Sentimental Analysis [Anaphora Resolution] using Hobb's Algorithm

Manjushree M¹ & Haripriya V²
¹Student, Jain University, Bangalore India
²Master of Science in Information Technology Department, Jain University, Bangalore India

Abstract: This document analyses and compares some of the approaches to the Anaphora Resolution task and so does it describe the algorithms used for the purpose. The developed system proposes to resolve the pronominal anaphors, namely third person, personal and possessive pronouns, relative and demonstrative pronouns. During the system development an annotation was made use of, allowing enriching text with anaphoric information on a quick way. The particular paper presents the application of Hobb’s algorithm for pronominal resolution. Hobb’s algorithm makes use of syntactic information rather than that of the semantic information and could thus be used as the baseline algorithm. The algorithm has nearly applied to almost all of the languages taking into account the roles of subject, object and its impact on anaphora resolution for reflexive and possessive pronouns. In case, if subject is dative, then possessive and reflexive pronouns loose the complimentary distribution, where reflexive pronoun binds with preceding nominative as well as dative noun phrase within a sentence, possessive pronoun extends the binding to the previous sentence as well. Anaphora Resolution has proven to be a very difficult problem; it requires the integrated information and application of syntactic, semantic and pragmatic knowledge. This paper further examines the hypothesis that instead of attempting to construct a monolithic method for resolving anaphora, the combination of multiple strategies, each exploiting a different knowledge source proves more effective both theoretically and computationally. Cognitive plausibility is established in that human judgment of the optimal anaphoric referent accord with those of the strategy-based methods. And human inability to determine a unique referent corresponds to the cases where different strategies offer conflicting candidates for the anaphoric referent.

1. INTRODUCTION
Recent years have witnessed the explosive development of online social media. Micro-blogs, which fuse information release and social networks, become one of the most popular information sharing platforms with high interactivity. However, researches mainly focus on sentiment classification of microblogs, which only demonstrates the sentiment orientation of one tweet or the sentiment fluctuation of tweets in a time interval. To be successful one needs those data to be structured so that the necessary information becomes available. When extracted, the information needs to be aggregated and presented to the interested parties in an understandable form.

Refer to the image 1.1:
1.1 Brief Introduction on Text Mining
Text mining or sentiment analysis is analysis of data contained in a natural language text, which deals with the computation of opinion, sentiment and subjectivity in text. Sentiment analysis refers to the use of natural language processing, text analysis and computational linguistics to identify and extract subjective information from the text documents. Basic task of sentiment analysis is to determine the polarity of a given texts. Tasks of dictionary making and sentiment analysis process are done by the means of KNIME, which is a user-friendly graphical workbench capable of entire analysis process. KNIME uses six different steps to process texts: reading and parsing documents, named entity recognition, filtering and manipulation, word counting and keyword extraction, transformation and visualization. Following workflows and tasks are developed and
executed using KNIME: Retrieving data from database Dictionary development and implementation Review scoring. Finally, building an opinion mining prototype for tweets using the above processing modules. A function of the prototype includes:
(1) Analysing sentiments of re-tweeting comments along a re-tweeting tree, and through a re-tweeting sentiment graph, showing the sentiment turning point while the tweet is re-tweeted.
(2) Calculating numbers of positive and negative re-tweeting comments in a time internal and through a sentiment fluctuation line chart, presenting how sentiments aroused by a tweet change with time.
(3) Mining opinions in re-tweeting comments and through an opinion summarization chart, telling the hot opinions generated in re-tweeting procedure.

1.2 The main contributions of the paper are as follows:
• Proposing an efficient method to mine and summarize opinions using association rule mining algorithm.
• Building a micro blog-oriented sentiment lexicon based on real tweets corpus to deal with the specificity of micro-blogs such as emoticons and online language. We then propose an effective algorithm to calculate sentiment orientations.
• Building a real-time system to perform opinion mining for tweets, which observes interesting or abnormal phenomena. Experiments on a set of real-world Web tweets show that our method is efficient and performs well. Our goal was to demonstrate a practical approach to solving an alarming healthcare issue through a systematic, computational approach centred on mining useful patterns out of public data.

