# — INF4820 —
# Algorithms for AI and NLP

## *Semantic Spaces*

Murhaf Fares & Stephan Oepen

Language Technology Group (LTG)

September 22, 2016

# “You shall know a word by the company it keeps!”

- Alcazar?

- The alcazar did not become a permanent residence for the royal family until 1905
- The alcazar was built in the tenth century
- You can also visit the alcazar while the royal family is there

# Vector space semantics

- Can a program reuse the same intuition to automatically learn word meaning?
  - By looking at data of actual language use
  - and without any prior knowledge
- How can we represent word meaning in a mathematical model?

## Concepts

- Distributional semantics
- Vector spaces
- Semantic spaces

# The distributional hypothesis

## AKA the contextual theory of meaning

- *Meaning is use.* (Wittgenstein, 1953)
- *You shall know a word by the company it keeps.* (Firth, 1957)
- *The meaning of entities, and the meaning of grammatical relations among them, is related to the restriction of combinations of these entities relative to other entities.* (Harris, 1968)

# The distributional hypothesis (cont'd)

- The hypothesis: If two words share similar contexts, we can assume that they have similar meanings.
- Comparing meaning reduced to comparing contexts,
  – no need for prior knowledge!
- Our goal: to automatically learn word semantics based on this hypothesis.

# Distributional semantics in practice

**A distributional approach to lexical semantics:**

- Given the set of words in our vocabulary $|V|$
- Record contexts of words across a large collection of texts (corpus).
- Each word is represented by a set of contextual features.
- Each feature records some property of the observed contexts.
- Words that are found to have similar features are expected to also have similar meaning.

# Distributional semantics in practice - first things first

- The hypothesis: If two words share similar contexts, we can assume that they have similar meanings.
- How do we define *word*?
- How do we define *context*?
- How do we define *similar*?

# What is a word?

| | |
|---|---|
| Raw: | “The programmer’s programs had been programmed.” |
| Tokenized: | the programmer ’s programs had been programmed . |
| Lemmatized: | the programmer ’s program have be program . |
| W/ stop-list: | programmer program program |
| Stemmed: | program program program |

- Tokenization: Splitting a text into sentences and words or other units.
- Different levels of abstraction and morphological normalization:
  - What to do with case, numbers, punctuation, compounds, . . . ?
  - Full-form words vs. lemmas vs. stems . . .
- Stop-list: filter out closed-class words or function words.
  - The idea is that only *content words* provide relevant context.

# Token vs. type

... Tunisian or French cakes and it is marketed. The bread may be cooked such as Kessra or Khmira or Harchaya ...

... Chile, cochayuyo. Laver is used to make laver bread in Wales where it is known as" bara lawr"; in ...

... and how everyday events such as a Samurai cutting bread with his sword are elevated to something special and ...

... used to make the two main food staples of bread and beer. Flax plants, uprooted before they started flowering ...

... for milling grain and a small oven for baking the bread. Walls were painted white and could be covered with dyed ...

... of the ancients. The staple diet consisted of bread and beer, supplemented with vegetables such as onions and garlic ...

... Prayers were made to the goddess Isis. Moldy bread, honey and copper salts were also used to prevent ...

... going souling and the baking of special types of bread or cakes. In Tirol, cakes are left for them on the table ...

... under bridges, beg in the streets, and steal loaves of bread. If the path be beautiful, let us not question where it ...

... When Jesus the Christ, who is the Word and the bread of Life, comes a second time, the righteous will be raised ...

# Token vs. type

"Rose is a rose is a rose is a rose." *Gertrude Stein*

Three types and ten tokens.

# Defining ‘context’

- Let’s say we’re extracting (contextual) features for the target *bread* in:

I bake bread for breakfast.

## Context windows

- Context $\equiv$ neighborhood of $\pm n$ words left/right of the focus word.
- Features for $\pm 1$: {`left:bake`, `right:for`}
- Some variants: distance weighting, $n$grams.

## Bag-of-Words (BoW)

- Context $\equiv$ all co-occurring words, ignoring the linear ordering.
- Features: {`I`, `bake`, `for`, `breakfast`}
- Some variants: sentence-level, document-level.

# Defining ‘context’ (cont’d)

I bake bread for breakfast.

