Detecting Points of View from News Articles

Tanmoy Patra

A Dissertation
Presented to the Faculty
of Indian Statistical Institute
in Candidacy for the Degree
of Master in Computer Science

Adviser: Dr. Debapriyo Majumdar

June 2015

Declaration

I, **Tanmoy Patra** (**CS1316**), registered as a student of **M.Tech** program in **Computer Science**, **Indian Statistical Institute**, **Kolkata** do hereby submit my Dissertation Report entitled "**Detecting Points of View from News Articles**". I certify

- The work contained in this Dissertation Report is original and has been done by me under the guidance of my supervisor.
- The material contained in this Dissertation Report has not been submitted to any University or Institute for the award of any degree.
- I followed by guidelines provided by the Institute in preparing the report.
- Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of report and giving their details in the bibliography.

.....

Place : ISI, Kolkata **Tanmoy Patra**

Date : July, 2015 **(CS1316)**

Certificate

This is to certify that the thesis titled "**Detecting Points of View from News Articles**" submitted by **Tanmoy Patra** in partial fulfillment for the award of the degree of **M.Tech** in **Computer Science** is a bonafide record of work carried out by him under our supervision. The thesis has fulfilled all the requirements as per the regulations of this Institute and, in my opinion, has reached the standard needed for submission. The results embodied in this thesis have not been submitted to any other university for the award of any degree or diploma.

.....

Dr. Debapriyo Majumdar

CVPR Unit

Indian Statistical Institute, Kolkata

Abstract

Conventional models concentrate on what the journalist perceives as news. But the news process is a two-way transaction, involving both news producer (the journalist) and the news receiver (the audience), although boundary between the two is rapidly blurring with the growth of citizen journalism and interactive media. Different newspapers have different perspectives from different journalists regarding a topic. In this thesis, we propose a novel approach to automatically extracts the multiple *Points of View* from different newspapers articles talking on a similar topic. Rather than ranking or summarisation of cluster topics, we try to bridge the gap of information a audience might have when we doesn't read multiple newspaper. Thus we can view the documents as being composed of different *Points of View* regarding a same event clustered by source and target, which we have to infer, and the visible variables which are the words of documents are just means of expressing these views and the weighted links of the graph defines the sentiment polarity, which is the data that we have. A sentence quotation can be viewed as describing a single event or maybe connecting different events of the document. Here we are dealing with extraction based on these opinion and identifying the polarity of these views of the multiple documents and giving the user Information dessert in form of *Points of View*.

Our approach is distinguished from existing approaches in that we use models to capture the *Points of View* after identification of important entities [Source,Target]. Work also involves picking up the sentences without paying attention to the details of grammar and structure of the documents and also stating the polarity of the respective views.

Acknowledgements

I would like to thank my advisor, Dr. Debapriyo Majumdar, for his support over the course of researching and documenting this work. He has been a steady source of insight and inspiration, while providing calm reassurance through challenging circumstances. Thanks also to Dr. Mandar Mitra and Dr. Utpal Garain, their sage advice and guidance helped greatly. Thanks also to my peers as I'll also never forget my conversations and interactions that helped shed light on vital questions of Information Retrieval, and of human thinking.

I want to dedicate it to my mother, father, without whom my life would not be possible. The smooth completion of this project has only been possible because of everyone's support and good will.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

These days, Internet users are frequently turning to the Web for news rather than going to traditional sources such as newspapers or television. The Internet is the richest source of opinion collection. Through opinions, humans can flux together diverse approaches, experiences, wisdom and knowledge of people for decision making. Humans like to take part in discussions and present their points of view. This trend is likely to continue, according to a recent report, which found that people who have been using the Web for several years are more likely to cut back on their reading of print newspapers than people with less Internet experience. Already, the New York Times online news source (http://nytimes.com) has traffic from over two million distinct users weekly, while the print edition circulation is just over one million copies. Google News (http://news.google.com), is visited by several million unique visitors over a period of few days. The number of items, news stories as identified by the cluster of news articles, is also of the order of several million. The challenge is in finding the right content that gives you all the perspectives: something that will answer your current information needs, without having to read through all newspapers.

Present day systems though effective doesn't take into account several factors and also creates a trade off between personalization and actual information a user is

entitled to have. But there's this shift in how information is flowing online, and it's invisible.

1.1 Present Systems

Google News

Google News is a computer-generated news site that aggregates news articles from more than 4,500 news sources worldwide, groups similar stories together and displays them according to each readers personalized interests or clicks. When queried on a news and you put the results side-by-side, you don't even have to read the links to see how different these two pages are. But when you do read the links, it's really quite remarkable. That's how different these results are becoming. Even if you're logged out, there are 57 signals that Google looks at everything from what kind of computer you're on to what kind of browser you're using to where you're located that it uses to personally tailor your query results Figure 1.1.

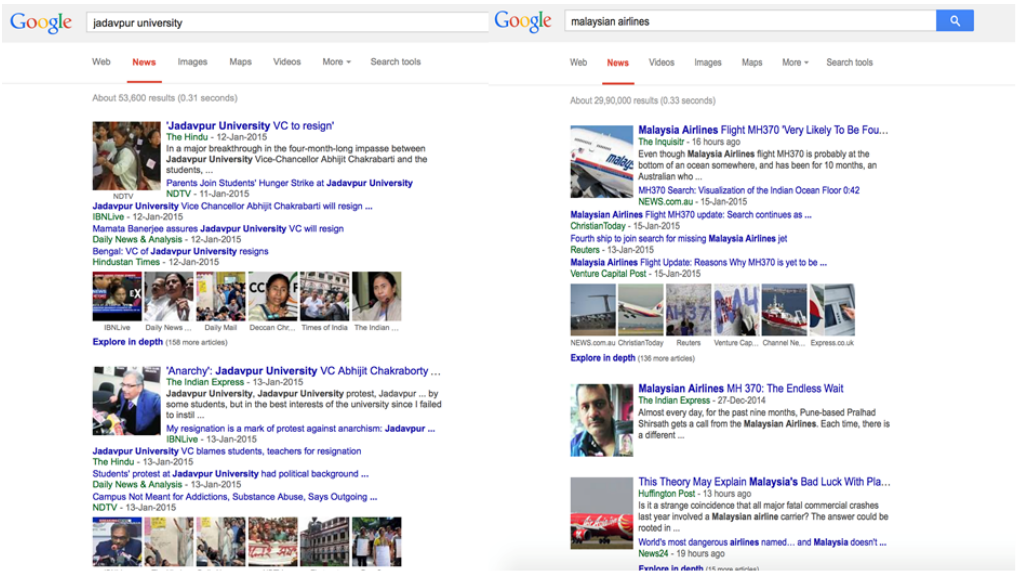

Figure 1.1: Google News

Facebook

The Facebook trending module has warped the minds of publishers around the Internet. The richer design shows personalized lists of the most mentioned words and phrases of the moment with short explanations of why each is blowing up. A click-through leads to a Page of mentions by friends, Pages, and public posts by anyone who lets people "Follow" them. Often home to inane news stories, days-old headlines, and a pusher of fake holidays it's also become an interesting if inconsistent tool for gauging what's going viral on the network. It is personalized by interest and the links you click on Figure 1.2.

