**NLP – Keyword Extraction Project Report**

Name: Hao-Lun Lo

1. **Abstract**

In this project, implementation of five approaches of keyword extraction are discussed. The purpose of this project is to implement several keyword extraction methods and compare the results. The output keywords are strongly affected by the applied algorithm, and the length of input texts (corpus). Besides, the output format can be a single word, a phrase, or a short sentence, depending on the method used.

1. **Introduction**

For most people, being capable of reading well has become an important skill. When we read a big amount of texts, what we do most of time is not reading word-by-word but skimming fast just to capture a few keywords, so that we can not only save time but also get the most important ideas from the reading. As the reason above, keyword extraction could be a strong technique to help us obtain useful information more rapidly. It can help us quickly judge a sentence, categorize an article. For higher-level application, we can even take advantage of those keywords as a feature for a supervised model to train.

Keyword extraction can perform different output format, such as a word, a phrase, or even a sentence for summarization, based on needs. All applied methods in this project generate few keywords or few phrases. The five methods are: **frequent noun extraction (method 1), keywords extraction based on collocations (method 2), frequent noun phrase extraction (method 3), rapid automatic keyword extraction (RAKE, method4) and TD-IDF method (method 5)**. I will show the basic ideas of these algorithms and how the methods deal with keyword extraction problem. I will also provide appropriate input examples for each method and explain the difference between the outputs produced by different approaches.

1. **Data**
   1. Input Data

Since there are many methodologies that do not apply any machine learning algorithm to do keyword extraction tasks, data training is not needed. In this project, the input data could be very simple. The expected input is a text file containing a corpus. The corpus should be at least a complete sentence for a better performance (which makes sense in keyword extraction tasks). The corpus could also be a long article, but to obtain better keyword extraction results, the article should be focused on a definite topic.

For each keyword extraction method, we should choose a corpus with an appropriate length. In my experiment, method 1, 2, 3 produce better keyword set with a larger corpus; method 4 is able to generate useful keywords with just a sentence or a long article; in contrast, method 5 does well if the input data is short and “special” (this will be discussed in approaches section.) To conclude, I randomly took a long article regarding to Google Project FI for method 1, 2, 3, and 4, a randomly took short sentence for testing method 4 and a specific input for method 5. (See **Appendix** for more details.)

|  |  |  |
| --- | --- | --- |
| Method Number | Method Name | Output Format (Each Keyword) |
| 1 | Frequent noun extraction | Long article |
| 2 | Keywords extraction based on collocations | Long article |
| 3 | Frequent noun phrases extraction | Long article |
| 4 | RAKE (Rapid Automatic Keyword Extraction) | Short sentence |
| 5 | TF-IDF method | Specific sentences |

* 1. Output Data

Each method outputs 5 most possible keywords, and the output data differs from applied method. The following table shows the output formats for each method:

|  |  |  |
| --- | --- | --- |
| Method Number | Method Name | Output Format (Each Keyword) |
| 1 | Frequent noun extraction | Single word |
| 2 | Keywords extraction based on collocations | Collocation pair |
| 3 | Frequent noun phrases extraction | Noun phrase |
| 4 | RAKE (Rapid Automatic Keyword Extraction) | Set of keywords |
| 5 | TF-IDF method | Single word |

1. **Approaches**
   * 1. Frequent noun extraction (Method 1)

This method is the simplest but efficient way to extract keywords from a corpus. We separate the method into 2 steps: Part-Of-Speech tagging and frequent noun extraction. In step 1, we may remove stop-words in advance. We then tag each word and extract nouns in the bag-of-words only. Next step, we simply count the frequency for each noun and retrieve the most frequent five nouns. These words are the final extracted keywords.

* + 1. Keyword extraction based on collocations (Method 2)

In the first method, the output is just a single word. We may be interested in multi-word expressions so that the extracted keywords can perform clearer and more meaningful interpretation. The second method is to search for collocations, which are words following one another, by looking for BIGRAMS in the corpus. Finally, we find the most frequent keywords coming with collocations.

* + 1. Frequent noun phrase extraction (Method 3)

Method 3 works similarly to method 2, extracting noun phrases as multi-word expressions rather than extracting a single word. We can easily reach this by defining a regular expression of a noun phrase pattern, for example:

“<DET>?<ADJ>\*<NOUN>+”

The above regular expression shows a noun phrase pattern. We first do POS-tagging for each word, and then check if the “tag sequence” contains the noun phrase pattern. Extracting all keyword sets matching the pattern and we get the noun phrases as keywords.