1.3 Hobb’s Algorithm:
• Works off “surface parse tree”
• Starting from the position of the pronoun in the surface tree)
  o First go up the tree looking for an antecedent in the current sentence(left-to-right, breadth-first);
  o Then go to the previous sentence, again with the same traversal transaction.
  o And keep going back and forth in order to repeat the same sort of steps.

• Intrasentential Anaphora:
  o Steps 2 and 3 deal with this intrasentential anaphora in order to incorporate basic syntactic constraints which also includes the diagrammatic representation. In the below diagram 6.1 John’s portrait of “him” has been cancelled or removed as it is intrasentential and then the entire verb and noun pronoun has been once for all as the pronoun. It’s clearly represented below in the diagram the split ups of the sentence where the looking for an antecedent takes place.

2. METHODOLOGY:

The methodology is divided into three main phases: Preprocessing Anaphora detection Sentiment analysis

2.1 Preprocessing

2.2 Anaphora detection

2.3 Sentiment analysis
2.1 Preprocessing
The tweets contain many unwanted and malicious content which is not required, called as noises which must be processed in order to retain clean data. Manually removing these unwanted noises takes a lot of time. But this can be achieved by programmatically. A program sequence (preprocessing-package) is developed to remove these noises such as links, hyperlinks, hashtags etc. The process also includes correcting misspelled words and replacing sms language words with actual words.

Pseudo code:
Input: Load [Tweets]
Logic:
Import [preprocessing-package]
For Sentence in tweets:
   If tweets are Not-Empty:
      For Word in Sentence:
         Word = Replace-wrong-words (Word);
         Word = Remove-Hyperlink (Word);
         Word = Remove-Link (Word);
         Word = Remove-Email-id (Word);
         Word = Remove-Hashtags (Word);
         Word = Remove-Mentions (Word);
      Output: Preprocessed [Tweets]

2.2 Anaphora detection
This is the main phase of this research project. We created separate modules for all POS tables and its related words. The dictionaries or hash tables were created using python language and was saved in a separate file. These files were later imported as packages. There are two separate packages created POS package and Emoji Package. Both the packages were able to detect words and emoticons over 80% of the time. This process begins with taking preprocessed tweets as input and finally generates new sets anaphoric values for next phase.

Pseudo code:
Input: Preprocessed [Tweets]
Logic:
Import [POS-package];
Import [Emoji-package];
Import csv;
For Sentence in Tweets:
   If Len (Sentence) equal to Len (A-Val):
      If A-Val greater than median-Valence:
         Tweets are positive;
         New-Val="Positive"
      If A-Val lesser than median-Valence:
         Tweets are negative;
         New-Val="Negative"
      If A-Val equal to median-Valence:
         Tweets are Neutral;
         New-Val="Neutral"
   Output: Sentiments of tweets with actual [Tweets]
2.4 Sample code:

```python
#Main file
import csv
import string

list_of_sentences={};

print("An Anophora Detection:")

with open('n.csv','r',encoding='utf-8') as f:
    reader=csv.reader(f)
    i=0;
    for col in reader:
        t=col[0].encode('unicode-escape'); #To encode as unicode value
        list_of_sentences[i]=t;
        i=i+1;

for i,j in list_of_sentences.items():
    print("Sentence ",i,"->",j,"Length of Sentence (",len(j.split()),")")

print("\n\nTotal Number of sentences uploaded=",len(list_of_sentences))
line()
print("press Enter to continue and n to cancel")
cont=input()
if cont.isalpha():
    if cont=='s':
        exit()
else:
    exit()

print("Pre-processing")

for i,j in list_of_sentences.items():
    print("Sentence ",i,"->",j) #To encode as unicode value
    print("Length of Sentence (",len(j.split()),")")

for i,j in list_of_sentences.items():
    print(\n    \n    Total Number of sentences uploaded=\n    \n    len(list_of_sentences))
line()
print("press Enter to continue and n to cancel")
cont=input()
if cont.isalpha():
    if cont=='s':
        exit()
else:
    exit()

print(Pre-processing')

for i,j in list_of_sentences.items():
    print("Sentence ",i,"->",j) #To encode as unicode value
    print("Length of Sentence (",len(j.split()),")")

for i,j in list_of_sentences.items():
    print(\n    \n    Total Number of sentences uploaded=\n    \n    len(list_of_sentences))
line()
print("press Enter to continue and n to cancel")
cont=input()
if cont.isalpha():
    if cont=='s':
        exit()
else:
    exit()

print(Pre-processing')

for i,j in list_of_sentences.items():
    print("Sentence ",i,"->",j) #To encode as unicode value
    print("Length of Sentence (",len(j.split()),")")

for i,j in list_of_sentences.items():
    print(\n    \n    Total Number of sentences uploaded=\n    \n    len(list_of_sentences))
line()
print("press Enter to continue and n to cancel")
cont=input()
if cont.isalpha():
    if cont=='s':
        exit()
else:
    exit()
```

