Study and Implementation of Coreference Resolution For Text Classification Using NLTK

A Project Report
Presented to
The Faculty of the College of Engineering
International Technological University
CSC 520 - Python Programming

By
Sai Kaushik Mudela- 90144
Shruti Modi – 88594
Pooshan Vyas - 90727

PREFACE

Natural language processing (NLP) is a field of computer science, artificial intelligence, and computational linguistics concerned with the interactions between computers andhuman (natural) languages. NLP is related to the area of human–computer interaction. Many challenges in NLP involve natural language understanding, enabling computers to derive meaning from human or natural language input; and others involve natural language generation.

Our project is based on implementation of coreference resolution using NLTK (Natural Language Tool Kit) which will help in segregating the spam and ham emails.

ACKNOWLEDGEMENTS

We would like to take this opportunity to express our sincere gratitude to everyone who has helped us to make this project successful.

We are deeply thankful to our project instructor and advisor Prof. Ming Hwa Wang for his continued support and encouragement, which gave us the opportunity to build this project. Prof. Wang's guidance, monitoring and constant encouragement throughout the trimester helped us greatly to accomplish the tasks in a timely manner.

Our acknowledgements will never be complete without expressing our vote of thanks to the authors of the IEEE and ACM papers using which we could get a broader view of the topic. It helped us to get the background knowledge needed for this project. At last, our sincere regards to all our colleagues who have directly or indirectly helped us in this project.

ABSTRACT

The model followed in the project “Study and Implementation of coreference Resolution using NLTK” is select and classify model which requires usage of machine learning based on the existing corpora and the training data that we provide to our categorizer. The main aim of our project is to bifurcate that training data into spam and ham. This is used widely by companies such as yahoo, google, etc. for the email bifurcation.

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Table of Contents

List of Figures

Chapter 1. Project Overview

Introduction

Our objective is to implement Coreference resolution using machine learning by designing a data categorizer that can learn how to segregate the given set of emails based on the training data that we provide it. Once it has the knowledge on the categorization, we can pass any email to determine if that is a spam or ham.

Coreference resolution is one of the most important tasks on natural language programming(Bird, Klein, & and Loper, 2016). The challenge is to increase the accuracy of the results and reduce the latency to process the raw data. Once we teach the machine to parse the data, it's a matter of running the right test data for recognizing the type of emails. Then the machine can take in any email, parse it and categorize it based on what it learned from the test data. There are many other applications for such a categorizer and approach.

We plan to write the machine code in Python using the Natural Language Tool Kit (NLTK). We plan to use the tokenize Tree and Parented Tree modules in order to solve this problem. Also we would need the naïve Bayes classifier as well for building the categorizer.

Most of the approaches are either based on statistical data sets that act like a dictionary for translating a set of targets to a particular group of antecedents. This is a very efficient way of solving the problem with a condition that the rules/dictionary stay relevant with any given time. But that is not the case as the translation sets might go obsolete after a given point of time. Also a syntactic approach is also used that translates based on various conditions like the location of the antecedent with respect to the target

or its relevance with the target, etc. This approach is very intuitive as it deals with each case individually without any previous library to refer to. But when it comes to parsing a huge data set, it could be less efficient and slow.

We would like to propose an approach that takes the good points from both the worlds and produces a result that stays relevant with any given time and also is very efficient. We are going to implement co-reference resolution and apply it to an email set by designing a machine that takes in a particular training data set and learns how to parse the data. Once the learning process is done, it can use that knowledge to parse the input test data. So this makes the approach as good as the training data set and we can pass a new training data for every given interval to keep it relevant. As the categorizer adds the successfully parsed data to it corpora, its efficiency and accuracy increases with every iteration.

Implementing a more accurate and efficient machine for Coreference resolution by using the parsed data for segregating any given email based on the training data set. The main concern that we want to address with this kind of approach is to increase the accuracy and reduce the latency for every time we run the machine. Our scope is to consider the whole context of the given data to analyze it and then categorize it based on the knowledge it has got from the training data.

Chapter 2. Theoretical Basis and Literature Review

Coreference resolution is the task of finding all expressions that refer to the same entity in a text. It is an important step for a lot of higher level NLP tasks that involves natural language understanding such as document summarization, question answering, and information extraction(Coreference resolution, stanford university.)

This study sought to establish whether or not coreference resolution could have a positive impact on NLP applications, in particular on text summarization, recognizing textual entailment, and text categorization(Mitkov, Evans, Orasan, Dornescu, & and Rios, 2012).