## Grammatical context

- Context $\equiv$ the grammatical relations to other words.
- Intuition: When words combine in a construction they often impose semantic constraints on each other:

  . . . *to* {*drink* | *pour* | *spill*} *some* {*milk* | *water* | *wine*} . . .
- Features: {`dir_obj(bake)`, `prep_for(breakfast)`}
- Requires deeper linguistic analysis than simple BoW approaches.

# Different contexts → different similarities

- What do we mean by *similar*?
- *car, road, gas, service, traffic, driver, license*
- *car, train, bicycle, truck, vehicle, airplane, bus*
- Relatedness vs. sameness. Or domain vs. content. Or syntagmatic vs. paradigmatic.
- Similarity in domain: {*car, road, gas, service, traffic, driver, license*}
- Similarity in content: {*car, train, bicycle, truck, vehicle, airplane, bus*}
- The type of context dictates the type of semantic similarity.
- Broader definitions of context tend to give clues for *domain-based relatedness*.
- Fine-grained and linguistically informed contexts give clues for *content-based similarity*.

# Representation – Vector space model

- Given the different definitions of ‘word’, ‘context’ and ‘similarity’:
- How exactly should we represent our words and context features?
- How exactly can we compare the features of different words?

# Distributional semantics in practice

**A distributional approach to lexical semantics:**

- Record contexts of words across a large collection of texts (corpus).
- Each word is represented by a set of contextual features.
- Each feature records some property of the observed contexts.
- Words that are found to have similar features are expected to also have similar meaning.

# Vector space model

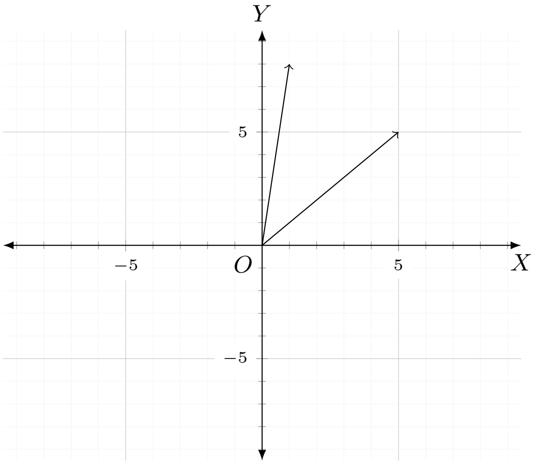
- Vector space models first appeared in IR.
- A general algebraic model for representing data based on a spatial metaphor.
- Each object is represented as a vector (or point) positioned in a coordinate system.
- Each coordinate (or dimension) of the space corresponds to some descriptive and measurable property (feature) of the objects.
- To measure similarity of two objects, we can measure their geometrical distance / closeness in the model.
- Vector representations are foundational to a wide range of ML methods.

# Vectors and vector spaces

- A vector space is defined by a system of $n$ dimensions or coordinates where points are represented as real-valued vectors in the space $\Re^n$.

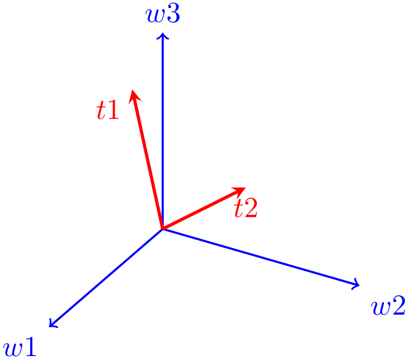
- The most basic example is 2-dimensional Euclidean plane $\Re^2$.

$$v1 = [5, 5],\ v2 = [1, 8]$$



# Semantic spaces

- AKA distributional semantic models or word space models.
- A semantic space is a vector space model where
  - points represent words,
  - dimensions represent context of use,
  - and distance in the space represents semantic similarity.




Dimensions: $w1, w2, w3$

$t1 = [2, 1, 2] \in \Re^3$

$t2 = [1, 1, 1] \in \Re^3$

# Feature vectors

- Each word type $t_i$ is represented by a vector of real-valued features.
- Our observed feature vectors must be encoded numerically:
  - Each context feature is mapped to a dimension $j \in [1, n]$.
  - For a given word, the value of a given feature is its number of co-occurrences for the corresponding context across our corpus.
- Let the set of $n$ features describing the lexical contexts of a word $t_i$ be represented as a feature vector $\vec{x}_i = \langle x_{i1}, \ldots, x_{in} \rangle$.

## Example

- Given a grammatical context, if we assume that:
- the $i$th word is *bread* and
- the $j$th feature is `OBJ_OF(bake)`, then
- $x_{ij} = 4$ would mean that we have observed *bread* to be the object of the verb *bake* in our corpus 4 times.