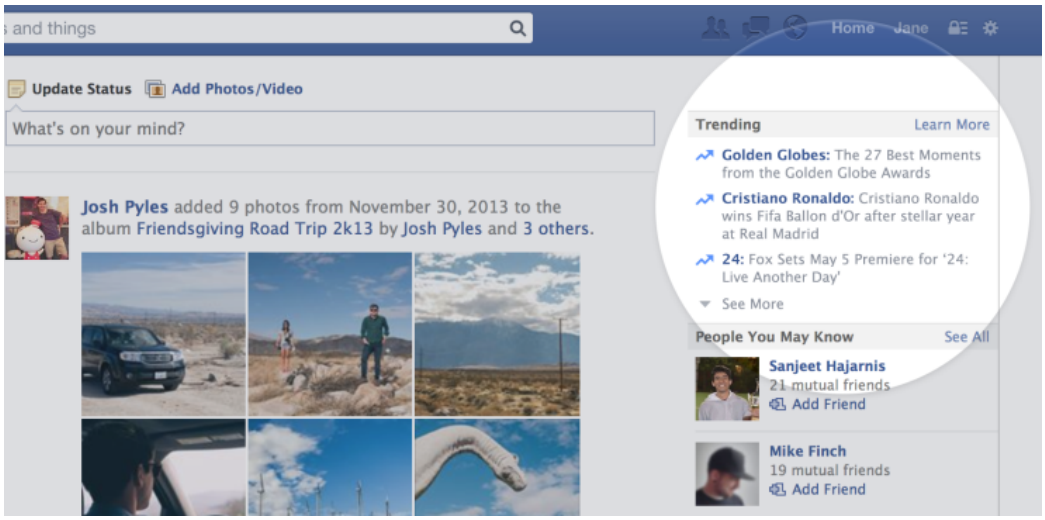

Figure 1.2: Facebook Trends

Twitter

Twitter, a micro blogging service, has emerged as a new medium in spotlight where users create status messages (called "tweets"). These tweets sometimes express opinions about different topics. On twitter a user can follow any other user, and the user being followed need not follow back. Being a follower on Twitter means that the user receives all the tweets from those the user follows. When a new topic be-

comes popular on Twitter, it is listed as a trending topic, which may take the form of short phrases. Facebook Trending aggregates the headlines of the day, while Twitter Trending Topics check the pulse of the moment Figure 1.3.

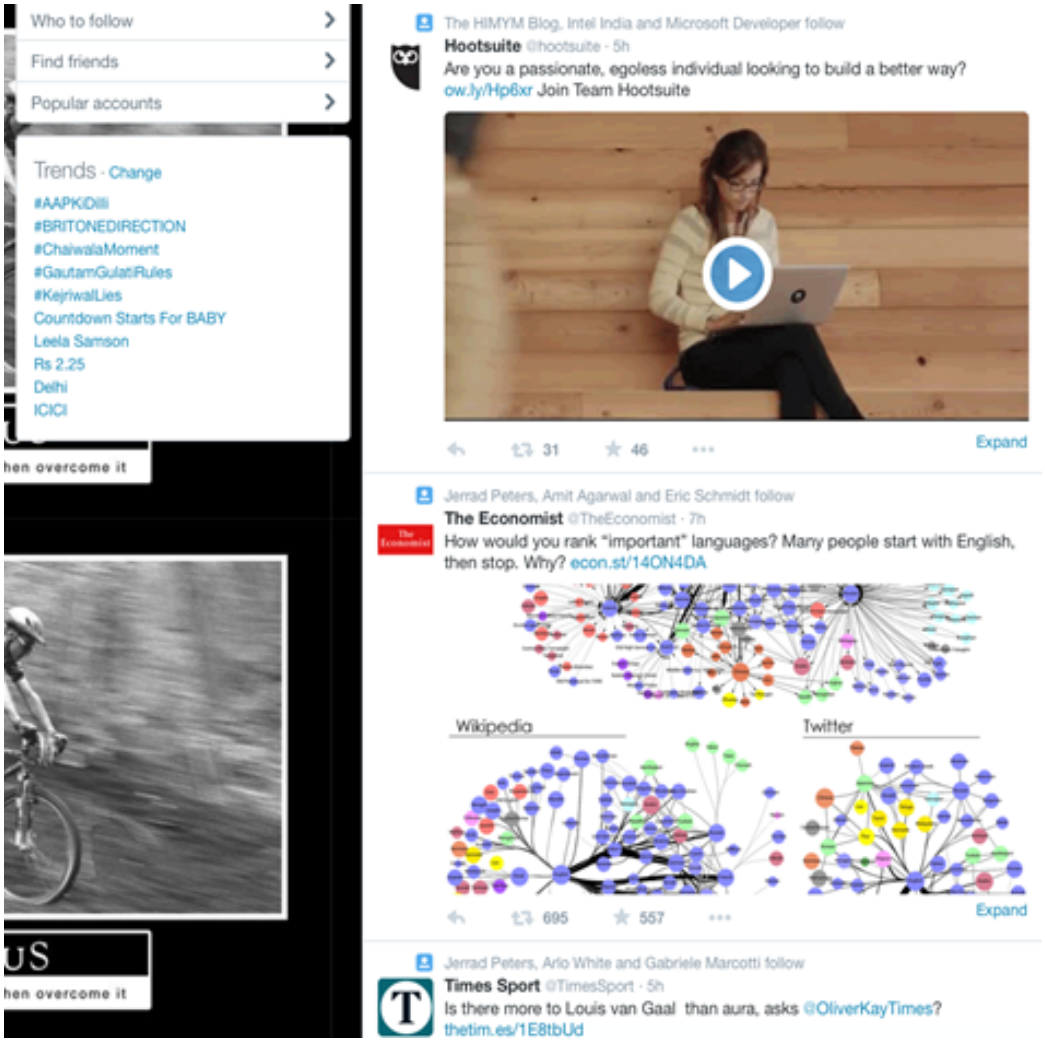

Figure 1.3: Twitter Trends

1.2 Motivation

Given the trend News reports are being produced and disseminated in over-whelming volume, making it difficult to keep up with the newest information. Again different newspapers state the same fact in different ways. Thus making it difficult for a user

to get all the perspectives that can be covered. Most previous research in automatic news organization treated news topics as a flat list, ignoring the intrinsic connection among individual reports. Focus was just on basic clustering based on title, text and topic.

The present day systems are sweeping the Web. There are a whole host of companies that are doing this kind of personalization. Yahoo News, the biggest news site on the Internet, is now personalized, different people get different things. Huffington Post, the Washington Post, the New York Times all flirting with personalization in various ways. And this moves us very quickly toward a world in which the Internet is showing us what it thinks we want to see, but not necessarily what we need to see. As a user, one would want to have all points of view on the topic or event. It gives us some information vegetables; it gives us some information dessert. We need to make sure that they also show us things that are uncomfortable or challenging or other important points of view such that they're transparent enough to give us a clear picture. Little has been done to define equivalent factors that determine audience perception of news. This is largely because it would appear impossible to define a common factor, or factors, that generate interest in a mass audience. Dataset collection in the form of source target and polarity on a same event not available and hence difficult.

Based on the current situation , it motivates us to define a few goals some of which are completed in the thesis and some would be future work.

1.3 Problem Statement

Detecting the *Points of view* from a collection of new articles representing an event. Our approach is distinguished from existing approaches in that we use models to capture the points of view after identification of important entities[Source, Target] Pick up the sentences without paying attention to the details of grammar and structure of

the documents and also stating the polarity of the respective views. Taking several factors into account our algorithm looks at text similarity to identify these goals :

- **Definite Goals :**
  - Identifying different points of views, in terms of statement (text) and sentiment (polarity). To measure the effectiveness of the system, we can measure

    Precision of points of views extracted: Out of those presented, how many are actually points of views.

    Diversity: How many are distinct points of views. Different people may be talking about the same thing in the same way. Are we able to put them together and distinguish between the different points of views?

    Recall: From a dataset we need a collection of all points of views (difficult). Then compute recall.

- **Usecase Goals :**
  - What insight do we get from this? Things such as

    In topic x, these few people are the most vocal persons.

    Can we say who has the most authority?

    Can we say who is very popular? Unpopular?

    Can we determine a degree of controversiality of the topic? [Lit survey]

1.4 Overview of Our System

Our work provides a methodology for capturing the multiple point of views from a collection of news articles representing a topic. The most important use case being the user gets to cover the entire spectrum without having to read through all newspapers.

It is different from the present systems in a way that it helps the user get the actual flow of information without any personalization, so every user gets to view the same set of point of views, and help them read through without any redundant information Figure 1.4. Documents are segregated into POVs. POVs in the same cluster are similar in terms of opinion and two different clusters have higher difference in terms of opinions. The Clusters are called pure if this separation is done efficiently. Identifying the point of controversy and the important source and target in the perspectives. Our approach is distinguished from existing approaches in that we use models to capture the points of view after identification of important entities [Source, Target]. Picking up the sentences without paying attention to the details of grammar and structure of the documents and also stating the polarity of the respective views. In our system we check for sentence similarity by making use of the inherent semantics of the sentence and of target entities and sentiments. Then clustering the point of views and group them based on a threshold set based on domain knowledge. Thus eliminating redundant or similar point of views and finally showcasing it to the users.