**Temp:**

```python
#Main file
import csv
import string

list_of_sentences={};

print("An Anophora Detection:")

with open('n.csv','r',encoding='utf-8') as f:
    reader=csv.reader(f)
    i=0;
    for col in reader:
        t=col[0].encode('unicode-escape'); #To encode as unicode value
        list_of_sentences[i]=t;
        i=i+1;

for i,j in list_of_sentences.items():
    print("Sentence ",i,"->",j,"Length of Sentence (",len(j.split()),")")

print("\n\nTotal Number of sentences uploaded=",len(list_of_sentences))
line()
print("press Enter to continue and n to cancel")
cont=input()
if cont.isalpha():
    if cont=='s':
        exit()
else:
    exit()

print("Pre-processing")

for i,j in list_of_sentences.items():
    print("Sentence ",i,"->",j) #To encode as unicode value
    print("Length of Sentence (",len(j.split()),")")

for i,j in list_of_sentences.items():
    print(\n    \n    Total Number of sentences uploaded=\n    \n    len(list_of_sentences))
line()
print("press Enter to continue and n to cancel")
cont=input()
if cont.isalpha():
    if cont=='s':
        exit()
else:
    exit()

print("Remove misspelled words:")

print(\n\nRemove misspelled words:")

print("Pre-processing")

for i,j in list_of_sentences.items():
    print("Sentence ",i,"->",j) #To encode as unicode value
    print("Length of Sentence (",len(j.split()),")")

for i,j in list_of_sentences.items():
    print(\n    \n    Total Number of sentences uploaded=\n    \n    len(list_of_sentences))
line()
print("press Enter to continue and n to cancel")
cont=input()
if cont.isalpha():
    if cont=='s':
        exit()
else:
    exit()

print("Pre-processing")

for i,j in list_of_sentences.items():
    print("Sentence ",i,"->",j) #To encode as unicode value
    print("Length of Sentence (",len(j.split()),")")

for i,j in list_of_sentences.items():
    print(\n    \n    Total Number of sentences uploaded=\n    \n    len(list_of_sentences))
line()
print("press Enter to continue and n to cancel")
cont=input()
if cont.isalpha():
    if cont=='s':
        exit()
else:
    exit()
```