* + 1. Rapid automatic keyword extraction, a.k.a. RAKE (Method 4)

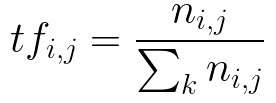
RAKE method calculates a score for each word which is part of a candidate keyword. The way to score a word is to scan each candidate keyword, count the frequency of that word, and see how many times it is co-occurring with other words (word degree). For each word, the algorithm calculates the score with the formula: (word degree) / (word frequency).

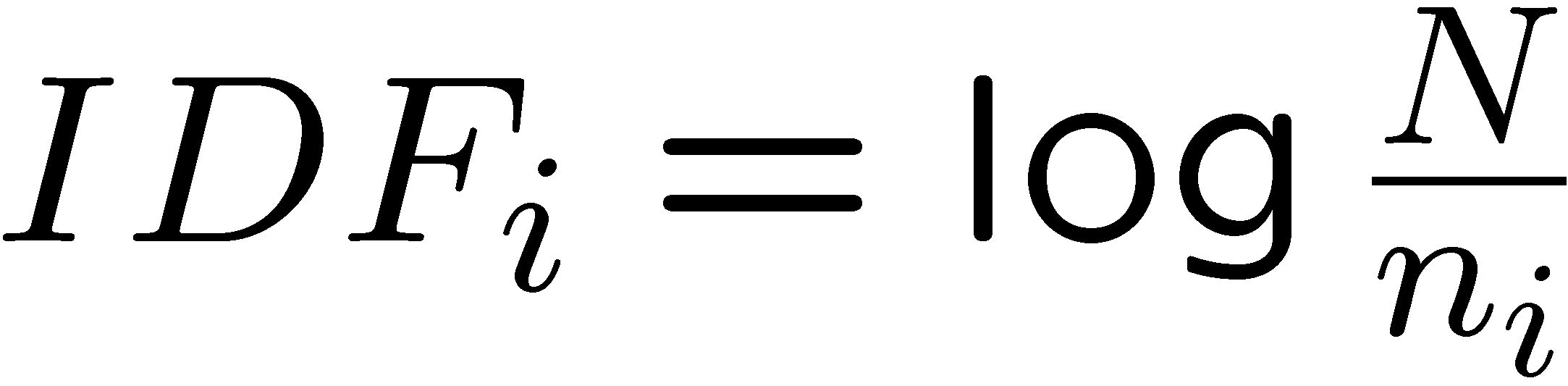
Finally, the algorithm calculates the RAKE score for a candidate keyword by summing the scores of all words in the candidate keyword. The algorithm then returns the candidate keywords and the scores.

* + 1. Term Frequency – Inverse Data Frequency, a.k.a. TD-IDF (Method 5)

The TF-IDF method provides a different idea with method 1. In method 1, we think intuitively that words appear most frequent have a higher probability to be keywords. However, stop-words appear frequently as well. TF-IDF calculates the weight (or the importance) of a word with a different way. It considers that words seldom appear do help in distinguishing between the data. Therefore, rare words may also get higher weight.

The method comes with two factors: term frequency (TF) & inverse data frequency (IDF). The two factors are calculated by the following equations:





TFij counts the frequency of a word in document in the corpus. Since we have only one document as an input, we can consider a sentence in the article as a single document. In the formula, “nij” denotes the frequency of a word i in a sentence j; the denominator is the total number of words in a sentence j. Therefore, the TF value increases as the frequency of a word i increases.

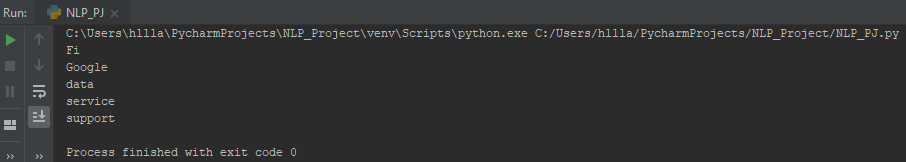
IDFi calculates the weight of rare words across all documents in the corpus. Here we also treat each sentence as a document since we have only one input article. In the formula, “N” is the number of sentences in the article; ni is the number of sentences that mention a word i. Hence the IDF value indicates that a word occurs rarely will have higher IDF score.

Eventually, we can get the TF-IDF score by multiplying TF score with IDF score. Simply extract the highest ones and we can get the keywords.