To visualize the nature of data, the model first constructs a weighted directed graph to reflect the relationships between the source and target and the thematic level words used to identify the sentiment polarity of the POVs. Weighing the edges as per sentence level occurrence of source and target and then aggregates the sentiment between the same Source and Target Figure 1.5.

Some of our objectives also involve identifying the main targets in an unsupervised way and if the multiple sources talk about the Common Target with what polarity. It helps us detect bias in the network regarding the opinion people have towards the common Target. Figure 1.6 and Figure 1.7 visually illustrates that in any news article survey on a topic has a few targets with most incoming links and hence they are the people who gives us controversiality of the network and influenced POVs.

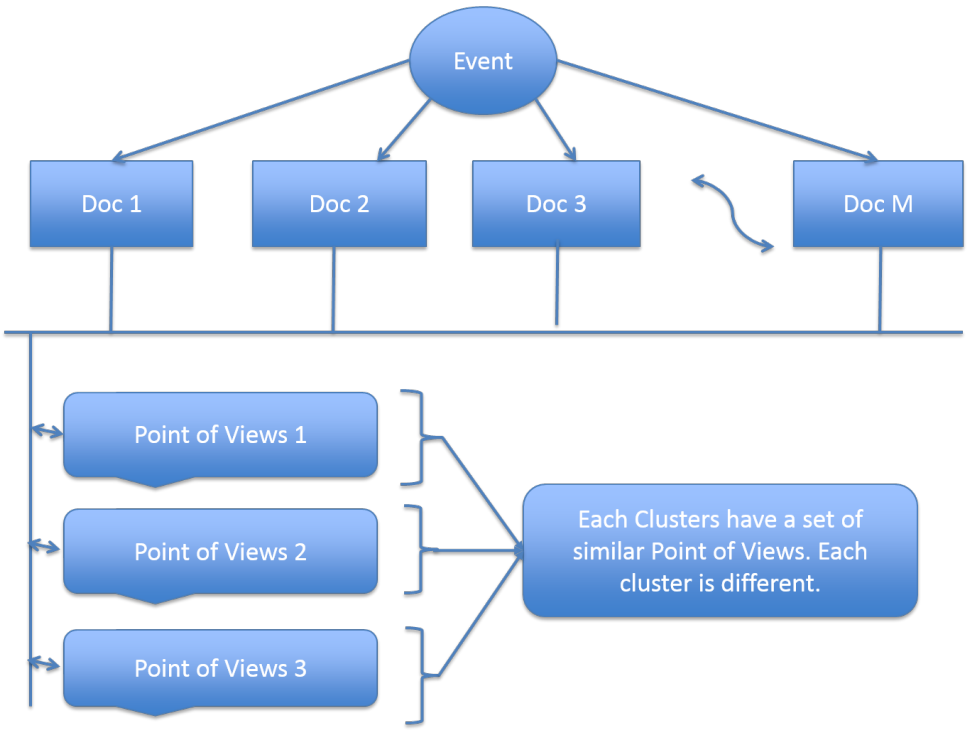

Figure 1.4: Basic Design 1

In the next chapters, we will slowly brief you with the related work in the field and the topic we are working on. Next Moving on to the chapters of explaining the algorithm and the experiment part. Appendices is there as a pre-requisite.

1.5 Our Contribution

As described in Section 1.4 This thesis thus presents a comparative study on the methods and resources that can be employed for mining opinions from quotations (reported speech) in newspaper articles. We show the difficulty of this task, motivated by the presence of different possible sources and targets and the large variety of affect phenomena that quotes contain. We do not claim that we will detect such instances.

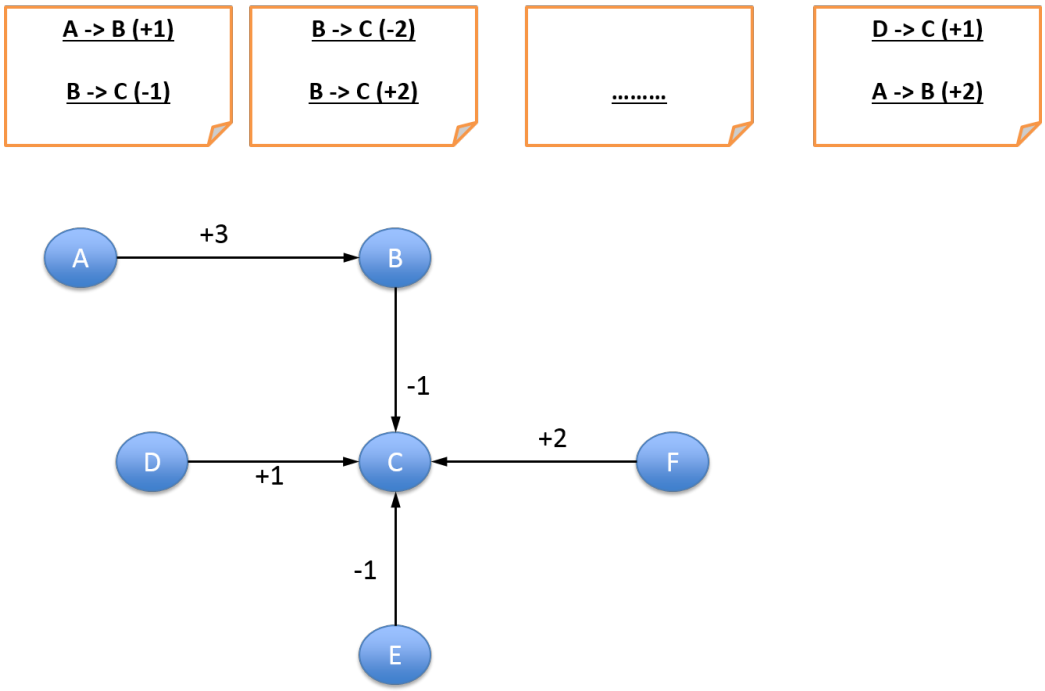

Figure 1.5: Aggregates Sentiment SRC $->$ TAR

Instead, we focus on detecting those (relatively) explicit opinion statements found in the news, and especially in quotations.

Another major difference between the news and product reviews is that the target of the sentiment is much less concrete. Since a quotation may contain multiple targets. It is also rather tricky to detect whether any negative sentiment detected refers to which of the target. We normalized it over multiple entities based on relevance of the entity. To find the relevance of the entity we used wikipedia dumps to identity correct entity map and relevance score. Instead of trying to tackle all of these complex issues, our current aim is to categorise quotations for subjectivity and to determine the polarity of the subjective quotations. Unlike full articles, quotations are relatively short and often are about one subject, hence we used similarity measures for short sentences and also giving weightage to both the target entities and the the polarity of the quote to determine clusters.

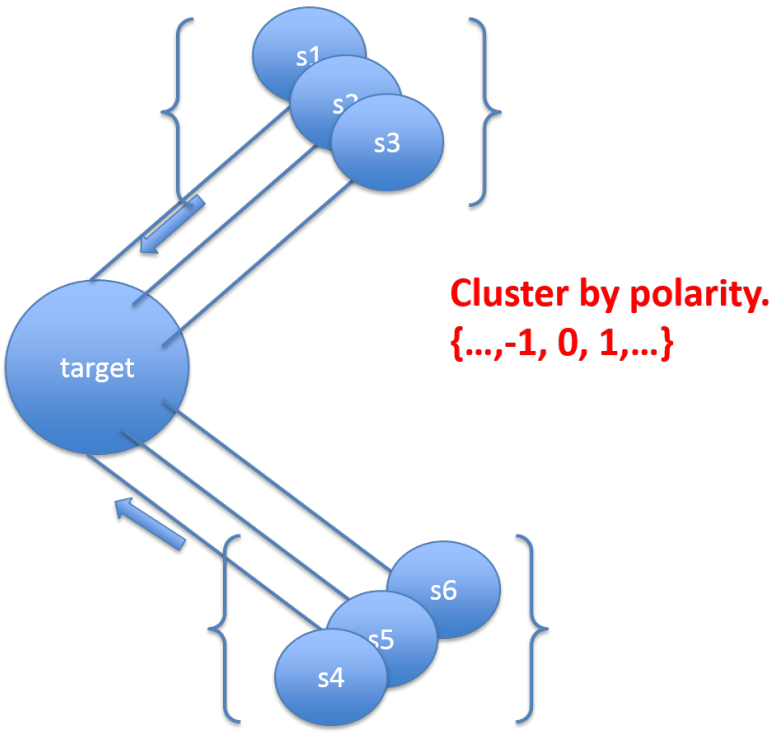

Figure 1.6: Cluster by Polarity SRC $->$ TAR

Further our system serves the purpose of cleaning information junk and gives the user the information in form of Points of Views. So that the user has a entire view of the spectrum covering the news. It also tries a revision procedure which is necessary to address the cohesion problems that cause some of the 'flow' problems. For instance, if a sentence begins with the pronoun 'he', but the reader cannot tell who 'he' refers to, the revision module should replace the pronoun with the correct name i.e Anaphora resolution should be done properly when such that when Point of views are produced such that the flow is not broken and it is more concrete that just document summarization. For identification of the amount of redundancy in Point of Views(POVs), and then showcasing the relevant POVs we propose a novel approach defining a similarity metric which not only considers text similarity measures but also the semantics, the target entities and sentiment of the quotations.To the best of our knowledge no work have prior been done on Perspective detection on granularity level

