Distributional Semantics
Lab Session

Diego Frassinelli

November 16-17 2019

The main goal of this lab session is to make you familiarize with the concept of *distributional semantics*. In this session, you will compute the similarity between words both using pen&paper and in an automatic way. You can continue working on these tasks in tomorrow's session too.

Practicalities

Try to work in pairs, write the code by yourself but discuss with the others the results you obtain. This would make this session more interesting and definitely more fruitful. When you are asked to write your own code, feel free to chose the language you feel more comfortable with. We will discuss the results of this handout in the next lab session. In this tutorial you will find a combination of questions that are useful to recap some key topics of the previous lectures (Review) and some practical exercises that make you familiarise with real data.

Task 1

According to distributional semantics, it is possible to capture semantic similarities between words based on their distributional properties in large corpora.

1. **Review:** what does the Distributional Hypothesis state?
2. **Review:** what are the main differences between syntagmatic and paradigmatic word space models according to the Peirsman et al. (2008) paper?

Using the toy corpus in the box below, you will build a vector representation of the target words: "Batman", "Wayne" and "Joker".

> Batman is an American superhero. The secret identity of Batman is Bruce Wayne, an American billionaire from Gotham City. The Joker is a supervillain that embodies the ideas of anarchy and chaos. The Joker and Batman fight the battle for Gotham's soul.

Let's manually compute (pen&paper) the similarity between the three target words using a first order bag-of-words distributional semantic model. Your task is to discover if it is more probable to find Wayne behind the mask of Batman or behind the painted face of the Joker.

1. Build a target-by-context matrix with the following constraints:
    - the text has to be lemmatised;
    - only nouns, verbs and adjectives are context words;
    - do not use any stop-word list;

- context words occur in a symmetrical window of 3 words on each side;
- collocations are sentence-based (e.g., the window does not continue after a full stop);
- vector dimensions are counts.

2. Compute manually the cosine similarity between the vector for Batman and the one for Wayne, and the vector for Joker and the one for Wayne.

$$\text{sim}_{\text{cosine}} = (\vec{v}, \vec{w}) = \frac{\vec{v} \cdot \vec{w}}{|\vec{v}||\vec{w}|} \qquad \text{where: } |\vec{v}| = \sqrt{\sum_i^N (v_i^2)}$$

3. Based on the small corpus provided, what is the most probable (the most semantically similar one) second identity of Bruce Wayne?

4. Discuss your answer:
   (a) what is the range of possible values a cosine can have?
   (b) what high cosine values (closer to 1) indicate?
   (c) why is cosine a good measure for computing semantic similarity scores?

Task 2

Let's now write a script that computes the cosine similarity between the pairs of vectors you have already manually generated in the previous task. When you write this code, try to make it working also on bigger chunks of text.

1. Copy the text from in the toy corpus (Task 1) to the new file corpus.txt.
2. You need the lemma and POS of each word in order to construct your matrix.
3. Build the vector representations for the three target words.
4. Compute the cosine similarity between the two pairs of targets (Batman - Wayne, Joker - Wayne) without using existing libraries.
5. Did you get the same results as in Task 1?

Task 3

Word embeddings are becoming very popular in distributional semantics.

1. **Review:** Do you remember what these embeddings are and why they are so popular?

I have downloaded the GoogleNews pre-trained embeddings from `https://code.google.com/archive/p/word2vec/` and extracted the vector representations of the three targets (Batman, Wayne, Joker). You find the embedding for the three targets in: GoogleNews-subset. If you are curious to see one way to extract embeddings from the downloaded file, look at: gensimExtraction.html (on ILIAS).

1. **Review:** Read the description of the GoogleNews embeddings.
2. Compute the similarity between the embeddings of "Batman" and "Wayne" and "Joker" and "Wayne" adapting your existing code.
3. Compare the results with your previous computation.

Task 4

- As it is often the case, it is possible to find existing libraries that can do the hard work for you.
- Go to `https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics.pairwise` and check the functions that compute similarity scores between vectors. Note that this is just one possible library to compute similarity scores.
- Using this library, compute cosine similarity and check if the resulting scores correspond to the values you computed.