# TextAnalyzer class

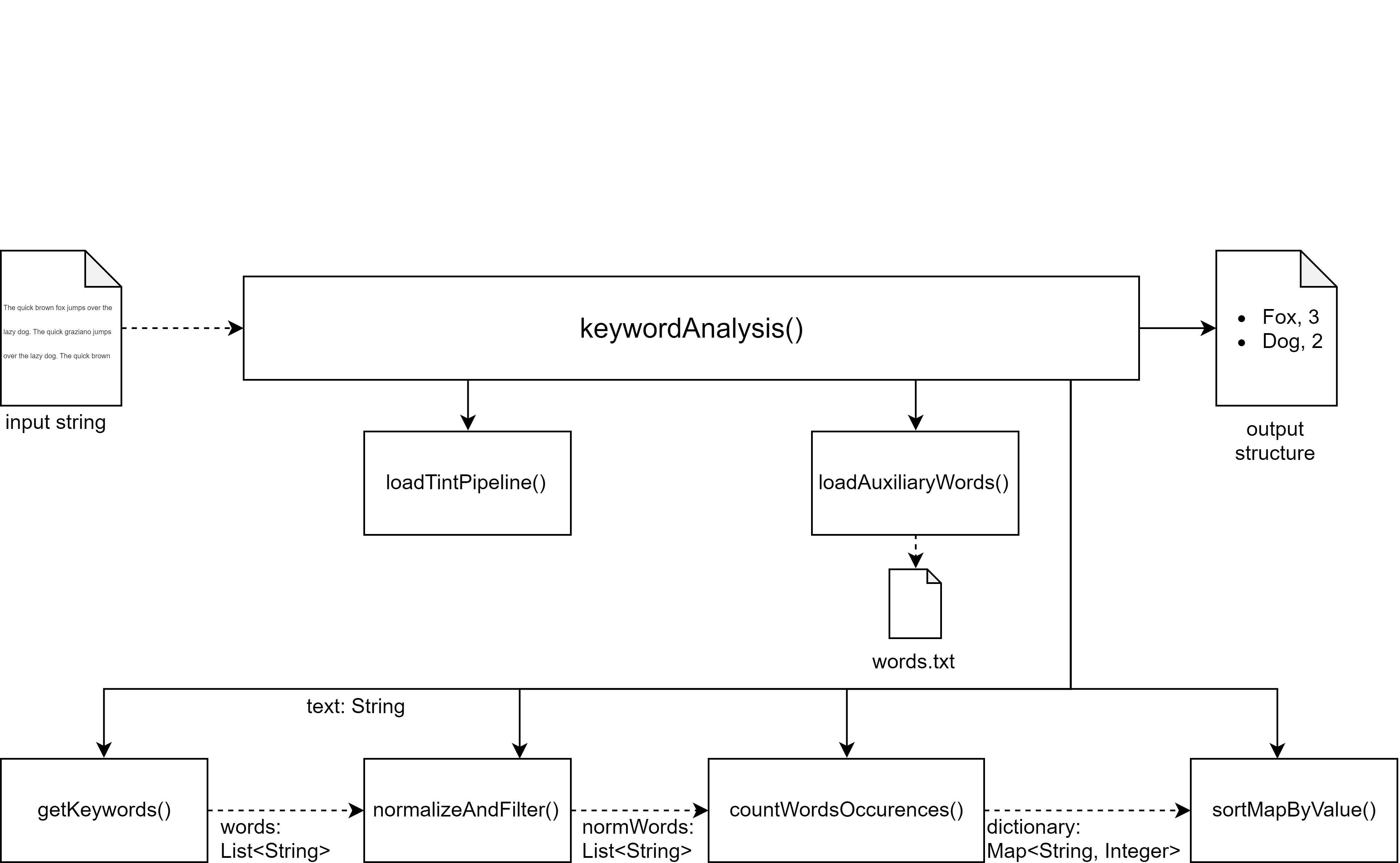
TextAnalyzer class contains the methods for analyzing a text and extracting its keywords and their number of occurrences within the text. Keywords are proper or common nouns, and adjectives: each verb, adverb and other parts of the speech are not considered, since they don’t define the content of an article.

This class contains only one public static method, that is keywordAnalysis(). It takes a string as a parameter, and it will call some other functions to return the list of all the keywords, together with their occurrences, sorted by number of occurrences.

TextAnalyzer exploits a NLP (Natural Language Processing) framework, called Tint (The Italian NLP Tool), based on Stanford CoreNLP. This framework provides useful methods for the recognition of the role of a word within a sentence, and it’s designed for the Italian language.

A scheme representing the function calls for keywordAnalysis() is the following:

## Scheme



## keywordAnalysis() method

This is the main, public static method of the class: it has the task to call the other methods in order to deliver the keyword analysis of the text that it’s passed to it as a string parameter. The output of this method is a map between string and integer: for each keyword found in the article, its number of occurrences is given.

## loadTintPipeline() method

This method creates a new TintPipeline and prepare it in order to prepare the Tint tools for the text analysis.

## loadAuxiliaryWords() method

loadAuxirialyWords() loads the content of “words.txt” file, that is the file containing all the words that must not be considered as keywords. This exclusion regards all the nouns with a non-specific meaning, or very trivial one. In fact, these words would pass the filter due to the Tint analysis, since they’re mere nouns.

## getKeywords() method

This method deletes all the special characters in the string passed as parameter (the text to analyze), because they sometimes create problems with the Tint pipeline. Then the text is fed to the Tint pipeline to get the so called ‘token annotation’: for each word, its grammar role is specified. The words that are not recognized as nouns or adjectives are deleted. Proper names of people or cities that are distributed in two or more words are glued together to get a unique word.

This method returns a list of string, each one representing a word (or a cluster of words in the case of proper names).

## normalizeAndFilter() method

This method takes the list of strings found after the getKeywords() call, and converts all the keywords to lowercase. It also removes the words that appear in the file “words.txt”. The remaining words are the keywords of the text to analyze.

## countWordsOccurrences() method

countWordsOccurrences() takes a list of normalized keywords, and returns a map between string and integer; this is the method that counts the number of occurrences for each word in the list.

## sortMapByValue() method

This method only sorts the map between string and integer returned by countWordsOccurrences() by the number of occurrences, in order to get a more usable version of the map.