of 'distributional translation' is to find regular correspondences between vector spaces representing each language:

1. train monolingual distributional models for languages L1 and L2;
2. take a small bilingual dictionary L1→L2 with *n* frequent words present in both models;

Possible applications

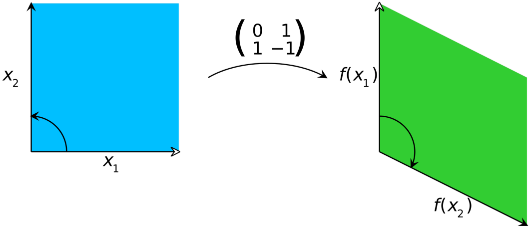
The Mikolov's idea of 'distributional translation' is to find regular correspondences between vector spaces representing each language:

1. train monolingual distributional models for languages L1 and L2;
2. take a small bilingual dictionary L1→L2 with *n* frequent words present in both models;
3. learn linear transformation matrix *M* between embeddings of dictionary words in L1 and their translations in L2;

Possible applications

The Mikolov's idea of 'distributional translation' is to find regular correspondences between vector spaces representing each language:

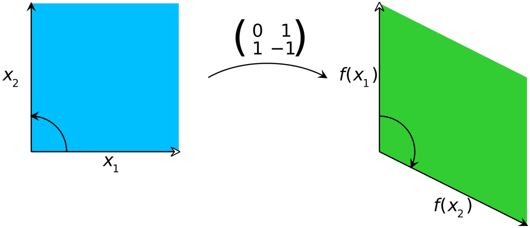
1. train monolingual distributional models for languages L1 and L2;
2. take a small bilingual dictionary L1→L2 with *n* frequent words present in both models;
3. learn linear transformation matrix *M* between embeddings of dictionary words in L1 and their translations in L2;



Possible applications

The Mikolov's idea of 'distributional translation' is to find regular correspondences between vector spaces representing each language:

1. train monolingual distributional models for languages L1 and L2;
2. take a small bilingual dictionary L1→L2 with *n* frequent words present in both models;
3. learn linear transformation matrix *M* between embeddings of dictionary words in L1 and their translations in L2;



But how we do this?

Contents

Projecting one model into another: models alignment

- Mapping one model into another by finding a linear transformation matrix M is an optimization problem of minimizing prediction error – linear regression, to be exact;
- to find M, one either:

Projecting one model into another: models alignment

- Mapping one model into another by finding a linear transformation matrix M is an optimization problem of minimizing prediction error – linear regression, to be exact;
- to find M, one either:
  1. learns the optimal weights iteratively (with gradient descent, etc..);

Projecting one model into another: models alignment

- Mapping one model into another by finding a linear transformation matrix M is an optimization problem of minimizing prediction error – linear regression, to be exact;
- to find M, one either:
  1. learns the optimal weights iteratively (with gradient descent, etc..);
  2. or solves the quadratic minimization problem by finding normal equation.

Projecting one model into another: models alignment

- Mapping one model into another by finding a linear transformation matrix M is an optimization problem of minimizing prediction error – linear regression, to be exact;
- to find M, one either:
  1. learns the optimal weights iteratively (with gradient descent, etc..);
  2. or solves the quadratic minimization problem by finding normal equation.
- normal equation approach is guaranteed to find the global optimum.

Projecting one model into another: models alignment

- Mapping one model into another by finding a linear transformation matrix M is an optimization problem of minimizing prediction error – linear regression, to be exact;
- to find M, one either:
  1. learns the optimal weights iteratively (with gradient descent, etc..);
  2. or solves the quadratic minimization problem by finding normal equation.
- normal equation approach is guaranteed to find the global optimum.

$$\beta_i = (\mathbf{X}^\intercal \times \mathbf{X})^{-1} \times \mathbf{X}^\intercal \times y_i \tag{3}$$

$\mathbf{X}$ is the matrix of L1 dictionary word embeddings (input), y_i is the vector of the i^{th} components of corresponding L2 dictionary words (correct predictions), and β_i is our aim: the vector of optimal coefficients which transform an L1 embedding into the i^{th} component of the L2 embedding.

