Text Analysis with Spark RDD API

1 Introduction

This project is to implement simple data analytic workloads using the SparkRDD API. The data set that will work on is adapted from Contract Understanding Atticus Dataset (CUAD). You are asked to extract representative keywords from the anno- tated legal clauses using the RAKE algorithm described inAutomatic Keyword Extraction from Individual Documents[1]. **You are required to design and implement the extrac- tion algorithm using ONLY basic Spark RDD API operations and Spark SQL to load the original CSV files for convenience.**

2 Input Data Set Description

The complete CUAD data set contains 510 legal contracts as well as 13,000+ clauses extracted from the contracts. The clauses belong to 41 categories. The data set you will work with in this assignment contains only the extracted clauses in a few categories. They are selected from the label report section of the original data set. The label report section has a number of Excel files. Most files store clauses of a single category. Some files store clauses belonging to a number of related categories. All Excel files have a column storing the contract file name. The heading of that column is "Filename". The other columns are used to store clauses belonging to the category. The headings are the category names. The cell contains extracted clauses belonging to that category from a contract. Some contracts may not contain any clause belonging to a category. In that case, the corresponding cell may be empty or has a "nan" value. Other contracts may contain one or more clauses belonging to a category. All extracted clauses from one contract belonging to one category will be put in a single cell. The page number(s) is attached to each clause to indicate the location of the clause. If a clause appears on a single page, for instance, on page 5; it will be indicated as "(Page 5)". If a clause appears across pages, for instance across page 17 and 18; it will be indicated as "(Pages 17-18)".

You are asked to analyse three categories: "Governing law", "Change of Control" and "Anti-assignment". The governing law clauses are stored in a single Excel file: "Label

Report - Governing Law.xlsx". The clauses belonging to the other two categories are stored in another Excel file: "Label Report - Anti-assignment, CIC (Group 3).xlsx".

The "Label Report - Governing Law.xlsx" spreadsheet contains two columns: "Filename" and "Governing Law". It has 510 data rows. Any contract without a governing law clause would have a "nan" value. Below are three example rows in this file. The filenames and some clauses are shortened in the interests of readability.

Filename	Governing Law
Innerscope***.pdf	This Agreement shall be governed in all respects by the laws of the United States and the State of Florida, except for conflict of laws provisions. (Page 5)
Gridiron***.pdf	nan
GOOSE***.PDF	This Agreement will be interpreted and *** (Pages 57-58) The Franchise Agreement requires application of the laws of the State of Texas. (Page 84)

The first data row contains data of a contract with a single governing law clause appearing on Page 5. The second data row contains data of a contract with no governing law clause. The last data row contains data of a contract with two governing law clauses: one on pages 57-58 and the other on page 84.

The "Label Report - Anti-assignment, CIC (Group 3).xlsx" spreadsheet contains three columns: "Filename", "Change of Control" and "Anti-assignment". The file has 376 data rows. Contracts that do not contain a clause in either category are not included. Contracts with clauses in a single category will have an empty cell for the other category in the same row. Below are two example rows in this file. The filenames and clauses are shortened in the interests of readability.

Filename	Change of Control	Anti-assignment
Cybergy***.pdf		MA may not assign, *** (Page 12)
Gopage***.pdf	For purposes of*** (Page 15)	Licensee shall not assign *** (Page 15) Any purported assignment*** (Page 15)

The first data row contains data of a contract with no "change of control" clause and one "anti-assignment" clause. The second data row contains data of a contract with one "change of control" clause on page 15 and two "anti-assignment" clauses, both on page 15.

Two corresponding CSV files are created for easy manipulation by PySpark. They are: "Governing_Law.csv" and "Anti_assignment_CIC_g3.csv". The CSV files keeps the structure and the data of the corresponding xlsx files.

3 Rapid Automatic Keyword Extraction Algorithm

The keywords of a document may contain one word or multiple words. For instance, "text analysis" and "Spark" are reasonable keywords for this assignment instruction. The RAKE algorithm [1] generally favours longer keywords. It is designed to extract keywords from a single document. The basic algorithm has the following three steps:

- Step 1: Identify candidate phrases by removing stop words and punctuation. This is described in section 1.2.1 of the paper.
- Step 2: Compute and assign score for each candidate. This is described in section 1.2.2 of the paper.
- Step 3: Rank and retain top N candidates based on scores computed in step 2. This is described in section 1.2.4 of the paper. Section 1.2.3 describes an optional step to look for additional candidates. We will skip that step.