The data we are working on is very specific, so we chose to use a rule based approach to coreference resolution. This means that we try to learn the characteristics of each coreference link ourselves, and program a method for the link manually. We examine concepts in a file, and if they meet our criteria, we create a method to link them. The idea is to create specific rules, yet generalized enough to apply to similar mentions in all documents(Morton, 2000).

We need to use two data sets: Training Data Set and Test Data set. We use the training data set to build the conditions for translations. Apply these conditions to the test data to optimize the Translations. This can be extended to implement the main concept of co-reference analysis where we can translate to a different language(Nagard & Koehn, 2010).

Leass and Schwall came up with a huge set of rules that mostly acts as a dictionary for translation of most of the pronouns. In the works of Winograd, it is stated

that if a pronoun is used multiple times in adjacent sentences, all of its instances need to map to the same entity

Klapholz and Lockman stated that antecedents of a pronoun can be found in one of the sentences, where n is as small as possible.

The author suggests the use of Selection-Then-Classification Model which helps in selecting most likely candidate antecedent as target-noun phrase using Antecedent Identification Model. The target-noun phrase should be classified as Anaphoric or Non-Anaphoric. Support Vector machines (SVMs) are used for antecedent identification and anaphoricity classification(Iida, Inui, & Matsumoto, 2005).

Solution to the Problem

As mentioned above the problem of coreference has been approached in a more static way by using a predefined rule base for translation or a syntactic way in which various parameters such as the location, gender, number of the prospective antecedent are used to calculate the right reference. These approaches are better in some ways, but do not have the capacity to be accurate and efficient at the same time. We intend to integrate both the approaches and design a more dynamic algorithm to solve the problem.

Using this approach we can provide an algorithm that solves the problem based on the training data. This means that the results are going to be as good as the data we provide it to learn from. Due to this we can increase the accuracy of the machine by providing it a better training data from time to time.

Chapter 3. Multiple Hypothesis

Multiple hypothesis is the way to minimize the number of errors. Our project uses 2 types of hypothesis Positive and Negative.

Positive Hypothesis

Our goal is to developed algorithm for text classification in Natural Language Processing. We are focusing on spam & ham classification in our sample data set. We processed our random email data which include lots of spam and ham, we analyzed data, apply our algorithm to identify more connection and link, compare with requirement and classified it to make more sense out of it (spam-ham). This will result in clear bifurcation of email in spam and ham. As a future scope, we can use the same logic and algorithm for twitter sentiment analysis and text summarization as well. We used Natural Language Tool Kit with Python in our coding.

Negative Hypothesis

In case of negative hypothesis, the result will vary depending upon the sample test data. Our requirement data set also needs to match to get the same results. However, if we change data set we can get results as per our requirement.

Chapter 4. Methodology

In recent years the use of email and data have become important in daily life as well as in sensitive business communication. We have many spam filters and they work fine. However, the current challenges are making spam and ham direction more smarter and efficient. There are also many other applications that can take advantage of this project, including social network analysis, recommendation systems, movie reviews, sentiment analysis/opinion mining, and others.

Algorithm Design

When we choose to approach spam filtering from a machine learning perspective, we view the problem as a classification problem. That is, we aim to classify an email as spam or not spam (ham) depending on its feature values. An example data point might be (x,y) where x is a d- dimensional vector < x1, x2, x3 … xn > describing the feature values and y has a value of 1 or 0, which denotes spam or not spam.

Machine learning systems can be trained to classify emails. As shown in Figure 1, a machine learning system operates in two modes: training and testing.

Training

During training, the machine learning system is given labeled data from a training data set. In our project, the labeled training data are a large set of emails that are labeled spam or ham. During the training process, the classifier (the part of the machine learning system that actually predicts labels of future emails) learns from the training data by determining the connections between the features of an email and its label.

Testing (Classification)

During testing, the machine learning system is given unlabeled data. In our case, these data are emails without the spam/ham label. Depending on the features of an email, the classifier predicts whether the email is spam or ham. This classification is compared to the true value of spam/ham to measure performance.