# Euclidean distance

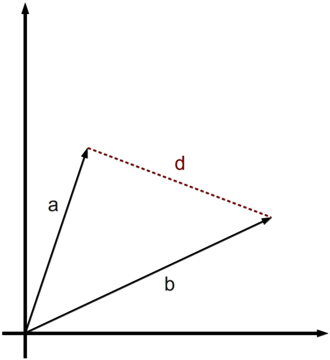
- We can now compute *semantic similarity* in terms of *spatial distance*.
- One standard metric for this is the *Euclidean distance*:

$$d(\vec{a},\vec{b}) = \sqrt{\sum_{i=1}^{n} \left(\vec{a}_i - \vec{b}_i\right)^2}$$

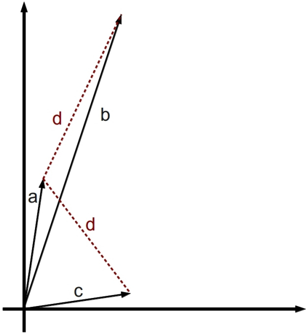
- Computes the norm (or *length*) of the *difference* of the vectors.
- The norm of a vector is:

$$\|\vec{x}\| = \sqrt{\sum_{i=1}^{n} \vec{x}_i^2} = \sqrt{\vec{x}\cdot\vec{x}}$$

- Intuitive interpretation: The distance between two points corresponds to the length of the straight line connecting them.



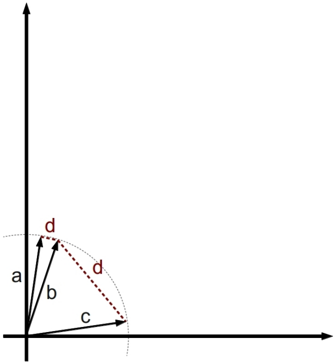
# Euclidean distance and length bias




- a: automobile
- b: car
- c: road
- $d(\vec{a}, \vec{b}) = 10$
- $d(\vec{a}, \vec{c}) = 7$

- However, a potential problem with Euclidean distance is that it is very sensitive to extreme values and the length of the vectors.
- As vectors of words with different *frequencies* will tend to have different length, the frequency will also affect the similarity judgment.

# Overcoming length bias by normalization



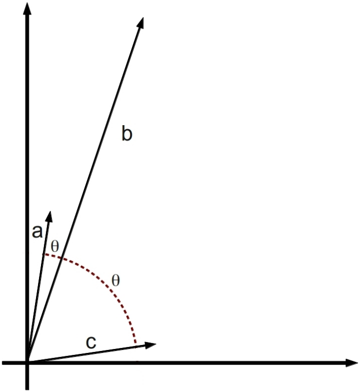

- One way to reduce frequency effects is to first normalize all our vectors to have unit length, i.e. $\|\vec{x}\| = 1$
- Can be achieved by simply dividing each element by the length: $\vec{x}\frac{1}{\|\vec{x}\|}$
- Amounts to all vectors pointing to the surface of a unit sphere.

# Cosine similarity

- We can measure (cosine) *proximity* rather than (Euclidean) *distance*.
- Computes similarity as a function of the angle between the vectors:

$$\cos(\vec{a}, \vec{b}) = \frac{\sum_i \vec{a}_i \vec{b}_i}{\sqrt{\sum_i \vec{a}_i^2} \sqrt{\sum_i \vec{b}_i^2}} = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|}$$

- Constant range between 0 and 1.
- Avoids the arbitrary scaling caused by dimensionality, frequency, etc.
- As the angle between the vectors shortens, the cosine approaches 1.



# Cosine similarity (cont'd)

- For *normalized* (unit) vectors, the cosine is simply the *dot product*:

  $\cos(\vec{a}, \vec{b}) = \vec{a} \cdot \vec{b} = \sum_{i=1}^{n} \vec{a}_i \vec{b}_i$

- Can be computed very efficiently.
- The *same relative rank order* as the Euclidean distance for unit vectors!

# Practical comments: Co-occurrence matrix

- Conceptually, a vector space is often thought of as a matrix, often called co-occurrence matrix or word-context matrix.
  - Dimensions correspond to columns; each feature vector is a row.
  - For $m$ words and $n$ features we have an $m \times n$ co-occurrence matrix.