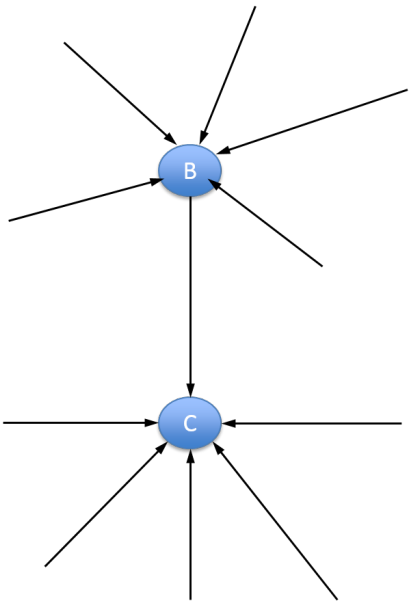

Figure 1.7: Identifying main Targets

more than just positive negative neutral. We also are able to give more concreteness to the problem of identifying similar short sentences in the form of POV.

Chapter 2

Related Work

In the past few years, there was a growing interest in mining opinions in reviews from both academia and industry. Opinion mining is the task of extracting from a set of documents opinions expressed by a source on a specified targets. As already mentioned extensive work has been conducted on opinion mining, at different levels of text and on different polarity scales. Applications include a variety of areas, depending on the source and final user of the extracted data from monitoring the image of public figures to company reputation or trust, monitoring and analyzing social media to detect potentially dangerous situations and what is done about them, or tracking opinion across time for market and financial studies.

Recent research in sentiment analysis is focused on Opinion Mining in customer reviews [Pang and Lee, 2008]. One major aspect of Opinion Mining in product reviews is the collection of sentiment bearing words [Harb et al., 2008, Kaji and Kitsuregawa, 2007] or the construction of complex patterns which represent not only the sentiment, but also extract the relationship between the sentiment and the features of the product [Kobayashi et al., 2005]. The point of view aspect plays less important role, because a customer expresses only one view (his/her own view) and different viewpoints mostly occur by comparisons of different prod-

ucts [Liu, 2010]. As shown in the examples, the viewpoints are almost essential in the news area domain and some statements do not have any sentiment without a viewpoint.

However, the existing work on news articles has been mainly focused on extracting and summarizing opinions from reviews using natural language processing and data mining techniques. The main objective of a news aggregation service is to provide the user with a single entry point for news articles that are likely to be of interest in an easily accessible form and covers entire spectrum. To achieve this, several approaches have been tried, some involve grouping articles that contain similar or related news. The challenge with grouping articles in this way is how to reduce the clusters into an overview that is easily accessible. News data is very different from product reviews in that sentiment is usually expressed much less explicitly. Bias or sentiment can be expressed by mentioning some facts while omitting others, or it can be presented through subtle methods like sarcasm. However, far too little attention has been paid to Opinion Mining in newspaper articles and especially the integration of viewpoints. Most of the approaches on this topic only work with single words/phrases [Wilson et al., 2009] or reported speech objects [Balahur et al., 2009, Balahur and Steinberger, 2009, Park et al., 2011], because quotations are often more subjective. The opinion holder (the speaker) can be deduced in the cases and sometimes even the object of the opinion (e.g. another person or an organisation which appears in the reported speech). But this technique can only capture opinions which are part of a quotation in the news. Some work in the field of Tracking Point of View narrative has been prior done in [Wiebe, 1994]. Combining this with the sentiment and perspective identification can help us derive all the point of views.

Previous work has attempted to automatically generate a multi-document summary of news articles using natural language processing or sentiment analysis tech-

niques [Godbole et al., 2007]. They presented a system that assigns scores indicating positive or negative opinion to each distinct entity in the text corpus and associates sentiment to entity. But their work doesn't look at the inherent textual property to showcase the different important perspective in news. Another relatively similar work [Balahur et al., 2009] presents a comparative study on the methods and resources that can be employed for mining opinions from quotations (reported speech) in newspaper articles. But again it involved both specialised training and testing data, doesn't make use of relevance of entities. Our system not only is unsupervised but also collects data from multiple domain, such that the audience has perspectives from different newspaper articles on the same topic. In most cases this is done with some form of ranking. However, with each new article comes a potential new viewpoint on a topic. With news aggregators that return ranked results it is difficult to get an overview of what are the various opinion over a range of articles on the same topics.

To determine the similarity of the various perspectives, the various types of work that have been done involves similarity metric for texts ,paraphrase detection and then clustering them based on the similarity score in common clusters. Measuring the similarity between sentences is an important problem that is relevant to many areas of language processing, including the identification of text reuse [Seo and Croft, 2008], textual entailment [Zanzotto et al., 2009], paraphrase detection [Barzilay and Lee, 2003, Fernando and Stevenson, 2008], short answer grading [Pulman and Sukkarieh, 2005, Mohler and Mihalcea, 2009], recommendation [Tintarev and Masthoff, 2006], and evaluation [Papineni et al., 2002, Lin, 2004].

Many of the previous approaches to measuring the similarity between texts have relied purely on lexical matching techniques, for example [Baeza-Yates et al., 1999, Papineni et al., 2002, Lin, 2004].In these approaches the similarity of texts is computed as a function of the number of matching tokens, or sequences of tokens,

they contain. However, this approach fails to identify similarities when the same meaning is conveyed using synonymous terms or phrases (for example, "The dog sat on the mat" and "The hound sat on the mat") or when the meanings of the texts are similar but not identical (for example, "The cat sat on the mat" and "A dog sat on the chair"). Significant amounts of previous work on text similarity have focussed on comparing the meanings of texts longer than a single sentence, such as paragraphs or documents [Baeza-Yates et al., 1999, Seo and Croft, 2008, Bendersky and Croft, 2009]. There are also a few work which decides a score for sentence similarity based on relevance of entities [Feng et al., 2008] , Word sense disambiguation-based sentence similarity [Ho et al., 2010] and WordNet-based similarity measures [Patwardhan and Pedersen, 2006] are more in trends.

Chapter 3

Our Algorithm

3.1 Reflections on Analyzing News Article Data

Context drives meaning while TF-IDF is a powerful tool that's easy to use, our specific implementation of it has a few important limitations that some of the related works conveniently overlooked. One of the most fundamental is that it treats a document as a "bag of words," which means that the order of terms in both the documents itself does not matter. For example, querying for "Obama Mr." would return the same results as "Mr. Obama" if we didn't implement logic to take the order into account or interpret the documents as a collection of phrase as opposed to a pair of independent terms. But obviously, the order in which terms appear is very important. In performing an n-gram analysis to account for collocations and term ordering, we still face the underlying issue that TF-IDF assumes that all tokens with the same text value mean the same thing. Clearly, however, this need not be the case. A homonym is a word that has identical spellings and pronunciations to another word but whose meaning is driven entirely by context, and any homonym of your choice is a counterexample. Homonyms such as book, match, cave, and cool are a few examples

that should illustrate the importance of context in determining the meaning of a word.