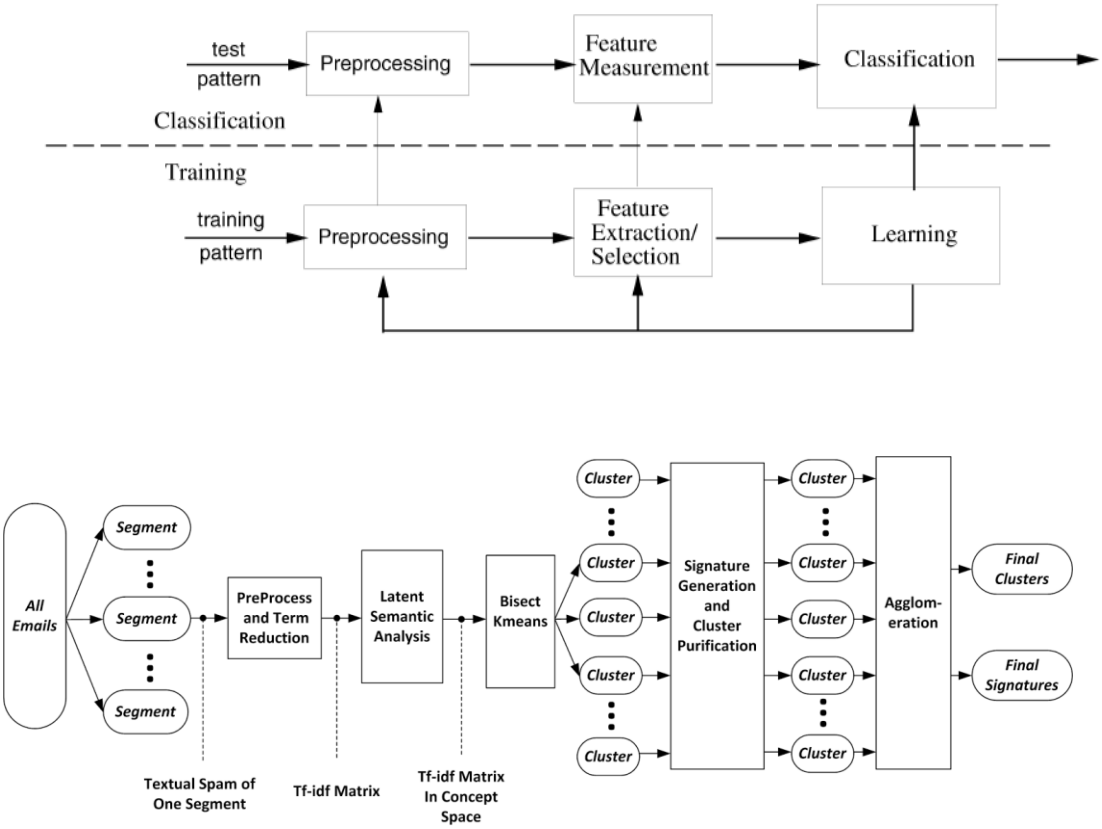

Figure 1: Model of Machine Learning Process

Methods

As shown in Figure 1, the above are the different steps in the machine learning process. First, in testing and training, the data must be preprocessed so that they able to

be used by the classifier. Then, the classifier operates in training or testing mode to either learn from the preprocessed data or classify future data.

Pre-processing

Our classifiers require a feature matrix that consists of the count of each word in each email. As will we discuss, this feature matrix can grow very large, very quickly. Thus, preprocessing the data involves two main steps: creating the feature matrix and reducing the dimensionality of the feature matrix.

Creating the Feature Matrix

There are a couple steps involved in creating the feature matrix, as shown in Figure 3. After preliminary preprocessing (removing HTML tags and headers from the email in the dataset), we take the following steps:

- Tokenize - We create "tokens" from each word in the email by removing punctuation.
- Remove meaningless words – Meaningless words, known as stop-words, do not provide meaningful information to the classifier, but they increase dimensionality of feature matrix. In Figure 3, the red boxes outline the stop-words, which should be removed. In addition to many stop-words, we removed words over 12 characters and words less than three characters.
- Stem - Similar words are converted to their “stem” in order to form a better feature matrix. This allows words with similar meanings to be treated the same. For example, history, histories, historic will be considered same word in the feature matrix. Each stem is placed into our "bag of words", which is a

list of every stem used in the dataset. In Figure 3, the tokens in blue circle are converted to their stems.

- Create feature matrix - After creating the "bag of words" from all of the stems, we create a feature matrix. The feature matrix is created such that the entry in row i and column j is the number of times that token j occurs in email i.

  These steps are completed using a Python NLTK (Natural Language Toolkit).