Projecting one model into another: models alignment

After solving such normal equations for all the components (their number is equal to the L2 embeddings dimensionality *i*), we have the matrix *M* which fits the data best:

Projecting one model into another: models alignment

After solving such normal equations for all the components (their number is equal to the L2 embeddings dimensionality i), we have the matrix M which fits the data best:

$$\sum_{a=1}^{n}(M\vec{x_a} - \vec{y_a})^2 \qquad (4)$$

is minimal over all training data (n dictionary pairs).

Projecting one model into another: models alignment

After solving such normal equations for all the components (their number is equal to the L2 embeddings dimensionality i), we have the matrix M which fits the data best:

$$\sum_{a=1}^{n} (M\vec{x}_a - \vec{y}_a)^2 \tag{4}$$

is minimal over all training data (n dictionary pairs).

- For any unseen word in L1, you can now multiply its vector $\vec{x}$ in the L1 model by M: $\vec{z} = M\vec{x}$.

Projecting one model into another: models alignment

After solving such normal equations for all the components (their number is equal to the L2 embeddings dimensionality i), we have the matrix M which fits the data best:

$$\sum_{a=1}^{n}(M\vec{x}_a - \vec{y}_a)^2 \tag{4}$$

is minimal over all training data (n dictionary pairs).

- For any unseen word in L1, you can now multiply its vector $\vec{x}$ in the L1 model by M: $\vec{z} = M\vec{x}$.
- $\vec{z}$ is the translation vector;

Projecting one model into another: models alignment

After solving such normal equations for all the components (their number is equal to the L2 embeddings dimensionality i), we have the matrix M which fits the data best:

$$\sum_{a=1}^{n}(M\vec{x_a} - \vec{y_a})^2 \tag{4}$$

is minimal over all training data (n dictionary pairs).

- For any unseen word in L1, you can now multiply its vector $\vec{x}$ in the L1 model by M: $\vec{z} = M\vec{x}$.
- $\vec{z}$ is the translation vector;
- Find the word in L2 with the embedding most similar to $\vec{z}$

Projecting one model into another: models alignment

After solving such normal equations for all the components (their number is equal to the L2 embeddings dimensionality i), we have the matrix M which fits the data best:

$$\sum_{a=1}^{n} (M\vec{x_a} - \vec{y_a})^2 \tag{4}$$

is minimal over all training data (n dictionary pairs).

- For any unseen word in L1, you can now multiply its vector $\vec{x}$ in the L1 model by M: $\vec{z} = M\vec{x}$.
- $\vec{z}$ is the translation vector;
- Find the word in L2 with the embedding most similar to $\vec{z}$
- Use that word as a translation.

Projecting one model into another: models alignment

Results

[Mikolov et al., 2013a] report accuracy about 0.5 @1 and 0.75 @5 for *English*→*Spanish* translation.

Projecting one model into another: models alignment

Results

[Mikolov et al., 2013a] report accuracy about 0.5 @1 and 0.75 @5 for *English→Spanish* translation.

Our results for *Ukrainian→Russian*:

	CBOW		SkipGram		Edit distance
	Training	Test	Training	Test	
@1	0.648	0.57	0.545	0.374	0.549
@5	0.764	0.658	0.644	0.486	0.619

Projecting one model into another: models alignment

Results

[Mikolov et al., 2013a] report accuracy about 0.5 @1 and 0.75 @5 for *English*→*Spanish* translation.

Our results for *Ukrainian*→*Russian*:

	CBOW		SkipGram		Edit distance
	Training	Test	Training	Test	
@1	0.648	0.57	0.545	0.374	0.549
@5	0.764	0.658	0.644	0.486	0.619

By the way, semantic fingerprints can be projected into another semantic space using the same transformation matrix!

Projecting one model into another: models alignment

Results

[Mikolov et al., 2013a] report accuracy about 0.5 @1 and 0.75 @5 for *English→Spanish* translation.

Our results for *Ukrainian→Russian*:

	CBOW		SkipGram		Edit distance
	Training	Test	Training	Test	
@1	0.648	0.57	0.545	0.374	0.549
@5	0.764	0.658	0.644	0.486	0.619

By the way, semantic fingerprints can be projected into another semantic space using the same transformation matrix!
It means we can ‘translate’ whole documents.