1. **Experiments & Results**
2. Frequent noun extraction (Method 1)

Input: Appendix a

Output: Fi, Google, data, service, support (Good result)

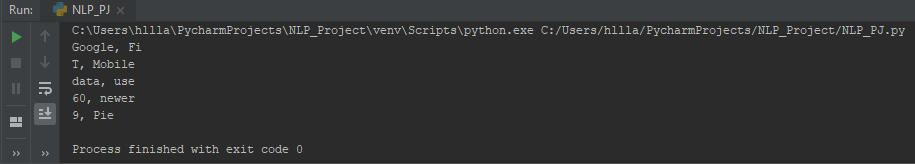


1. Keyword extraction based on collocations (Method 2)

Input: Appendix a

Output: (Google, Fi), (T, Mobile), (data, use), (60, newer), (9, Pie)

(Good result)

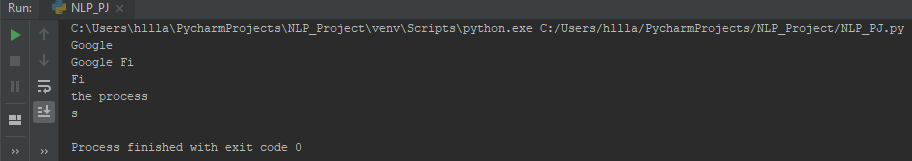


1. Frequent noun phrase extraction (Method 3)

Input: Appendix a

Output: Google, Google Fi, Fi, the process, s

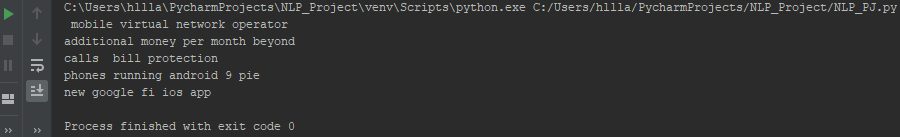
(Good result)



1. Rapid automatic keyword extraction, a.k.a. RAKE (Method 4)

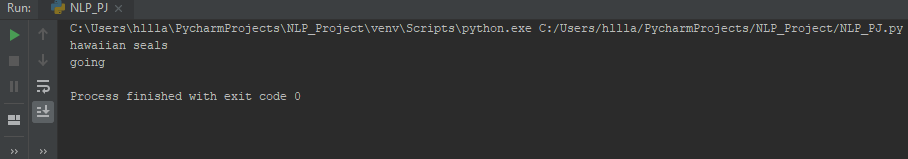
Input-1: Appendix a

Output-1: mobile, virtual network operator, additional money per month beyond, calls bill protection, phones running android 9 pie, new google fi ios app



Input-2: Appendix b

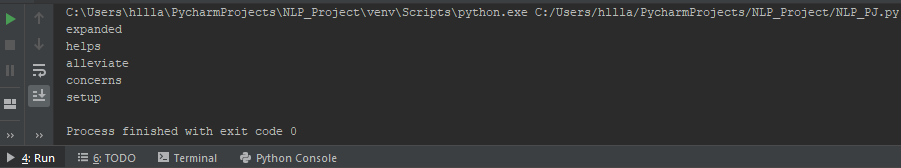
Output-2: Hawaiian seals, going (Good result)



1. Term Frequency – Inverse Data Frequency, a.k.a. TD-IDF (Method 5)

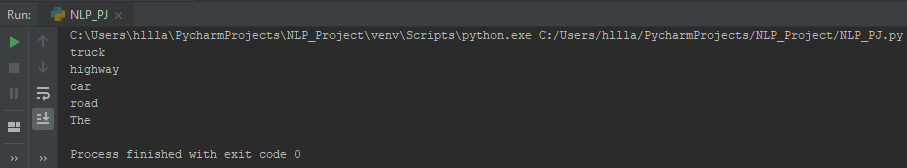
Input-1: Appendix a

Output-1: expanded, helps, alleviate, concerns, setup (Terrible result)



Input-2: Appendix c

Output-2: truck, highway, car, road, The (Good result)



1. **Discussion & Comment**

In method 1, 2, and 3, those output keywords are quite appropriate for summarizing this article. The article (Appendix a) is totally talking about Google Project Fi, we can see that most of the keywords cover the key points.

In method 4, although the output-1 keywords seem like related to the article, the weight order of the keywords are somewhat wrong because the output for RAKE algorithm are sorted in decreasing order. From my point, the fifth result (new google fi IOS app) should have a higher priority than the other four. But for output 2, the keywords make sense because the sentence is asking something about the Hawaiian seals. From the above two input example, I consider that RAKE may not be strongly affected by the size of the corpus.

In method 5, the output-1 is quite terrible in this case. None of the keywords are relative to the article. However, for the output-2 it performs perfectly. The reason for the terrible and good performance is the way to calculate the TF-IDF value, that is we have to consider rare words. The reason why the output-1 performs terrible may be due to the high frequency of important words like “Google” or “Project-Fi”, So that the TF-IDF score is pretty low.