Cosine similarity suffers from many of the same flaws as TF-IDF. It does not take into account the context of the document or the term order from the n-gram analysis, and it assumes that terms appearing close to one another in vector space are necessarily similar, which is certainly not always the case. As with TF-IDF, the obvious counter-example is homonyms. Our particular implementation of cosine similarity also hinges on TF-IDF scoring as its means of computing the relative importance of words in documents, so the TF-IDF errors have a cascading effect. So we are using an approach that not only looks at the semantics of the sentences and but also on the entities in the sentence which we called Target, alongside the sentiment of the sentence to determine the sentence similarity.

Detecting different Points of View

Quotations are ubiquitous in journalism. They are used to support claims and perspectives identified by the journalist. Through inclusion of quotations from experts, witnesses, persons involved, or observers, the news is made more concrete and more personal. A catchy sound bite may be quoted under multiple contexts over time as the news story evolves. Quotes that relate to each other can be seen as a thread that runs through an event or topic as it unfolds in news discourse.

- An NLP technique for identifying quotes and speakers in an article;
- Identifying the named-entities and mark them by relevance based on Wikipedia or higher confidence score;
- A methodology for selecting quotes relevant to a user query;
- A simple, tunable metric for scoring quote similarity in order to filter and cluster related quotes into threads fast and effectively;

- Tuning of the similarity metric and exploration of a quotes surrounding context and other perspectives.

What did (speaker) say about (subject) ?

where (speaker) and (subject) are specified by users. The (speaker) parameter could be specified as a specific entity (e.g. Obama), a facet (a type or category of entities, e.g. politician, basketball player), or anyone. And (subject) can be specified as combination of keywords, entities, and/or facets.

The source of a quotation is not always explicitly mentioned. It may not be located near the quotation, so syntactic parsing and named entity recognition are necessary. The use of pronouns is also common, such that anaphora resolution is needed to determine the name of the source. Our belief about the identity of the speaker for the sentence,

"If I had known about Michaels agenda, I would have done something differently," he said."

would change depending on whether it was preceded by Sentence prior. Due to ambiguities at different levels of text processing, automatic quote extraction does not guarantee perfect results. These are challenging NLP questions under active research [Das and Smith, 2009, MacCartney and Manning, 2007], but current analyzers have not reached the same level of maturity as some other NLP applications to be broadly applied to arbitrary text. Quote threads are clusters of related quotes. Specifically, a similarity score is computed between all pairs of relevant quotes, and nodes representing the quotes are linked if their similarity score surpasses a (tunable) threshold value set based on domain knowledge.

Entity-Centric Analysis

One interpretation is being able to detect the entities in sentence as described in brief earlier and using those entities as the basis of analysis, as opposed to just text-centric

analysis of the sentence or quotation involved. We simply extracted all the nouns and noun phrases from the sentence used to wikipedia dump to indentify the right and relevant entities and indexed them as entities appearing as Targets from the quotation text. The important underlying assumption being that nouns and noun phrases qualify as entities of interest. This is actually a fair assumption to make and a good starting point for entity-centric analysis. The results are annotated according to Penn Treebank conventions.

A certain amount of noise in the results is almost inevitable, but realizing results that are highly intelligible and useful, even if they do contain a manageable amount of noise is a worthy aim.The amount of effort required to achieve pristine results that are nearly noise-free can be immense. In fact, in most situations, this is downright impossible because of the inherent complexity involved in natural language and the limitations of most currently available toolkits, including NLTK. Though we made certain assumptions about the domain of the data through wikipedia link up and defined entities by the once most relevant with higher confidence score based on he topic the news articles were collected on.

An obvious starting point is to create a "golden set" of entities that we believed is absolutely crucial for a our algorithm to extract correct entities from the quotations as this helps us get a correct mapping from a subset of the entity phrase to actual entity phrase. For Example modi gets mapped to Narendra Modi ranked based on its confidence score, and then use this list as the basis of evaluation.

Identifying the centre of controversy

In this we take the quotations that represents a concept or point of view, and determine the Target entities, discover a group of other Target entities who apparently are always being opinionated about with the same sentiment. Hence we give lesser importance to the source as aggregate the sentiment of the Targets over the differ-

ent articles and identify if they are the point of controversy or important entities surrounding the controversy. A handy way to model data involving people and the things or people or Targets that they're interested in is called an interest graph; this is the primary data structure that supports analysis. In network science terminology, we now have what is called an ego graph, because it has an actor (or ego) as its focal point or logical center, which is connected to other nodes around it. An ego graph would resemble a hub and spokes if you were to draw it. Here the links are from Source to Targets with the edge weights as it sentiment. A Target is said to be seen in a negative light if it has too many incoming edges with negative polarity and positively in-fluent otherwise.

- How popular is the target entity ?
- How engaged are the various sources about the target and in what light?
- What are the most common topics and various point of views?

3.2 Algorithm

3.2.1 Phase 1

Documents are represented as random mixtures over latent Point of Views(POVs), where each POV is characterized by a distribution over words. Assuming we have a corpus of D documents. For each D_i where $i \in \{1, \ldots, N(D)\}$ consisting of a few POVs. We define the quotations to be the point of views and hence are extracted from the documents. For every D_i there might be different number of POVs. Quotation extraction is the task of extracting the content span of all of the direct quotations within a given document. More precisely, we consider quotations to be acts of communication and hence define it as POV. For each D_i, the processing includes the following steps:

- Sentence splitting : Split the document into paragraphs, and paragraphs into sentences.

- Parsing : For each sentence, apply linguistic parsing to assign part-of-speech tags (e.g. nouns, verbs), perform lexical analysis (e.g. detecting phrases), and determine grammatical roles (e.g. subjects, verbs, objects).Identify the quotations using Regex. Identification of Source and Target as defined earlier is also done.

- Coreference resolution : Link multiple mentions of the same entity across the whole document, including resolving pronoun coreference (e.g. "he", "him", "she"), aliases and abbreviations (e.g. "Bill Gates" referred to as "Gates", "General Mills" as "GM", "Alaska Airlines" as "Alaska"), and definite-noun anaphora (e.g. "the president", "the coach"). The coreference resolution step is very important to determining quotation attributions, because very often the speaker's full name is not provided for a given quote. Instead, the writer typically uses pronouns ("he said"), partial names ("said Gates"), or definite nouns ("the president said"). Similarly, in quotations, entities are often mentioned as aliases or pronoun anaphora. Applying coreference resolution would help identify such mentions, that otherwise would be missed by keyword matching techniques. Below is an example of coreference resolution. As the result, the quote is attributed to President Obama.

  - *I sensed a bit of frustration during President Obama's State of the Union address last month when he said, "The longer it [the health-care overhaul] was debated, the more skeptical people became."*

  For coreference resolution we used wikipedia dumps and Alchemy API whcih internally looks up Wikipedia to identify the respective maps, to create a list of

entity phrase to all possible entity phrase matching and selected the one with the highest confidence score.

- Topic and Entity tagging : To each entity, we assign its type and topic categories. For example, we tag entity "Michael Jackson" with type 'person' and topic [musician].

- Entity Disambiguation : Apply entity disambiguation such that each mention of an entity is linked to an entry in our repository of entities. As the result, different mentions of an entity are all marked with a unique identifier. This is obtained my the inverted map we created from an entity to a list of entity wikipedia provides on the same and we sorted them by the relevance score and sorted by topic as explained in previous bullet.

Relation Detection in Quotation:

- Action-modifier field: store context modifiers of the quotation, with entities marked up.

- Object field: store the actual quote, marked up with the entities recognized in the quote. During processing of documents, we index the subject-action-object relations extracted from all the sentences, not just from quotes.

For entities identified both within and outside of the quote, we index not only the entity names, but also their unique identifiers and assigned categories (i.e. topics).

- Speaker: the main subject of the verb, as well as its mod- ifier, such as title and affiliation of the speaker (e.g. given "said Microsoft chairman Bill Gates ...", we rec- ognize "Bill Gates" as the speaker, with "Microsoft" and "chairman" as the modifiers)

- Verb: quotation verb. In addition, we store the preposi- tional modifiers of the verb. The modifiers usually provide context of the quote being made (e.g. given

"said Bill Gates in the Microsoft shareholder meeting in Seattle", the modifiers are "in Seattle" and "in the Microsoft shareholder meeting")

- Quote: actual quote within the beginning and ending quotation marks. Note that a quote could span multiple sentences. We search for starting and ending quotation marks from the neighboring sentences, and determine the proper quote boundaries. Then, we store all the segments of the same quote.

