TLN Part 3 Lab 03 - Hanks

Paletto Andrea, Tuninetti André

September 12, 2023

Contents

1 Introduction

The goal of this exercise is to obtain semantic clusters using Hanks' valency theory on a transitive verb that occurs at least 200 times in a corpus. We decided to work with the verb "to play". In order to generate the corpus we used the wikipedia api and retrived the wikipedia pages of 10 famous guitarists. In this corpus our chosen verb occurs more than 400 times.

2 Methodology & Workflow

2.1 Project Overview

The ./res folder contains the .txt files necessary to build the corpus and to preprocess it. The "preprocessing.ipynb" file constitute the pipeline for corpus generation and preprocessing. In fact, after generating the corpus using the wikipedia api we filter it by keeping only the sentences where our chosen verb is present. We then remove the stopwords and punctuation and in order to get a better disambiguation we replace every name present in the "names.txt" file with the word person. We do the same thing with personal pronouns. This notebook produce the "sentences.txt" file wich contains all the preprocessed relevant sentences. The "hanks.ipynb" is where we implement the required task. After loading the data we used the Spacy "en_core_web_sm" model to extract the left and right dependencies of the verb "play" in each sentence using the get_left_and_right(sentences, verb) function. Then using the "get_supersenses(synsets)" function we retrieve the Wordnet supersenses of the left and right dependences. At this point all we need to do is create the connections between the dependencies, we achieve this by zipping the left and right supersenses into two "defaultdict(list)". Then we plot the frequencies using a seaborn barplot and we show the semantic clusters using pandas and Ipython's function display.

2.2 Preprocessing

In this section, we will provide brief descriptions of the functions used in the preprocessing.ipynb file:

- 'get_corpus' This function fetches Wikipedia text content for a list of topics and saves it to a file.
- 'load_topics' This function loads topics from a file and returns them as a list.
- 'load_data' This function loads sentences from a file and returns them as a list.
- 'filter_corpus' This function filters relevant sentences from the corpus based on a specific verb.
- 'clean_text' This function cleans and preprocesses text data by removing stopwords and punctuation.
- 'substitute' This function substitutes names in sentences with 'person' and cleans the sentences.
- 'load_txt' This function loads text data from a file and returns it as a list of rows.

2.3 Hanks

In this section, we provide brief descriptions of the functions used in the hanks.ipynb file:

- 'load_data' This function loads sentences from a specified file and returns them as a list.
- 'get_left_and_right' This function extracts left and right dependencies of a specified verb in a list of sentences.
- 'get_supersenses' This function extracts supersenses (lexical names) from a list of synsets.
- 'create_connections' This function creates connections between left and right dependencies based on provided lists of left and right dependencies.
- 'plot_freqs' This function plots the results of the connections between left and right dependencies using the Seaborn library.
- 'show_clusters' This function prepares and displays tables summarizing the connections between left and right dependencies.

3 Conclusion

The histograms provided below show the frequencies of subjects, referred to as "left elements", and objects, referred to as "right elements", within our scenario. In the 'hanks.ipynb' file, we also utilized the IPython library to present two tables containing semantic clusters. These tables offer a simplified representation of Hanks' concept of "semantic types." These semantic types reveal the syntactical connections between words, specifically focusing on subjects and objects, within sentences.

3.1 Frequency visualization

- We can see that the most common left element is the "noun.communication" supersense which occurs 162 times. The second one is the "adv.all" supersense wich occurs with a frequency of 32 and the third one is "noun.artifact" with a frequency of 15. All the other supersenses have similar frequencies, their number of occurrence fits in a range between 1 and 6.