As I mentioned in the Abstract, the output is strongly affected by the applied algorithm, and the length of input corpus. All the methods I implemented need to count the frequency of words, and four of them take advantage of the relations between words to calculate the most possible keywords. As a result, an input corpus with enough words is preferred for those methods.

There is no way to check the performance of keyword extraction because keyword identification is subjective. I think the next step to develop keyword technique is to make a rule to evaluate the “precision” of extracted keywords.

1. **References**

* Improved automatic keyword extraction given more linguistic knowledge <https://dl.acm.org/citation.cfm?id=1119383>
* Automatic Keyword extraction using RAKE in Python <https://www.thinkinfi.com/2018/09/keyword-extraction-using-rake-in-python.html>
* Automatic Keyword extraction using Topica in Python <https://www.thinkinfi.com/2018/09/automatic-keyword-extraction-using.html>
* How to process textual data using TF-IDF in Python <https://medium.freecodecamp.org/how-to-process-textual-data-using-tf-idf-in-python-cd2bbc0a94a3>
* An overview of keyword extraction techniques <https://www.r-bloggers.com/an-overview-of-keyword-extraction-techniques/>

1. **Appendix**
   1. Input article title: **Project Fi is now Google Fi, and it will work with iPhones and most Android devices (THE VERGE)**

Three years after it first launched, Google is making its cell service a little more official today. Project Fi is graduating into something a little more ambitious and getting a new name in the process: Google Fi. But the bigger news is that it’s also going to support more phones — a lot of phones — including the iPhone and “the majority of Android devices.”

This isn’t the first time that Fi has worked with Apple devices; you could get a data-only SIM for iPads as secondary devices before. And technically, a Fi SIM has always worked in an iPhone, provided you adjusted the data settings on the phone. But now Google is supporting iPhones directly for new customers, though it says that the support is in beta and requires “a few extra steps to get set up.” There will be a new Google Fi iOS app to help ease the process along.

More specifically, you’ll find that Visual Voicemail won’t work anymore, but iMessage will. iOS won’t pull the right MMS and data connections automatically, but it does work with just a few copy-and-pastes into settings.

For Android phones, setup should be a lot more straightforward. Though by going with Fi instead of a more traditional carrier, you should be aware that all customer support happens online or over the phone. In my experience, that hasn’t been a big issue, but sometimes it is nice to have a store to walk into to get in-person help.

Google Fi is an MVNO, which stands for “mobile virtual network operator.” That means that your actual service comes from larger carriers; Fi uses T-Mobile, Sprint, and US Cellular as its backbone. However, only a few phones (like the Pixels and others that Google sells) are able to dynamically switch between those carriers’ networks, and that doesn’t change today. Like before, phones that aren’t explicitly “designed for Fi” are stuck on T-Mobile’s network. Here’s how Google Fi will work with iPhones, for example.

Whatever network you’re technically on, Google allows people with phones running Android 9 Pie to route their data through its own VPN. However, Google Fi still has some catching up to do with other carriers when it comes to other features, including support for the RCS Universal Profile for texting and number sharing for things like LTE smartwatches.

But the real difference with Fi is the pricing model: it’s much simpler than what most other networks offer. It’s $20 for a phone line and $10 per gig of data you use — capped at $60 under a newer program it calls “Bill Protection.” The reason I like it is that you can get a data-only SIM, which costs no additional money per month beyond the data you use it on it. I think it’s one of the best deals in wireless. But depending on your data habits, the same might not be true for you.

Until now, I’ve had a shadow of a doubt about Google’s commitment to the Project Fi service. It’s not just that it was dubbed a “project,” but also that Google has, shall we say, a mixed record when it comes to providing communication services. Fi grew out of Google Voice in some ways, a service that has had sporadic stretches of stagnated support. And that’s to say nothing of Google Fiber, the broadband service that started with an aggressive rollout before faltering.

The expanded support helps to alleviate some of those concerns. And perhaps in another sign to show that it’s serious, Google is looking to juice sign-ups and sales with a deal that essentially pays for your phone, though it’s only going to be offered today, November 28th. It applies to both new subscribers and current customers:

For any phone you purchase [from Google Fi], you’ll receive the same value back in your choice of travel gift cards, which you can spend on flights with Delta and Southwest or lodging with Airbnb and hotels.com. Alternatively, if you’d rather set up Google Fi on your current phone, you’ll earn $200 of Fi service credit when you sign up today.”

* 1. Input short sentence for RAKE method:

**What's going on with Hawaiian seals?**

* 1. Input short sentence for TF-IDF method:

**The truck is driven on the highway. The car is driven on the road.**