Projecting one model into another: models alignment

- Analogical inference can be interpreted as linear regression as well:

Projecting one model into another: models alignment

- Analogical inference can be interpreted as linear regression as well:
- We learn a regression matrix from a pair of example instances *('France $\rightarrow$Paris')* and apply it to a test instance *('Germany')*

Projecting one model into another: models alignment

- Analogical inference can be interpreted as linear regression as well:
- We learn a regression matrix from a pair of example instances *('France →Paris')* and apply it to a test instance *('Germany')*
- Say, we have example vectors [3,5] and [7,2], and a test instance [4,4], which should map into [8,1]

Projecting one model into another: models alignment

- Analogical inference can be interpreted as linear regression as well:
- We learn a regression matrix from a pair of example instances *('France $\rightarrow$Paris')* and apply it to a test instance *('Germany')*
- Say, we have example vectors [3,5] and [7,2], and a test instance [4,4], which should map into [8,1]
- Linear regression will 'learn' weights [2.3, 0.4], and for the test instance it will output [9.2, 0.8], pretty close to the desired result [8,1].

Projecting one model into another: models alignment

What else?

- Cross-model projections can be used not only for MT.

Projecting one model into another: models alignment

What else?

- Cross-model projections can be used not only for MT.
- We can study models sharing the same vocabulary (if there are reasons to think that the semantics might be different).

Projecting one model into another: models alignment

What else?

- Cross-model projections can be used not only for MT.
- We can study models sharing the same vocabulary (if there are reasons to think that the semantics might be different).

Detecting semantic shifts

- Suppose we want to trace how word meaning changed throughout time;

Projecting one model into another: models alignment

What else?

- Cross-model projections can be used not only for MT.
- We can study models sharing the same vocabulary (if there are reasons to think that the semantics might be different).

Detecting semantic shifts

- Suppose we want to trace how word meaning changed throughout time;
- Can be useful for diachronic linguistics...

Projecting one model into another: models alignment

What else?

- Cross-model projections can be used not only for MT.
- We can study models sharing the same vocabulary (if there are reasons to think that the semantics might be different).

Detecting semantic shifts

- Suppose we want to trace how word meaning changed throughout time;
- Can be useful for diachronic linguistics...
- ...or for extracting new events from news texts.

Projecting one model into another: models alignment

What else?

- Cross-model projections can be used not only for MT.
- We can study models sharing the same vocabulary (if there are reasons to think that the semantics might be different).

Detecting semantic shifts

- Suppose we want to trace how word meaning changed throughout time;
- Can be useful for diachronic linguistics...
- ...or for extracting new events from news texts.
- We train distributional models on time-separated corpora:

Projecting one model into another: models alignment

What else?

- Cross-model projections can be used not only for MT.
- We can study models sharing the same vocabulary (if there are reasons to think that the semantics might be different).

Detecting semantic shifts

- Suppose we want to trace how word meaning changed throughout time;
- Can be useful for diachronic linguistics...
- ...or for extracting new events from news texts.
- We train distributional models on time-separated corpora:
  - XIX century

Projecting one model into another: models alignment

What else?

- Cross-model projections can be used not only for MT.
- We can study models sharing the same vocabulary (if there are reasons to think that the semantics might be different).

Detecting semantic shifts

- Suppose we want to trace how word meaning changed throughout time;
- Can be useful for diachronic linguistics...
- ...or for extracting new events from news texts.
- We train distributional models on time-separated corpora:
  - XIX century
  - XX century

Projecting one model into another: models alignment

What else?

- Cross-model projections can be used not only for MT.
- We can study models sharing the same vocabulary (if there are reasons to think that the semantics might be different).

Detecting semantic shifts

- Suppose we want to trace how word meaning changed throughout time;
- Can be useful for diachronic linguistics...
- ...or for extracting new events from news texts.
- We train distributional models on time-separated corpora:
  - XIX century
  - XX century
  - XXI century

Projecting one model into another: models alignment

What else?

- Cross-model projections can be used not only for MT.
- We can study models sharing the same vocabulary (if there are reasons to think that the semantics might be different).