The subject-modifier field would support search for quotes made by speakers of certain properties. Similarly, the action-modifier field supports searching for quotes within a particular context.

- Analysis of Patterns for Candidate Source : We are considering the Source as the one who has said the quote regarding the topic involving a few other named entities.
- Analysis of Patterns for Candidate Target(s) : In the sentence above we parse the sentence after determining the source and then look for the various named entities in the quotation and select the targets based on some threshold of relevance of the entity (people , location, place, organization)
- Challenge : In some cases the target (aspect) may not be in the text as well, it may be understood from the context or meta-data. A movie review, the name of the movie may not be explicitly mentioned
- Analysis of Directed graph between Source & Target :
  - The model first constructs a weighted directed graph to reflect the relationships between the source and target and the thematic level words used. Weighing the edges as per sentence level occurrence of source and target.
  - It then applies the graph-based algorithm to create a directed graph and clustering the links with similar opinion to identify the network.

* Cluster by Source - Target
* Cluster by polarity
* Combines all to obtain the graph based on Source, Target, Polarity Aggregation

In a second effort towards opinion mining, we applied sentiment analysis to reported speech quotations from one person about another, somewhat similar methodology was applied in [Balahur and Steinberger, 2009], with the objective of identifying support or criticism relationships between people. [Tanev, 2007] had developed software to detect such relationships in news texts and to construct signed social networks showing these relationships. Similarly we used Stanford Sentiment Analysis Api [Socher et al., 2013] to determine the sentiment score of the quotations, which is based on Recursive Neural Network that builds on top of grammatical structures. We tweaked it a bit to give us Sentiment scores which are real numbers in range[-1,1].'-1' defines it as entirely negative while '+1' is entirely positive.

3.2.2 Phase 2

Clustering similar point of views(POVs). Since the multiple documents contains a set of POVs , some of them might be redundant as may correspond to the same POV in different document. For identification of the amount of redundancy we propose a novel approach defining a similarity metric which not only considers text similarity measures but also the semantics, the target entities and sentiment of the quotations. We propose that two quotations A & B are identical if it has a similarity score of more than certain threshold. Initial threshold is decided based on domain knowledge and clustering of similar POV is done. Entity identification is done on the quotation to identify the Targets. Targets are identified again using the NER which is trained on the wikipedia, to give us an updated information also about the topic.

Near-duplicates are documents that are nearly, but not exactly, the same. The first two methods for measuring similarity involve identifying matching chunks of text in the POVs. Therefore one key challenge was to eliminate redundant information in the produced views. Articles about the same event often contains description of the same fact using different wordings. Hence to address this issue we also need to identify paraphrases. Next step then was to find the Semantic Similarity between the identified POVs in D_i with all other POVs. POV are made up of words, so it is reasonable to represent a sentence using the words in the sentence. Unlike classical methods that use a precompiled word list containing hundreds of thousands of words, our method dynamically forms the semantic vectors solely based on the compared POVs. Recent research achievements in semantic analysis are also adapted to derive an efficient semantic vector for a sentence.Given two POVs, the measurement determines how similar the meaning of two sentences is. The higher the score, the more similar they are. Here for text similarity we used the work on WordNet by [Li et al., 2006, Pedersen et al., 2004] because of its effectiveness.

Measuring similarity (MS1) : There are many proposals for measuring semantic similarity between two synsets: [Wu and Palmer, 1994, Resnik, 1998]. In this work, we experimented with two simple measurements:

$$Sim(s,t) = 1/distance(s,t). \tag{3.1}$$

where distance is the path length from s to t using node counting.

Measuring similarity (MS2) : This formula not only took into account the length of the path, but also the order of the sense involved in this path:

$$Sim(s,t) = SenseWeight(s) * SenseWeight(t)/PathLength \tag{3.2}$$

where s and t: denote the source and target words being compared.

- SenseWeight: denotes a weight calculated according to the frequency of use of this sense and the total of frequency of use of all senses.

- PathLength: denotes the length of the connection path from s to t.

Given two sentences X and Y, we denote m to be length of X, n to be length of Y. The major steps of the paper [Li et al., 2006, Pedersen et al., 2004] to capture semantic similarity between two sentences are described.

- Each sentence is partitioned into a list of words, and we remove the stop words.

- Perform word stemming.

- Perform part of speech tagging.

- Word sense disambiguation.

- Building a semantic similarity relative matrix R[m, n] of each pair of word senses, where R[i, j] is the semantic similarity between the most appropriate sense of word at position i of X and the most appropriate sense of word at position j of Y. Thus, R[i,j] is also the weight of the edge connecting from i to j. If a word does not exist in the dictionary, we use the edit-distance similarity instead and output a lower associated weight.

- We formulate the problem of capturing semantic similarity between sentences as the problem of computing a maximum total matching weight of a bipartite graph, where X and Y are two sets of disjoint nodes. We use the Hungarian method to solve this problem.

- The match results from the previous step are combined into a single similarity value for two sentences. There are many strategies to acquire an overall combined similarity value for sets of matching pairs. In the previous section, we presented two simple formulas to compute semantic similarity between two

word-senses. For each formula, we apply an appropriate strategy to compute the overall score:

$$\frac{2 \times Match(X, Y)}{(X)| + |(Y)|} \tag{3.3}$$

where match(X, Y) are the matching word tokens between X and Y. Matching average: This similarity is computed by dividing the sum of similarity values of all match candidates of both sentences X and Y by the total number of set tokens. An important point is that it is based on each of the individual similarity values, so that the overall similarity always reflects the influence of them. We apply this strategy with the MS1 formula.

$$\frac{2 \times |(X \bigcup Y|}{|X| + |Y|} \tag{3.4}$$

Dice coefficient: This strategy returns the ratio of the number of tokens that can be matched over the total of tokens. We apply this strategy with the MS2 formula. Hence, Dice will always return a higher value than Matching average, and it is thus more optimistic. In this strategy, we need to pre-define a threshold to select the matching pairs that have values exceeding the given threshold. (Cosine, Jaccard, Simpson coefficients can also be considered).

Further we go to identify the Targets in the POVs. Targets are Named-Entities with high relevance score as per Wikipedia. They are generally the names of People, Organization or Location. They key challenge has always been that the sentiment the Source(S) has in his quotation are distributed to the Target(T) entities identified. But case may be that regarding some Target. So if the Sentiment Toolkit says that the quote is negative we have a directed weighted graph $S - T$, with weight of the sentiment toolkit output. But Source might not have been negative about some of

the Targets. The graph is basically to understand the nature of the dataset. It also give us a few more insights like if further analysis is done.

3.2.3 Phase 3

Given two POVs, $POV1$ and $POV2$. To define a similarity score for how similar the different POVs are we define a metric involving the target entities, text similarity approach used in [Li et al., 2006] and sentiment. There are 3 factors and a metric designed combining the 3, gives us the new similarity metric. We use Jaccard to match the entity.

$$Sim(POV1, POV2) = X$$

$$OurSim(POV1, POV2) = X + \frac{\alpha E + \beta S}{\alpha + \beta} \times (1 - X)$$

- Where X gives the similarity score based on semantics as per the work in [Li et al., 2006], value lies in the range 0 to 1.
- And E is the entity similarity score based on jaccard similarity ordered by relevance of wikipedia on Target entities in POV1 and POV2
- And S represents the sentiment score based on the thematic level sentiment similarity of POV1 and POV2 alongside thresholds set on X and E .
- α, β are the respective weightage of the entity and sentiment. Experimentally we chose $\alpha = 5$ and $\beta = 2$.

