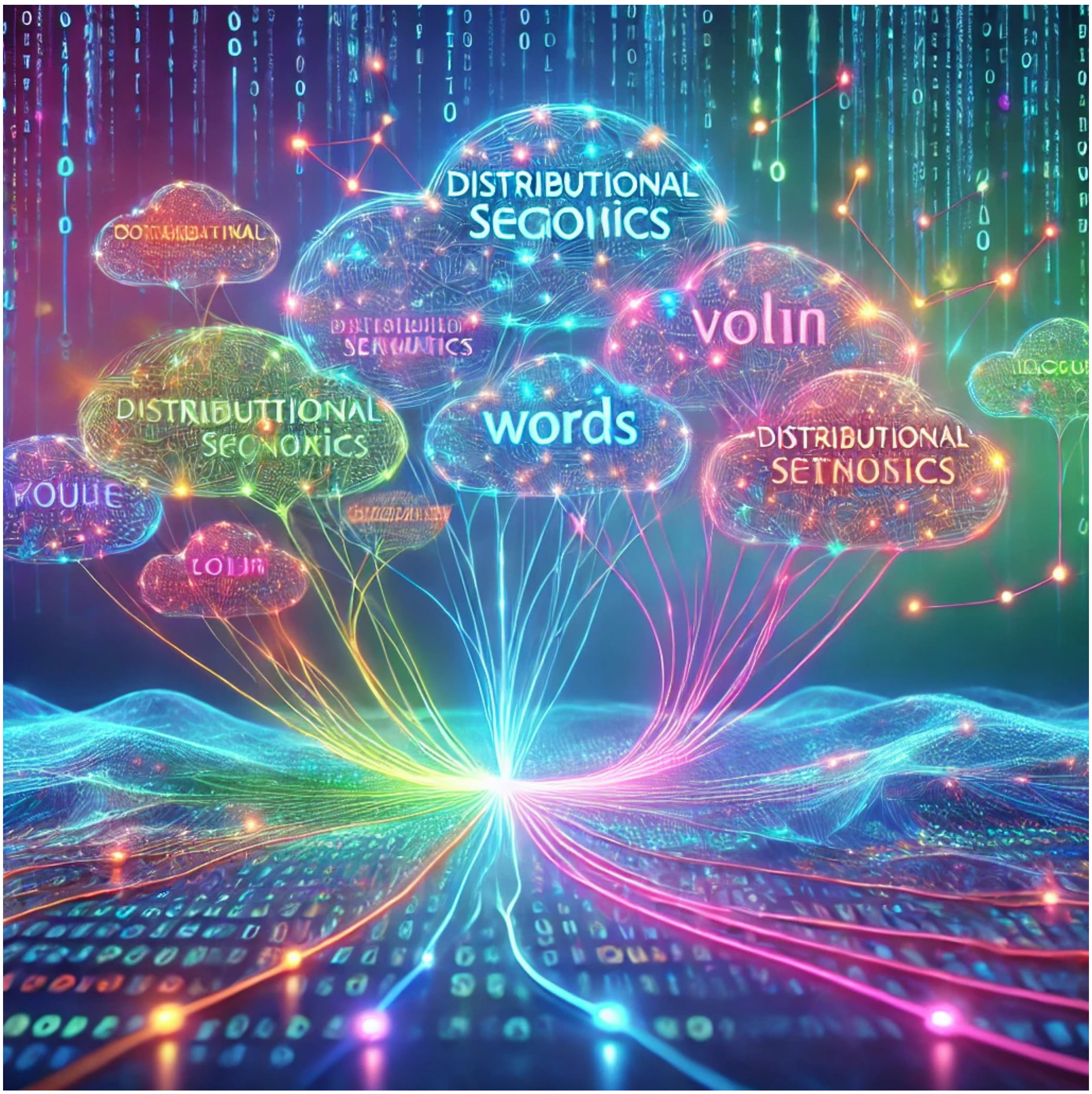
Introduction to Natural Language Processing, Assignment 2

Enrique Mesonero Ronco   Sergio Sánchez García   Ismael Cross Moreno

December 11, 2024



Contents

1 Distributional Semantics

1.1 Raw Co-occurrence Vectors

Given the following raw co-occurrence counts of words with contexts jealous (c_1) and gossip (c_2):

	c_1(jealous)	c_2(gossip)
w_1	2	5
w_2	3	0
w_3	4	0
w_4	0	4

1. **Compute the TF-IDF Weighted Co-occurrence Matrix**
   Use the following formulas:

$$\text{tf}(w, c) = \log\left(\frac{\text{freq}(w, c)}{\max_{w'} \text{freq}(w', c)} + 1\right)$$

$$\text{idf}(c) = \log\left(\frac{|V|}{|\{w \in V : \text{freq}(w, c) > 0\}|}\right)$$

   Where $|V| = 4$.