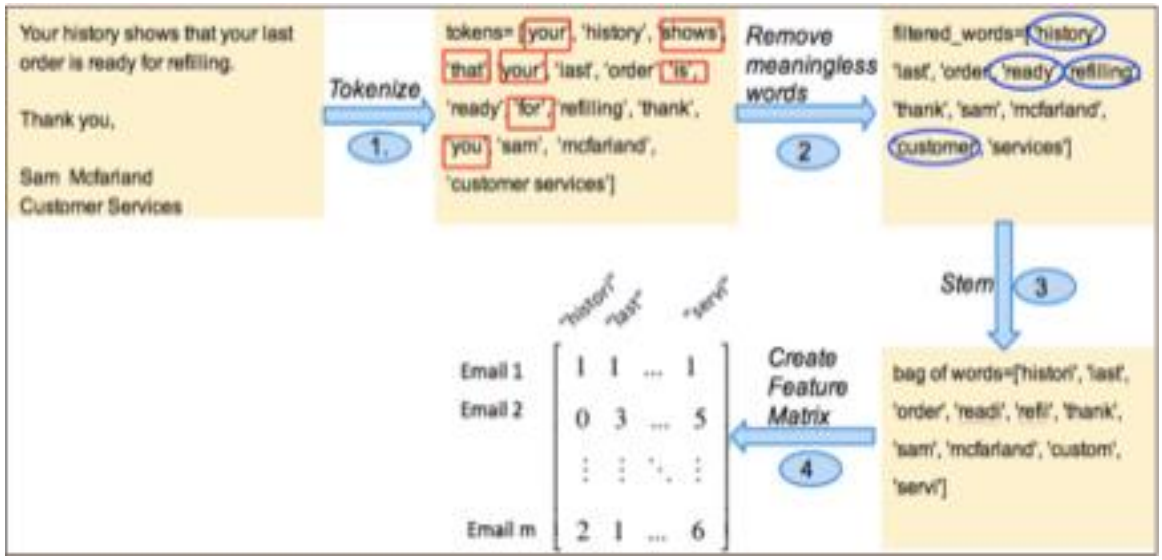

Figure 2: A Summary of the Pre-processing involved in creating Feature Matrix

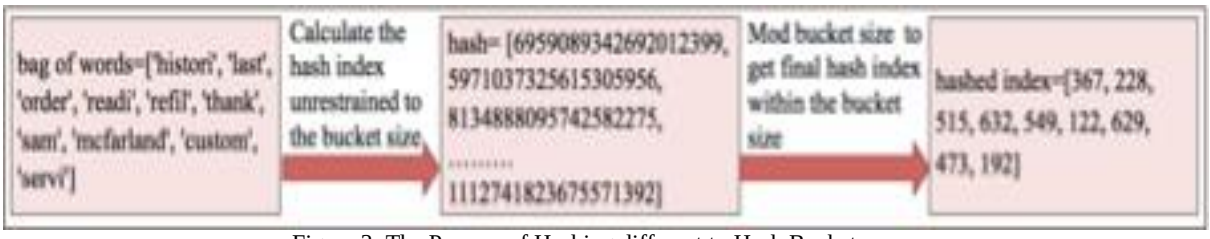

Figure 3: The Process of Hashing different to Hash Buckets

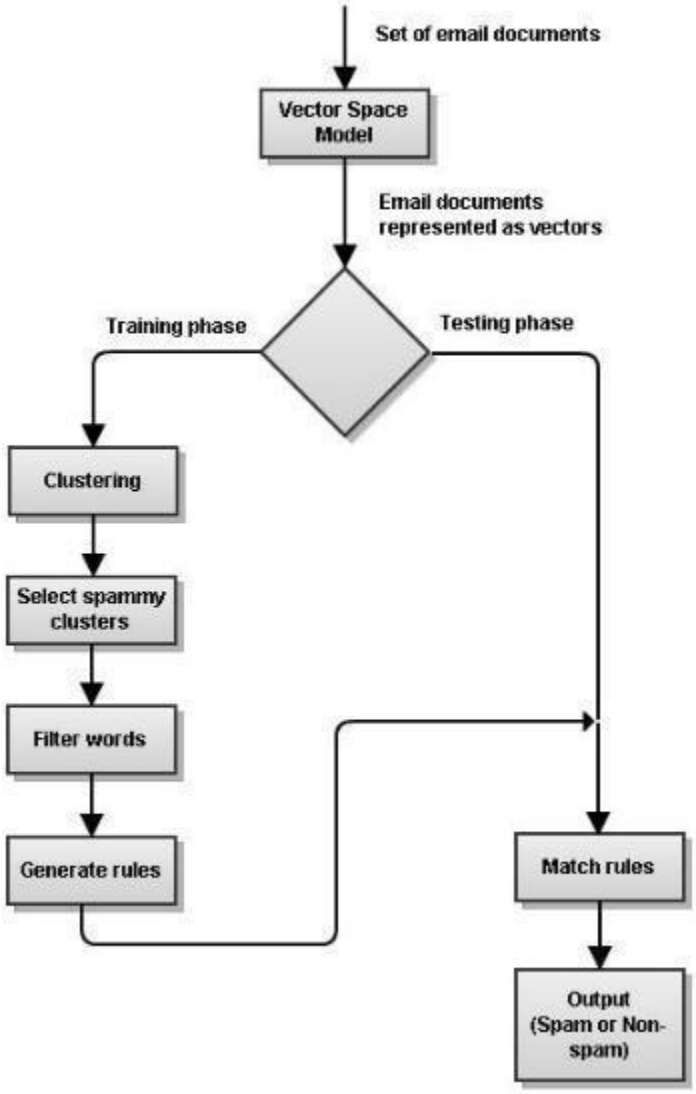

Figure 4: Algorithm Design Flow

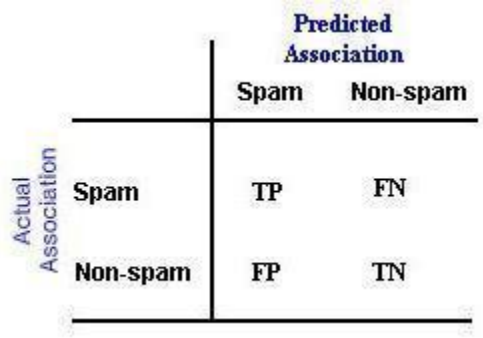

Figure 5: Table of all possible Conditions

In this project, we attempt to predict a text classification for spam and ham. A concepts we used are Tokenizing which means the process of breaking text into defined segments (usually using regexes or simple delimiters) and Stemming which means the process of breaking words to their stem removing plural forms, tense, etc. Example: Jump: jump-ing, jump-ed, jump-s

Collocations: Short sequences of words that commonly appear together. This is commonly used to provide search suggestions as users type.

N-Grams: Tokens consisting of one or more words: Unigrams, Bigrams, Trigrams

First, We have sample data set which has lots of emails, including both spam and ham. We parse this data in our code to check if it's spam or ham. We used text processing and to identify certain word which we think is more inclined towards spam. For that we are putting it on coreference chain so we get a set of references which refer to particular spam or ham.

Second, because the data we are working on is so specific, we chose to use a rule based approach to coreference resolution. This means that we try to learn the characteristics of each coreference link ourselves, and program a method for the link

manually. We examine concepts in a file, and if they meet our criteria, we create a method to link them. The idea is to create specific rules, yet generalized enough to apply to similar mentions in all documents/emails.

The methodology used will be python programming language, to solve the problem. There are a variety of tools available in Python that makes the task simple compared to other programming languages. The main tool that we intend to use is the Natural Language Toolkit (NLTK). NLTK offers many libraries that are simple and easy for the user to solve the task. The major libraries that will be helpful are tokenize, naive Bayes classifier, Tree and ParentTree.