Two POVs are put in the same cluster if the OurSim(POV1,POV2) based on K-Means Clustering (A.3) . But for K-Means to work, we need to identify the right value for K. This was determined by the well known elbow method (A.3) for identifying the right number of clusters. The estimation of the optimal number of clusters within a set of data points is a very important problem, as most clustering algorithms need

that parameter as input in order to group the data. If you graph the percentage of variance explained by the clusters against the number of clusters, the first clusters will add much information (explain a lot of variance), but at some point the marginal gain will drop, giving an angle in the graph. The number of clusters are chosen at this point, hence the "elbow criterion" [Tibshirani et al., 2001, Chen et al., 2002]. The computation of the gap statistic involves the following steps as per the original paper:

- Cluster the observed data, varying the number of clusters from $k = 1, ..., k_{\max}$, and compute the corresponding W_k.
- Generate B reference data sets and cluster each of them with varying number of clusters $k = 1, ..., k_{\max}$. Compute the estimated gap statistic $\text{Gap}(k) = (1/B)\sum_{b=1}^{B} \log W_{kb}^* - \log W_k$.
- With $\bar{w} = (1/B)\sum_b \log W_{kb}^*$, compute the standard deviation $\text{sd}(k) = [(1/B)\sum_b (\log W_{kb}^* - \bar{w})^2]^{1/2}$ and define $s_k = \sqrt{1 + 1/B}\,\text{sd}(k)$.
- Choose the number of clusters as the smallest k such that $\text{Gap}(k) \geq \text{Gap}(k + 1) - s_{k+1}$.

The silhouette coefficient [Rousseeuw, 1987] is one such measure. For each POV 'p', first find the average distance between p and all other points in the same cluster (this is a measure of cohesion). Then find the average distance between p and all points in the nearest cluster (this is a measure of separation from the closest other cluster).

$a(i)$ be the average dissimilarity of POV(i) with all other POVS within the same cluster. We can interpret $a(i)$ as how well i is assigned to its cluster (the smaller the value, the better the assignment). We then define the average dissimilarity of point i to a cluster c as the average of the distance from i to points in c.

Let $b(i)$ be the lowest average dissimilarity of POV(i) to any other cluster, of which POV(i) is not a member. The cluster with this lowest average dissimilarity is

said to be the "neighbouring cluster" of i because it is the next best fit cluster for point i. Silhouette as per wikipedia is defiend as :

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

Which can be written as:

$$s(i) = \begin{cases} 1 - a(i)/b(i), & \text{if } a(i) < b(i) \\ 0, & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1, & \text{if } a(i) > b(i) \end{cases}$$

From the above definition it is clear that $-1 \leq s(i) \leq 1$

For $s(i)$ to be close to 1 we require $a(i) \ll b(i)$. As $a(i)$ is a measure of how dissimilar i is to its own POV cluster, a small value means it is well matched. Furthermore, a large $b(i)$ implies that i is badly matched to its neighbouring cluster. Thus an $s(i)$ close to one means that the datum is appropriately clustered. If $s(i)$ is close to negative one, then by the same logic we see that i would be more appropriate if it was clustered in its neighbouring cluster. An $s(i)$ near zero means that the datum is on the border of two natural clusters. The average $s(i)$ over all data of a cluster is a measure of how tightly grouped all the data in the cluster are. Thus the average $s(i)$ over all data of the entire dataset is a measure of how appropriately the data has been clustered. Next we used the Within Sum of Squares approach to plot a graph to measure the goodness of a clustering structure without respect to external information. On the basis of this two plots we identified K.

After identification of right K, the cluster of POVs are created. POVs in the same cluster have similar POV and different clusters have difference in the POVs. The novelty of the approach comes from the fact that we consider multiple factors to

define a POV similarity. In the next chapter we will go forward with the evaluation part.

Chapter 4

Experimental Evaluation

4.1 Data Set and Pre-Processing

To evaluate our method we initially needed a collection of articles on the same topic. We used the New York Times Article Search V2 API to download a few such collections, each belonging to one topic. For this experiment, we then wrote a script to collect a few news articles each all on same topic query as our testing set. API monitors functionality may include breaking news, categorization of news according to pre-defined categories or user-defined search words, linking of related news over time, extraction and display of meta-information such as references to locations, persons and organizations, or quotations. Articles are collected by querying to the API and then parsing the news URL from the meta-data to get the content. Articles have text embedded inside along with a lot of unwanted boilerplate, ads, copyright messages etc. We segmented the article to just get the article text and threw away the rest. This is an information extraction problem. Most sites use HTML parsing along with a lot of other heuristics. We used reg-ex and a few HTML cleaners to get raw text. We further developed a parser based on NLP tools to sanitize and normalize the articles,

further among other features, we cleaned the content that weren't properly encoded in UTF-8 and stripped them of special characters to get well formatted text articles.

Articles are a lot of text, that include all kinds of conjunctions, connectives, pronouns, nouns, numbers, etc. When we are considering news, we are more interested in putting articles from an incident (or event) together. For example, you want articles from about racism in America to come together, rather than all articles on racism from around the world. So the query consists of "named entities" (proper nouns) which are best suited to characterize an incident. NYTimes API gives a lot of flexibility in querying. We used regex to parse the quotation. In this case the speaker of the quotation is called the Source entity and the named entities in the quote text are called Target entity.We then used this dataset to comparatively analyse the different possible methods and resources for opinion mining and we explored the possibility to combine them in order to increase the accuracy of the clustering that we are later going to do for point of view aggregation. The first approach is based on a "bag of words" the use of different lexicons containing positive and negative words. The second approach contemplates measuring similarity to corpora. Several Similarity measures are explained in the Appendix. Our proposed method is explain in 3.2

The source of a quotation is not always explicitly mentioned. It may not be located near the quotation, so syntactic parsing and named entity recognition using the tagger discussed prior was used. We used the Stanford Part of Speech (POS) tagger to identify the NNP in the quotation, as those are the Target entities. Then we mapped the entity to the correspoding relevant entity by looking up the wikipedia dump and identifying the relevant entity. Initially we bring an article to its vector form. A vector of keywords and weights - that depict importance to the article. Experiment are performed keeping in track the relevance score of every entity and calculating the confidence score thresholds for eliminate junk data. Where our system returns score for each of its annotations representing the amount of confidence the

engine has in the result. If you prefer to avoid false-positives in your application you may want to ignore results below a certain threshold. The best way to find an appropriate threshold is to run a sample set of your documents through the system and then manually inspect the results.

Article Search API v2 : With the Article Search API, you can search New York Times articles, retrieving headlines, abstracts, lead paragraphs, links to associated multimedia and other article metadata. Note: In URI examples and field names, italics indicate placeholders for variables or values. Brackets [ ] indicate optional items. Parentheses ( ) are not a convention when URIs include parentheses, interpret them literally.

The url collected from the json format metadata of the search query for news is being parsed and cleaned to collect news articles in text format. News articles related to same topic are collected. Quotations are parsed. Quotations are crucial carriers of information, particularly in news texts, with up to 90% of sentences in some articles being reported speech [Qu et al., 2004]. Finally we are going to show results with two datasets. Bad clusterings have purity values close to 0, a perfect clustering has a purity of 1. The details of the datasets are given in Table 4.1

In this section we describe four experiments aimed and comparing the effectiveness of the different point of views clustering with our proposed method, explained in 3.2. Our evaluation methodology is the same in all four experiments, baseline being the Cosine Similarity model(cosPOV) [Singhal, 2001] and Minhash Similarity using Jaccard(hashPOV) [Manku et al., 2007]. The baseline approaches are explained in appendices Section A.2 and Section A.2. Since the baseline doesn't use the semantic nature of text, there is a related approach based on WordNet(wordPOV) [Li et al., 2006, Pedersen et al., 2004] which looks at text similarity, explained prior in Section 3.2.2. The underlying assumption is that pairs of sentences that are known to be related in some way if they exhibit on average much greater similarity than unrelated pairs,

Table 4.1: Dataset

Dataset	**Attributes**
	Data 1
Topic	Racism in United States
No. of Articles	28
No. of Quotations obtained	51
Source	NY Times, TOI, Telegraph, NDTV, Reuters ...
No. of Classes (Human Evaluator)	17
	Data 2
Topic	Wimbledon Federer Murray
No. of Articles	10
No. of Quotations obtained	66
Source	NY Times, TOI, Telegraph, NDTV, Reuters ...
No. of Classes (Human Evaluator)	37

which we define by some threshold based on the data. We evaluate the effectiveness of several similarity measures by calculating how well they are able to distinguish similar quotations from unrelated ones. We identified the value of K for K-means clustering by using the elbow method. But again as the quote goes "clusters are in the eye of the beholder!". Based on the description given in Section 3.2.3, we determined the value of K for clustering. Next section we compare the results of different approach against our method ourPOV.