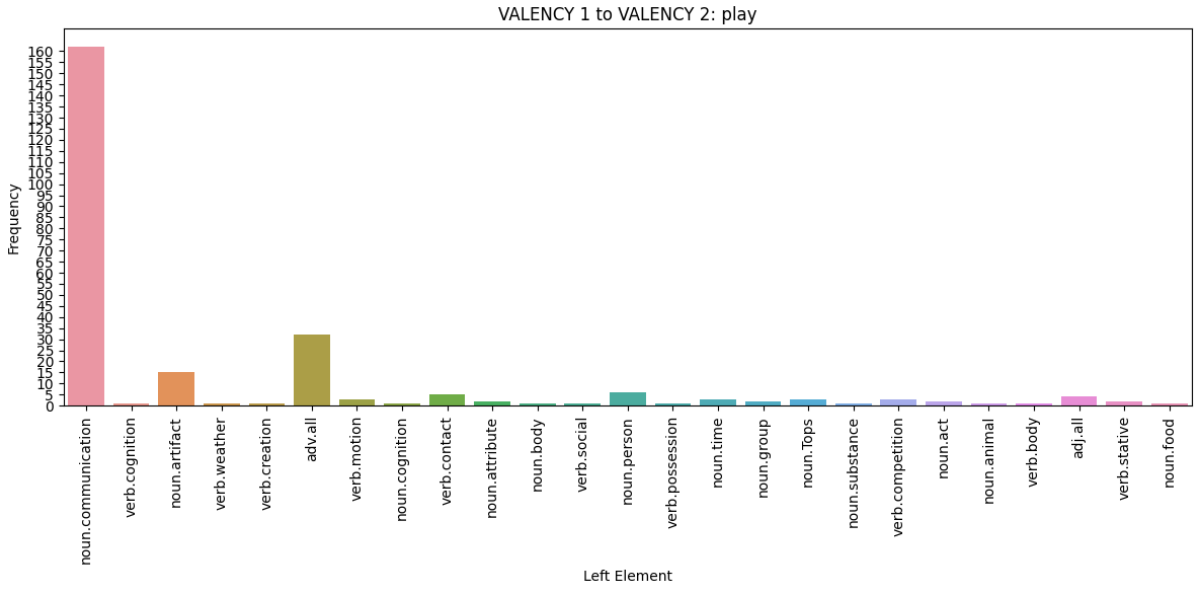

Figure 1: Histogram showing the frequencies of Left elements

- The Right Element histograms shows a different story. While the most frequent supersense is still "noun.communication" with a frequency of 101, this time it's followed by the "noun.artifact" supersense. Most of the other senses are more evenly distributed within a range that goes from 7 to 20.

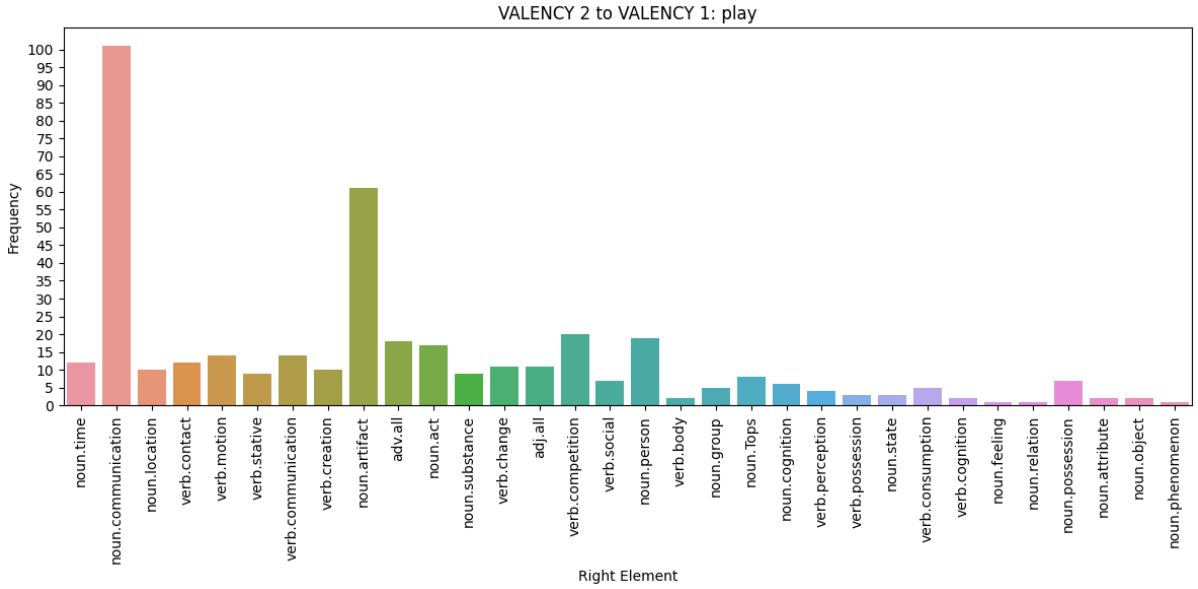

Figure 2: Histogram showing the frequencies of Right elements