   TF:

   - $\text{tf}(w_1, c_1) = \log\left(\frac{2}{4} + 1\right) = 0.176$
   - $\text{tf}(w_2, c_1) = \log\left(\frac{3}{4} + 1\right) = 0.243$
   - $\text{tf}(w_3, c_1) = \log\left(\frac{4}{4} + 1\right) = 0.301$
   - $\text{tf}(w_4, c_1) = \log\left(\frac{0}{4} + 1\right) = 0$
   - $\text{tf}(w_1, c_2) = \log\left(\frac{5}{5} + 1\right) = 0.301$
   - $\text{tf}(w_2 c_2) = \log\left(\frac{0}{5} + 1\right) = 0$
   - $\text{tf}(w_3, c_2) = \log\left(\frac{0}{5} + 1\right) = 0$
   - $\text{tf}(w_4, c_2) = \log\left(\frac{4}{5} + 1\right) = 0.255$

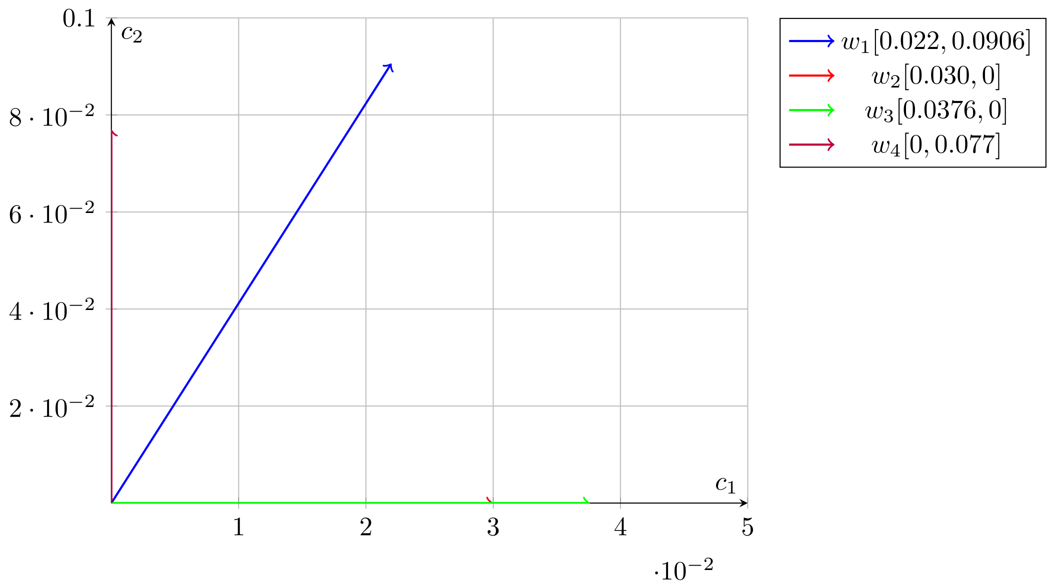
   IDF:

   - $\text{idf}(c_1) = \log(\frac{4}{3}) = 0.125$
   - $\text{idf}(c_2) = \log(\frac{4}{2}) = 0.301$

   TF-IDF

   - $\text{tf-idf}(w_1, c_1) = 0.176 \cdot 0.125 = 0.022$
   - $\text{tf-idf}(w_2, c_1) = 0.243 \cdot 0.125 = 0.030$
   - $\text{tf-idf}(w_3, c_1) = 0.301 \cdot 0.125 = 0.0376$
   - $\text{tf-idf}(w_4, c_1) = 0 \cdot 0.125 = 0$
   - $\text{tf-idf}(w_1, c_2) = 0.301 \cdot 0.301 = 0.0906$
   - $\text{tf-idf}(w_2, c_2) = 0 \cdot 0.301 = 0$
   - $\text{tf-idf}(w_3, c_2) = 0 \cdot 0.301 = 0$
   - $\text{tf-idf}(w_4, c_2) = 0.255 \cdot 0.301 = 0.077$

2. **Represent Each Word as a TF-IDF Vector**



3. **Compute the Euclidean Distance Between:**

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

   (a) w_1 and $w_2 \rightarrow d(w_1, w_2) = \sqrt{(0.022 - 0.030)^2 + (0.0906 - 0)^2} = 0.091$

   (b) w_2 and $w_3 \rightarrow d(w_2, w_3) = \sqrt{(0.0300 - 0.0376)^2 + (0 - 0)^2} = 0.0076$

4. **Discussion**
   Based on the Euclidean distances computed, evaluate whether Euclidean distance is an appropriate measure for capturing the relationships between the words.
   Answer: Is not a valid option because the Euclidean distance is affected by vectors length, a long vector would have larger magnitudes so the Euclidean distance would increase even if the direction of the vector is the same.

1.2 Prediction Based Word Vectors

- Why does Word2Vec use separate input vectors (u_w) and output vectors (v_w) for each word, and how does this benefit the model's performance?
  Answer: Word2Vec uses two random vectors (of dimension $d << |V|$) assigned to each word:

  - $u_w \rightarrow$ the "input" vector of the word w, when it is used to predict another word.
  - $v_w \rightarrow$ the "output" vector of the word w, when it is the one being predicted.

