Introduction to Natural Language Processing, Assignment 1

Enrique Mesonero Ronco Sergio Sánchez García Ismael Cross Moreno

November 28, 2024

Contents

1 Text Processing

1.1 Tokenization

- The first step is pre-tokenization, it consists on producing the set of word-leven tokens V_W with their occurrence frequencies
$$V_B = \{bat : 10, bar : 5, cat : 8, car : 4, cart : 6\}$$
- Secondly, it is necesary to initial base vocabulary of character tokens:
$$V_B = \{a, b, c, r, t\}$$
$$V_W(\text{as per} V_B) = \{b, a, t : 10; b, a, r : 5; c, a, t : 8; c, a, r : 4; c, a, r, t : 6\}$$
- Thirdly, start with the iteration until desired vocabulary size is reached (In this case, only one iteration is needed):
  - Count the frequency of each pair of consecutive tokens from V_B
  $$''b'' + ''a'' = 15, ''a'' + ''t'' = 18, ''a'' + ''r'' = 15, ''c'' + ''a'' = 18, ''a'' + ''r'' = 15, ''r'' + ''t'' = 6$$
  - Choose the highest frequency pair, even though "a" + "t" and "c" + "a" has the same frequency we are going to choose "c" + "a" (in the next iteration "a" + "t" would be chosen)
  $$''c'' + ''a''\text{occur the most, 18 times in total}$$
  - Merge the two tokens with the highest frequency (over all words from V_W)
  $$V_B = \{a, b, c, r, t, ca\}$$
  - Add the merged token to the vocabulary V_B
  $$V_W = b, a, t : 10; b, a, r : 5; ca, t : 8, ca, r : 4; ca, r, t : 6$$

1.2 Levenshtein Distance

- hund → handy

	_	H	U	N	D
_	0	1	2	3	4
H	1	0	1	2	3
A	2	1	1	2	3
N	3	2	2	1	2
D	4	3	3	2	1
Y	5	4	4	3	2

Total: 1 substitution operation + 1 insertion operation → Levensthein Distance = 2

- natty → gritty

	_	N	A	T	T	Y
_	0	1	2	3	4	5
G	1	1	2	3	4	5
R	2	2	2	3	4	5
I	3	3	3	3	4	5
T	4	4	4	3	3	4
T	5	5	5	4	3	4
Y	6	6	6	5	4	3

Total: 2 substitution operations + 1 insertion operation → Levensthein Distance = 3

2 Words and the Company They Keep

2.1 Pearson's Chi-sqaure Test

1. We start from the word co-occurrence table (aka contingency table)

	B = b_1	$B = b_2$	Total
$A = a_1$	9	1770	1779
$A = a_2$	75	219243	219318
Total	84	221013	221097

E_{ij}: Expected frequency of the element in the contingency table

$$E_{ij} = \frac{f(w_i) \cdot f(w_j)}{N}$$

	$B = b_1$	$B = b_2$
$A = a_1$	$\frac{84 \cdot 1779}{221097}$	$\frac{1779 \cdot 221013}{221097}$
$A = a_2$	$\frac{84 \cdot 219318}{221097}$	$\frac{221013 \cdot 219318}{221097}$

	$B = b_1$	$B = b_2$
$A = a_1$	0.676	1778.32
$A = a_2$	83.32	219234.6759

2. χ^2: statistic for a pair of words (w_i, w_j)

$$\chi^2 = \sum_{i,j} \frac{(O_i, j - E_i, j)^2}{E_i, j}$$

$$\chi^2 = \frac{(9 - 0.676)^2}{0.676} + \frac{(1770 - 1778.32)^2}{1778.32} + \frac{(75 - 83.32)^2}{83.32} + \frac{(219243 - 219234.6759)^2}{219234.6759} = 102.5+0.04+0.83+3.16 \cdot 10^{-4}$$

$$\chi^2 = 103.37$$

3. The chi-square value is 103.37, which is much larger than the critical value of 7.88 from the chi-square distribution (with 1 degree of freedom at the 0.005 significance level). The critical value of 7.88 is the threshold for deciding if these differences are too large to be explained by chance. In this case, the test value is 103.37, far exceeding the threshold, indicating the differences are highly unlikely to occur randomly. Therefore, we reject the null hypothesis of independence and conclude that A and B have a significant and meaningful relationship.

2.2 PMI: Pointwise Mutual Information

1. **When is PMI between two words negative and what does a negative PMI indicate?**

   PMI is negative when the observed probability of two words co-occurring is less than the expected probability if they were independent. A negative PMI indicates that the words co-occur less often than expected, suggesting a weak or even negative association between them.

2. **What are practical problems with PMI?**

   Some practical problems with PMI include:

   - **Bias towards low-frequency words:** PMI assigns high values to rare word pairs, even if their co-occurrence is not meaningful.

- **Sparsity issues:** PMI can be undefined if one of the words has zero occurrences in the data.
- **Misleading results in small datasets:** PMI depends on reliable probability estimates, which can be inaccurate for small corpora.

3. **What modifications will you make to PMI to address the problems? Give reasons for making these modifications.**

   - **Normalization:** Normalizing PMI values (e.g., by dividing by the log of the joint probability) reduces bias towards rare words.
   - **Smoothing:** Applying additive smoothing prevents undefined PMI values by avoiding zero probabilities.
   - **Shifted PMI (SPMI):** Adding a constant (e.g., subtracting the log of the dataset size) adjusts the scale and prevents overestimation of rare word pairs.

   These modifications ensure more robust results and better handling of frequency biases.

4. **How do the concepts of lexical associations evolve when moving from bigrams to n-grams where $n > 2$?**

   In bigrams, lexical associations capture relationships between two consecutive words. When moving to n-grams, the associations extend to sequences of n words, capturing more complex dependencies and contextual information. This is achieved with extension patterns that generalize lexical association measures to n-gram length.

5. **What are extension patterns in terms of lexical associations, and how do they enable the application of bigram-based measures?**

   Extension patterns consists on decomposing n-gram associations into a combination of smaller units, like bigrams or trigrams. Extension patterns enable application of bi-grand measures by generalizing to n-grams, some generalizations are more straightforward than others.

6. **How are the following two functions computing lexical association for n-grams differently?**

$$G_1 = \frac{LA(w_1, w_2...w_n) + LA(w1...w_{n-1}, w_n)}{2}$$

   - **G1:** This function treats the n-gram by breaking it into two sub-components for pairwise evaluation. It decomposes the n-gram into two n-1-gram lexical associations, it ends up dividing by 2 the sum of both decompositions to smooth variability and to get a balanced score.

$$G_2 = \frac{1}{n-1} \sum_{i=1}^{n-1} LA(w_i, w_{i+1})$$

   - **G2:** This method treats the n-gram breaking it into multiple sub-components for average association. It decomposes into multiple bigrams of the consecutive 'words', it ends up dividing by n-1 the sum of all the decompositions to normalize the score.