Detecting semantic shifts

- Suppose we want to trace how word meaning changed throughout time;
- Can be useful for diachronic linguistics...
- ...or for extracting new events from news texts.
- We train distributional models on time-separated corpora:
  - XIX century
  - XX century
  - XXI century
- Another application is comparing word meanings in different corpora (for example, fiction texts and web pages).

Projecting one model into another: models alignment

Houston, we have a problem

- But the resulting dense embeddings are not directly comparable (unlike traditional sparse count-based vectors)!

Projecting one model into another: models alignment

Houston, we have a problem

- But the resulting dense embeddings are not directly comparable (unlike traditional sparse count-based vectors)!
- Because of stochastic nature of our modeling, we cannot simply measure cosine distance between w_{xix}, w_{xx} and w_{xxi} to find whether the meaning has changed.

Projecting one model into another: models alignment

Houston, we have a problem

- But the resulting dense embeddings are not directly comparable (unlike traditional sparse count-based vectors)!
- Because of stochastic nature of our modeling, we cannot simply measure cosine distance between w_{xix}, w_{xx} and w_{xxi} to find whether the meaning has changed.
- What can be done?

Projecting one model into another: models alignment

How to compare incomparable?

- One can do the same: learn transformation matrix.

Projecting one model into another: models alignment

How to compare incomparable?

- One can do the same: learn transformation matrix.
- One can stick to comparing ranked lists of nearest associates, abstracting away from the vector values.

Projecting one model into another: models alignment

How to compare incomparable?

- One can do the same: learn transformation matrix.
- One can stick to comparing ranked lists of nearest associates, abstracting away from the vector values.
- One can transform the semantic space A so that it matches the semantic space B as closely as possible for the shared words, at the same time retaining pair-wise similarities.

Projecting one model into another: models alignment

How to compare incomparable?

- One can do the same: learn transformation matrix.
- One can stick to comparing ranked lists of nearest associates, abstracting away from the vector values.
- One can transform the semantic space A so that it matches the semantic space B as closely as possible for the shared words, at the same time retaining pair-wise similarities.
- For example, using the orthogonal Procrustes transformation:

Projecting one model into another: models alignment

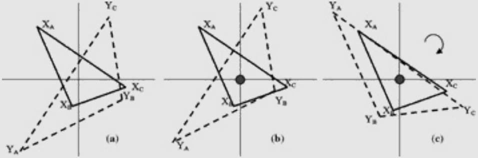
How to compare incomparable?

- One can do the same: learn transformation matrix.
- One can stick to comparing ranked lists of nearest associates, abstracting away from the vector values.
- One can transform the semantic space A so that it matches the semantic space B as closely as possible for the shared words, at the same time retaining pair-wise similarities.
- For example, using the orthogonal Procrustes transformation:
  - basically applying SVD to the dot product ($B \cdot A^{\intercal}$)

Projecting one model into another: models alignment

How to compare incomparable?

- One can do the same: learn transformation matrix.
- One can stick to comparing ranked lists of nearest associates, abstracting away from the vector values.
- One can transform the semantic space A so that it matches the semantic space B as closely as possible for the shared words, at the same time retaining pair-wise similarities.
- For example, using the orthogonal Procrustes transformation:
  - basically applying SVD to the dot product $(B \cdot A^{\intercal})$



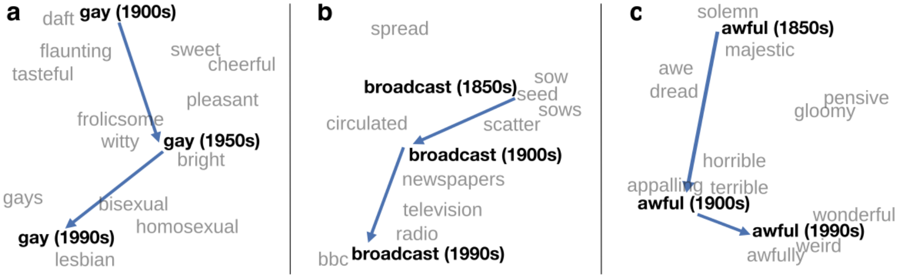
Orthogonal Procrustes (from [Schneider and Borlund, 2007])