Currently we plan to provide the script with an uncategorized and an untagged list of regular spam and ham emails. Based on the knowledge that the machine would have learnt using the training data, it should be able to categorize them, and present 2 lists as outputs.

In order to validate the legitimacy of the working of our script we would like propose 2 tests. One being the positive approach by which we pass through a known list of emails and compare the result with a list of the same email set that is categorized manually. Based on the delta these two lists, we can determine the error margin of the script. Another way of validation would be the negative way where in which we modify the training data set that we provide the machine and check if the that change reflects in the way it categorizes the emails. As said earlier, the machine is only as good as its training data. So, for any change in the training data, the results should be different.

Chapter 5. Bibliography

References

Bird, S., Klein, E., & and Loper, E. (2016). Analyzing sentence structure. Natural language processing with python - analyzing text with the natural language toolkit ()

Coreference resolution, stanford university. Retrieved from http://nlp.stanford.edu/projects/coref.shtml

Iida, R., Inui, K., & Matsumoto, Y. (2005). Anaphora resolution by antecedent identification followed by anaphoricity determination. ACM Transactions on Asian Language Information Processing, 4(4), 417-434. doi:10.1145/1113308.1113312

Mitkov, R., Evans, R., Orasan, C., Dornescu, L., & and Rios, M. (2012). Coreference resolution: To what extent does it help NLP applications?102 TSD 2012 draft(version 25th June 201), 1-12.

Morton, T. (2000). , 173-180. doi:10.3115/1075218.1075241

Nagard, R., & Koehn, P. (2010). Aiding pronoun translation with coreference resolution.(15-16 July 2010), 252-261.

http://ese.wustl.edu/ContentFiles/Research/UndergraduateResearch/CompletedProjects/WebPages/sp14/SongSteimle/Senior%20Design%20Final%20Report.pdf