The paper also proposes an algorithm to identify important keywords for the entire corpus. This is described in section 1.5.2.

The following subsections contain brief descriptions of the algorithms relevant to this assignment.

3.1 Single document keyword extraction

RAKE [1] uses stop words[1] and punctuation[2] as delimiters to identify content words and candidate keywords. The content words are the words in a document after removing stop words and punctuation. Below is a sample text used in a few online RAKE tutorials [2]. All stop words are grey shaded and the punctuation is highlighted with yellow background.

feature extraction is not thatcomplex . there aremany algorithms available that canhelp you withfe ature extraction .rapid automatic keyword extraction isone of those .

All words without shading are content words. There are eleven such words: feature, extraction, complex, many, algorithms, available, help, rapid, automatic, keyword, one. A content word sequence that is not separated by a stop word or punctuation forms a candidate phrase. In the example text, there are six candidate phrases: feature extraction, complex, many algorithms available, help, rapid automatic keyword extraction and one. We can optionally set a maximum sequence length of keywords. For example, if we set maximum sequence length as three, the sequence rapid automatic keyword extraction would not count as a candidate.

[1] Stop words are a list of commonly used words in a language. They are normally ignored in text retrieval and natural language processing. Different systems may have slightly different lists of stop words. The examples used in this assignment description uses the NLTK English stop word list.

[2] The punctuation characters as defined in Python string can be accessed from string.punctuation

Each content word has a frequency value $freq(w)$, which is simply the number of times it occurs in the document. For example, the frequency of the word *EXTRACTION* is 3 in the sample text.

Each content word also has a degree value $deg(w)$. The degree value is the accumulated co-occurrence counts of this word within the candidate keywords. The co-occurrence count of a word with itself always equals its frequency. If a word appears in a two word candidate phrase, it co-occurs with the other word once. If a word appears in a three word candidate phrase, it co-occurs with each of the other two words once. In the above example, extraction co-occurs with feature twice in the candidate phrase feature extraction. It also co-occurs once each with rapid, automatic and keyword in the candidate phrase rapid automatic keyword extraction. The degree of extraction is 8, which is the sum of its frequency (3) and co-occurrences with other words (2,1,1,1).

The score of each content word is computed as the ratio of $deg(w)$ to $freq(w)$. The score of a candidate phrase is the sum of all its member word's scores. Below is an example of computing the score of the phrase "feature extraction":

- Score of feature: $deg(\text{'feature'})/freq(\text{'feature'}) = 4/2 = 2$. feature occurs twice in the sample. Its frequency is 2. It also co-occurs twice with extraction, both in keywords feature extraction. Its degree is $4 = 2+2$.
- Score of extraction: $deg(\text{'extraction'})/freq(\text{'extraction'}) = 8/3 = 2.66$
- Score of feature extraction: $4.66 = 2 + 2.66$

Once the score of all candidates are computed, the top N candidates are selected as keywords of the document.

3.2 Keywords of corpus

A corpus contains many documents. Documents in a corpus are usually in the same domain. In many cases, it is worthwhile to identify important keywords in the corpus. The paper [1] uses the *ESSENTIALITY* score to extract important keywords.

The following document frequencies are used to compute the essentiality:

- The referenced document frequency of a keyword, $rdf(k)$, is the number of documents in which the keyword is identified as a candidate phrase.
- The extracted document frequency of a keyword, $edf(k)$, is the number of documents in which the keyword was selected for extraction.

An *ESSENTIAL* keyword of the corpus should have a high chance of being selected as keywords in all documents it is identified as candidate. For keywords with similar chance of being selected, the one that is selected by more documents are more essential to the corpus. The essentiality of a keyword is computed as:

$$ess(k) = \frac{edf(k)}{rdf(k)} \times edf\,(k)$$

The first component measures the chance of being selected as keywords, the second component measures the number of documents selecting it as keywords.

4 Workload Description

You are asked to implement two methods to compute the top 20 key- words in a category.

The first method treats each clause in the data set as a document and each category as a corpus. In this case, the data set contains 3 corpora each with a few hundred short documents. We assume that the maximum length of a keyword is four words. The number of keywords to be extracted from each document is four. You are asked to print out the top 20 keywords of each corpus and their respective *rdf*, *edf* and *ess* scores.

The second method treats the entire category as a document consisting of all clauses identified from different contracts. In this case, the data set contains three long documents. We assume the maximum length of a keyword is four words. You are asked to print out the top 20 keywords of each document and their respective scores computed as the sum of words' degree to frequency ratios.