Projecting one model into another: models alignment

Once the models are aligned, cosine distances between words from different models (across models boundaries) become meaningful. Semantic shifts can be traced:

Projecting one model into another: models alignment

Once the models are aligned, cosine distances between words from different models (across models boundaries) become meaningful. Semantic shifts can be traced:

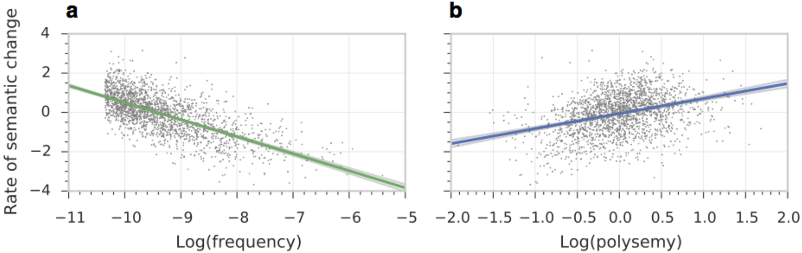


Projecting one model into another: models alignment

Such results can be produced even by comparing nearest associates lists. But models alignment allows uncovering statistical laws of semantic change on large scale:

Projecting one model into another: models alignment

Such results can be produced even by comparing nearest associates lists. But models alignment allows uncovering statistical laws of semantic change on large scale:



See more at `http://nlp.stanford.edu/projects/histwords/` and in [Hamilton et al., 2016].

Projecting one model into another: models alignment

Event extraction

- While updating a model with new data, continue aligning the previous model with the current one;

Projecting one model into another: models alignment

Event extraction

- While updating a model with new data, continue aligning the previous model with the current one;
- monitor changes in semantic relations;

Projecting one model into another: models alignment

Event extraction

- While updating a model with new data, continue aligning the previous model with the current one;
- monitor changes in semantic relations;
- detect current events:

Projecting one model into another: models alignment

Event extraction

- While updating a model with new data, continue aligning the previous model with the current one;
- monitor changes in semantic relations;
- detect current events:
- ‘The country A is no more in adversary relations to the country B’.

Projecting one model into another: models alignment

Event extraction

- While updating a model with new data, continue aligning the previous model with the current one;
- monitor changes in semantic relations;
- detect current events:
- 'The country A is no more in adversary relations to the country B'.
- Lots of opportunities for automatizing the handling of large textual data.

Projecting one model into another: models alignment

Questions?

INF5820
Distributional Semantics: Extracting Meaning from Data
Lecture 5
Kings and queens, men and women:
semantic relations between word embeddings

Contents

In the next week (last lecture)

What’s going on: recent advances and trends in the word embeddings world

- Discussion on the results of the obligatory assignment.
- The exam: what to expect?
- Multilingual word embeddings.
- Language and vision embeddings.
- etc...

References I

Hamilton, L. W., Leskovec, J., and Jurafsky, D. (2016).
Diachronic word embeddings reveal statistical laws of semantic change.
In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1501. Association for Computational Linguistics.

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., et al. (2016).
Google's multilingual neural machine translation system: Enabling zero-shot translation.
arXiv preprint arXiv:1611.04558.

References II

Levy, O., Remus, S., Biemann, C., and Dagan, I. (2015).
Do supervised distributional methods really learn lexical inference relations?
In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 970–976. Association for Computational Linguistics.

Mikolov, T., Le, Q., and Sutskever, I. (2013a).
Exploiting similarities among languages for machine translation.
arXiv preprint arXiv:1309.4168.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b).
Distributed representations of words and phrases and their compositionality.
Advances in Neural Information Processing Systems 26.

References III

Pennington, J., Socher, R., and Manning, C. D. (2014).
GloVe: Global vectors for word representation.
In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Schneider, J. W. and Borlund, P. (2007).
Matrix comparison, part 2: Measuring the resemblance between proximity measures or ordination results by use of the Mantel and Procrustes statistics.
Journal of the American Society for Information Science and Technology, 58(11):1596–1609.

Van der Maaten, L. and Hinton, G. (2008).
Visualizing data using t-SNE.
Journal of Machine Learning Research, 9(2579-2605):85.