## Corpus

- An automobile is a wheeled motor vehicle used for transporting passengers .
- A car is a form of transport, usually with four wheels and the capacity to carry around five passengers .
- Transport for the London games is limited , with spectators strongly advised to avoid the use of cars .

| | advise | avoid | capacity | carry | ... | vehicle | wheel | ... |
|---|---|---|---|---|---|---|---|---|
| automobile | 0 | 0 | 0 | 0 | ... | 1 | 1 | |
| car | 1 | 1 | 1 | 1 | ... | 0 | 1 | |

# Practical comments: Sparsity

- As we move towards more realistic set-ups:
  - Semantic spaces will be extremely high-dimensional
  - The number of *non-zero* elements will be very low.
  - Few active features per word.
- We say that the vectors are sparse.
- This has implications for how to implement our data structures and vector operations:
- Don't want to waste space representing zero-valued features.

# Practical comments: Vector operations

- In theory, you can view formulas like Euclidean norm and cosine as "pseudo-code" that you can translate directly into Lisp.
- But again; our feature vectors are sparse.
- Taken directly, a formula like the Euclidean norm requires iterating over every dimension $n$ in our space.
- But we don't want to waste time iterating over zero elements if we don't have to!

# Word–context association

- Problem: Raw co-occurrence frequencies are not very discriminative, and therefore not always the best indicators of relevance.
- Imagine we have some features recording information about direct objects and we've collected the following counts for the noun *wine*:
  - $\texttt{OBJ\_OF}(\texttt{buy}) = 14$
  - $\texttt{OBJ\_OF}(\texttt{pour}) = \ 8$
  - ...but the feature $\texttt{OBJ\_OF}(\texttt{pour})$ seems more indicative of the semantics of *wine* than $\texttt{OBJ\_OF}(\texttt{buy})$.
- Solution: Weight the counts by an *association function*, "normalizing" our observed frequencies for chance co-occurrence.
- A range of different tests of statistical are used; e.g. *pointwise mutual information*, *log odds ratio*, *the t-test*, *log likelihood*, ...
- Note: We'll skip this step in our implementation (assignment 2a).

# Vector space models in IR

- So far we've looked at vector space models for detecting *words* with similar *meanings*.
- It's important to realize that vector space models are widely used for other purposes as well.
- Vector space models are commonly used in IR for finding *documents* with similar *content*.
- Each document $d_j$ is represented by a feature vector, with features corresponding to the terms $t_1, \ldots, t_n$ occurring in the documents.
- Spatial distance $\approx$ similarity of content.
- Can also represent a search query as a vector.
- The relevance of documents given by their distance to the query.

# Vector space models in IR (cont'd)

- The most commonly used weighting function is $\text{tf-idf}$:
  - The term frequency $\text{tf}(t_i, d_j)$ denotes the number of times the term $t_i$ occurs in document $d_j$.
  - The document frequency $\text{df}(t_i)$ denotes the total number of documents in the collection that the term occurs in.
  - The inverse document frequency is defined as $\text{idf}(\text{t}_\text{i}) = \log\left(\frac{N}{df(t_i)}\right)$, where $N$ is the total number of documents in the collection.
  - The weight given to term $t_i$ in document $d_j$ is then computed as

$$\text{tf-idf}(t_i, d_j) = \text{tf}(t_i, d_j) \times \text{idf}(t_i)$$

  - A high $\text{tf-idf}$ is obtained if a term has a *high* frequency in the given *document* and a *low* frequency in the document *collection* as whole.
  - The weights hence tend to filter out common terms.

# Conclusions

- Word meaning can be represented as a vector characterized by $n$ dimensions.
- The $n$ dimensions of our feature vectors represent the contextual features we observe.
- Raw co-occurrence counts are good but not the best way to quantify relevance.
- Semantic similarity can be computed based on spatial distance and proximity.
- We need to be careful when deciding on a data structure to represent the co-occurrence matrix and when we implement vector operations.

# Next week

- Computing neighbor relations in the semantic space
- Representing classes
- Representing class membership
- Classification algorithms: KNN-classification / $c$-means, etc.

Firth, J. R. (1957). A synopsis of linguistic theory 1930–1955. In *Studies in linguistic analysis.* Philological Society, Oxford.

Harris, Z. S. (1968). *Mathematical structures of language*. New York: Wiley.

Wittgenstein, L. (1953). *Philosophical investigations*. Oxford: Blackwell.