**Temp:**

```python
#Main file
import csv
import string

list_of_sentences={};

print("An Anophora Detection:")

with open('n.csv','r',encoding='utf-8') as f:
    reader=csv.reader(f)
    i=0;
    for col in reader:
        t=col[0].encode('unicode-escape'); #To encode as unicode value
        list_of_sentences[i]=t;
        i=i+1;

for i,j in list_of_sentences.items():
    print("Sentence ",i,"->",j,"Length of Sentence (",len(j.split()),")")

print("\n\nTotal Number of sentences uploaded=",len(list_of_sentences))
line()
print("press Enter to continue and n to cancel")
cont=input()
if cont.isalpha():
    if cont=='s':
        exit()
else:
    exit()

print("Pre-processing")

for i,j in list_of_sentences.items():
    print("Sentence ",i,"->",j) #To encode as unicode value
    print("Length of Sentence (",len(j.split()),")")

for i,j in list_of_sentences.items():
    print(\n    \n    Total Number of sentences uploaded=\n    \n    len(list_of_sentences))
line()
print("press Enter to continue and n to cancel")
cont=input()
if cont.isalpha():
    if cont=='s':
        exit()
else:
    exit()
```
num=0
for k in j.split():
    if not (k=='\n'):
        print(num,'"',k,'"',',lst[p])
num=num+1
p=p+1
lst=[]
for k in j.split():
    if not (k==' '):
        lst.append(k)
        num=num+1
p=p+1
print('Parts of Speech Detection succesfully completed!')
print('press Enter to continue and n to cancel')
cont=input()
if cont.isalpha():
    cont='s'
else:
    exit()
print('-----------------------------')
print('Parts of Speech Detection succesfully completed!')
print('Continued.............')

from sentimentdetection.Featurebased.Feature1 import context_approach
context_approach(list_of_sentences)

3. RESULTS

Finally, after building an opinion mining prototype for tweets using the processing modules and anaphoric module, we were able to analyse sentiments of tweet comments including re-tweets and we were able to display a sentiment as shown in our graph, showing the sentiment turning point while the tweet is posted. Calculating numbers of positive and negative tweet comments in a time internal and through a sentiment fluctuation chart. Mining opinions in tweet comments and through an opinion summarization. And we were also able to successfully complete anaphora detection with over 75% precise detection. A separate chart is shown which represents the efficiency of anaphora POS detection.

4. FUTURE WORK

In the future work we will be concentrating on unstructured data and will be working on unsupervised learning and classifying.

5. CONCLUSION

Anaphora resolution, using only syntactic and semantic constraints, without resorting to encyclopedia or world knowledge, has both a bright and a dark side. The temptation is to claim that there is so much more to anaphora resolution that could be seen from the data collected on the surface of the text; in terms of Natural Language Processing by computers. In my view, is just transferring the problem to a higher level of abstraction without solving it. Any solution bought to anaphora can contribute to other areas of language study such as ambiguity resolution, textual cohesion and eventually on reading comprehension and text production. The relationship between anaphora and ambiguity, for example, is so close that is probably impossible to refer to one without using the other. In more practical terms this paper has been successful in solving the Anaphoric resolution problems that most of the pre-proposed algorithms couldn’t at the highest. The entire analysis of the twitter data has been done and the tweets are taken for which it is broken down into grammatical words and then used a tool to manipulate it in this paper and thus has been successful in completing this particular task.

References
1. Whiting, A. and D. Williams, “Why people use social media: a uses and gratifications approach”
5. Hiroshi Shimodaira “Text Classification using Naive Bayes”
7. Ms. Harvinder Jeet Kaur, Mr. Rajiv Kumar, “Sentiment Analysis from Social Media in Crisis Situations” ICCCA, 2015
10. Abdellatif Rahmoun and Zakaria Elberrichi, “Experimenting N-Grams in text Categorization”, October 2007
11. Subhabrata Mukherjee and Akshat Malu Balamurali A.R and Pushpak Bhattacharyya. “Sarcasm detection in twitter data”


17. Haruna Isah, Paul Trundle, Daniel Neagu “Social Media Analysis for Product Safety using Text Mining and Sentiment Analysis”


20. David Bamman and Noah A. Smith “Contextualized Sarcasm Detection on Twitter”

21. Zhaoxia WANG, Victor Joo Chuan TONG, Xin XIN, Hoong Chor CHIN “Analysis on Social Media Data”