  It uses two vectors for each word to minimize the probablity because with just one vector used for context and center word the minimization is not possible, the vector associated to the center word is different that the vector associated to the context of the word.

- What are the primary differences between the Skip-Gram and Continuous Bag-of-Words (CBOW) models in Word2Vec, and in what scenarios might one outperform the other?
  Answer:

  - Skip-Gram predicts each context words from the center word. More efficient with less amounts of data.

- CBOW predicts the center word from the whole context. Faster, more efficient with big amounts of data.

- How does negative sampling improve the efficiency of training Word2Vec models compared to using the full softmax function?
  Answer: Denominator in softmax sums over words in V_N, instead of the whole $V \rightarrow N << |V|$, this improves the efficiency because sotftmax is applied on a shorter vector.

- How does the choice of window size in Word2Vec affect the type of semantic relationships the model captures?
  Answer:
  - Small windows $\rightarrow$ more sintactic relationships catched
  - Big windows $\rightarrow$ more semantic relationships catched

- What strategies canWord2Vec employ to handle out-of-vocabulary (OOV) words, and what are the implications of these strategies?
  Answer: The main strategy is tokenization, that enrich word embeddings with subword information, this way the vector of OOV words is composed of embeddings that belong to the vocabulary and new infromation about the words can be obtained from them.

2 Topic Modeling

Consider a simple corpus with the following characteristics:

- **Vocabulary (V)**: {apple, banana, cherry}
- **Number of Topics (K)**: 2
- **Number of Documents (M)**: 2

The initial topic distributions over words (ϕ_k) and document distributions over topics (θ_m) are randomly initialized as follows:

$$\phi_1 = \left[\tfrac{1}{3}\,\tfrac{1}{3}\,\tfrac{1}{3}\right], \quad \phi_2 = \left[\tfrac{1}{3}\,\tfrac{1}{3}\,\tfrac{1}{3}\right]$$

$$\theta_1 = \left[\tfrac{1}{2}\,\tfrac{1}{2}\right], \quad \theta_2 = \left[\tfrac{1}{2}\,\tfrac{1}{2}\right]$$

Documents

- **Document 1**: apple, banana
- **Document 2**: banana, cherry

1. For each word in each document, compute the probability of assigning it to each topic using the current ϕ and θ values. Specifically, calculate:

$$P(z_{mn} = k) \propto \phi_k[w] \times \theta_m[k]$$

   for each word w in document m.
   Answer:

   - "apple" in document 1 $\rightarrow P(Z_{11} = 1) \propto \frac{1}{3} \cdot \frac{1}{2} = \frac{1}{6}$
   - "banana" in document 1 $\rightarrow P(Z_{11} = 1) \propto \frac{1}{3} \cdot \frac{1}{2} = \frac{1}{6}$
   - "banana" in document 2 $\rightarrow P(Z_{11} = 1) \propto \frac{1}{3} \cdot \frac{1}{2} = \frac{1}{6}$
   - "cherry" in document 2 $\rightarrow P(Z_{11} = 1) \propto \frac{1}{3} \cdot \frac{1}{2} = \frac{1}{6}$

   As we expected, the probabilities for all words are equal, so no preference exists between topics.

2. Based on the probabilities computed earlier, assign a new topic to each word in each document. Assume you sample deterministically by choosing the topic with the higher probability.
   Answer: We assign the topics deterministically to each word in each document:

   - **Document 1:** Assign topic 1 to "apple" and topic 2 to "banana".
   - **Document 2:** Assign topic 1 to "banana" and topic 2 to "cherry".

3. Update the ϕ_k and θ_m distributions based on the new topic assignments. Compute the new probabilities:

$$P(w|k) = \frac{C(w,k)}{\sum_{w'} C(w',k)}$$

$$P(k|d) = \frac{C(k,d)}{\sum_{k'} C(k',d)}$$

   where $C(w,k)$ is the count of word w assigned to topic k across all documents, and $C(k,d)$ is the count of topic k in document d.

Answer:
Updated ϕ_k:

$$C(w,k): \begin{array}{l} C(\text{apple},1)=1, \quad C(\text{banana},1)=1, \quad C(\text{cherry},1)=0 \\ C(\text{apple},2)=0, \quad C(\text{banana},2)=1, \quad C(\text{cherry},2)=1 \end{array}$$

$$\phi_1 = \left[\tfrac{1}{2}\,\tfrac{1}{2}\,0\right], \quad \phi_2 = \left[0\,\tfrac{1}{2}\,\tfrac{1}{2}\right]$$

Updated θ_m:

$$C(k,d): \begin{array}{l} C(1,\text{Doc }1)=1, \quad C(2,\text{Doc }1)=1 \\ C(1,\text{Doc }2)=1, \quad C(2,\text{Doc }2)=1 \end{array}$$

$$\theta_1 = \left[\tfrac{1}{2}\,\tfrac{1}{2}\right], \quad \theta_2 = \left[\tfrac{1}{2}\,\tfrac{1}{2}\right]$$