4.2 Evaluation

We compare four different approaches in this to verify and validate the performance. The ground truth table was prepared manually by human evaluators who did not know the outcome of the clustering obtained by different methods. The evaluators marked each quotation with a *class id* (such as $A, B, ...$) . Next the goal was attaining high intra-cluster similarity (documents within a cluster are similar) and low inter-cluster similarity (documents from different clusters are dissimilar). This is an internal criterion for the quality of a clustering. Purity is an external evaluation criterion of cluster quality. It is the percent of the total number of objects(data points) that were classified correctly, in the unit range [0..1] [1]. To compute purity, each cluster is assigned to the class which is most frequent in the cluster, and then the accuracy of this assignment is measured by counting the number of correctly assigned documents and dividing by N.

Figure 4.1 and 4.2 show the purity vs k plots obtained by the four methods. As we see, our method outperforms the baselines for most values of k. In Table 4.2 we also show the average purity of all the methods (average taken over all k) and we find that our method performs better than the baselines in average.

[1] http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html

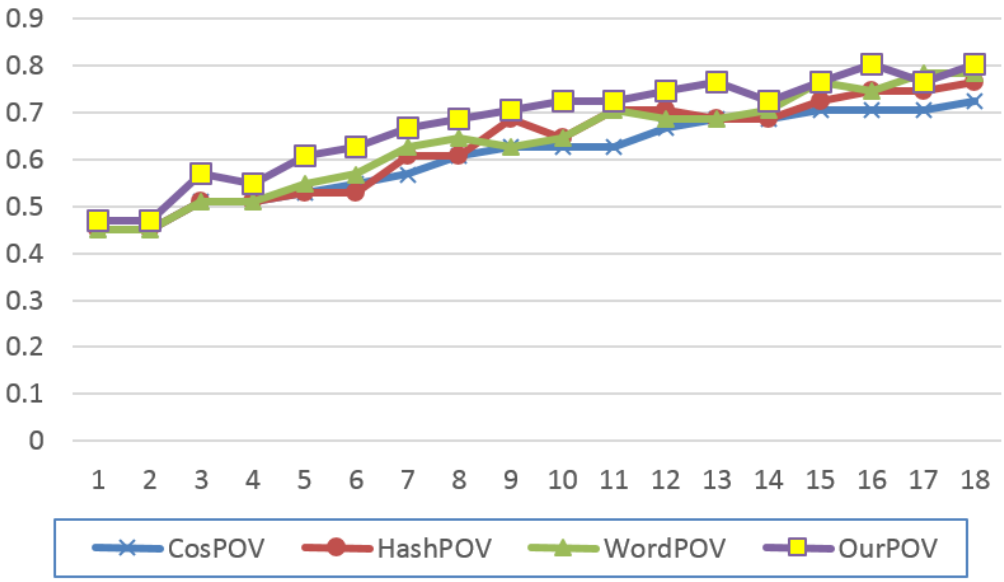

Figure 4.1: Purity vs K plot for dataset 1

$$Purity = \frac{1}{N}\sum_{i=1}^{k} max_j |c_i \cap t_j|$$

where N = number of objects(data points), k = number of clusters, c_i is a cluster in C, and t_j is the classification which has the max count for cluster c_i

Table 4.2: Evaluation Comparison

Method	**Dataset1**	**Dataset2**
Cosine Similarity	0.61	0.42
Min-Hash Similarity	0.62	0.46
Semantic Similarity Word-net	0.64	0.44
Our Approach	**0.68**	**0.47**

In order to assess the validity of this approach a more rigorous evaluation, in the context of existing news aggregation systems, is required. Surprisingly little work has been undertaken to understand the role of automated information retrieval processes for new aggregation in terms of usability and

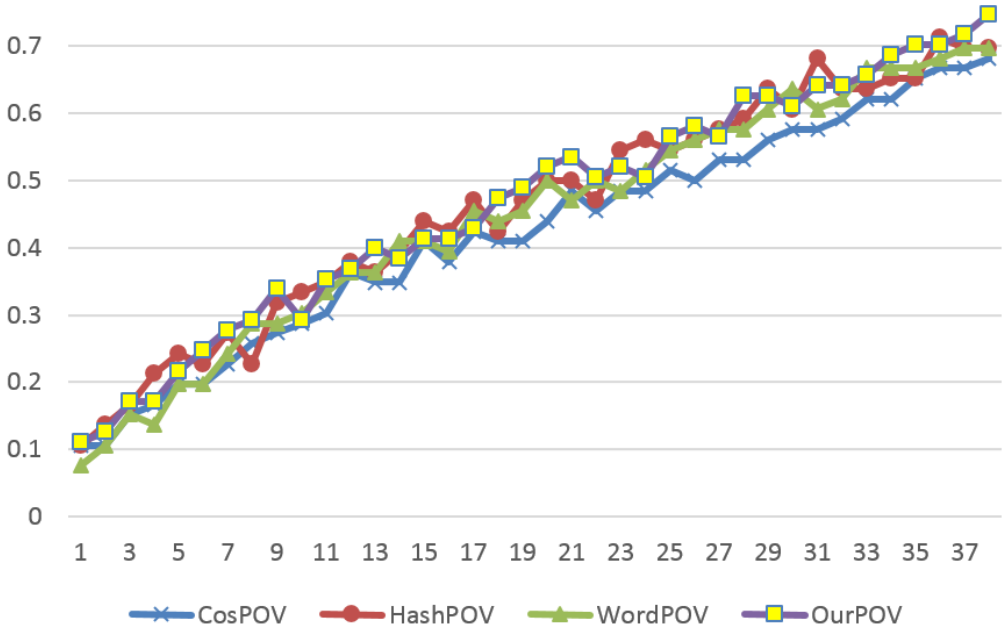

Figure 4.2: Purity vs K plot for dataset 2

impact on user performance. Those existing studies are based on document summarization approaches [Barzilay and McKeown, 2005]. Typically these are assessed using a quantitative method which compares generated summaries to some hand crafted gold standard summary or other user-independent approaches [Bogers and Van den Bosch, 2007]. However, since our approach is entirely unsupervised, this form of assessment often struggles to examine the generalisability of an approach. A preferred alternative is to use an information retrieval task, completed by human participants, where performance is measured by how quickly and accurately users retrieve some required information [Jing et al., 1998].

Chapter 5

Conclusion

In this thesis, we have discussed the goals for providing a new concept of identifying the various point of views in different newspaper articles on the same topic. On that basis, we have formally defined a family of new measures of short sentence similarity. We described an unsupervised syntactic approach for learning of Point of Views(POVs) from news articles. It uses novel and efficient algorithms for syntactic pre-processing to finally feeding us the POVs. Major Steps involved were collecting articles on the same topic from multiple newspapers. Then extract the quotations, and identifying the source and target entities. Similarities between two quotation is checked and if above a threshold, they are put in the same clusters. Finally we get a number of cluster based on the data, and we call each a POV. A user or audience can have all the information going through those POVs.

Future Work

There are many interesting directions that can be explored.

- **Visualization of Points of view**: Moving from a Document centric views of news, to a Points of view centric view. The task is to summarize each cluster

of Points of view, and present the summary to the user, along with pointers to original articles.

- **Scalability**: Ongoing work is continuing to improve the scalability . Currently the algorithm performs sequentially but there are several parts in the codes which can be parallelized. Defining new thresholding and tuning techniques are again expected to improve the performance of the code and effectiveness of the proposed algorithm.

- The next step for this work is to undertake a comparative user study against state-of-the-art news article summarisation and ranking approaches.

- Choosing the optimal number of clusters automatically. Since on query we have a different kind of dataset, some of them which we have no prior domain knowledge of. Hence determination of right K automatically is important, though it being a tough problem.

- The study will be task-based requiring users to answer questions related to facts presented within a set of news articles. We will use a post task questionnaire to produce satisfaction ratings and assess the perceived success in providing an overview of perspectives on a topic.

- Two of the most promising lines of investigation in NLP motivated by our application are quote-aware co-reference resolution for speakers and subjects to improve quote recall, and more sophisticated similarity metrics for clustering quotes into threads.

From our preliminary study it is clear that there are associated issues with generating perspectives related to the subjective nature of perspective. We should evaluate the ability of our proposed system to truly represent the various viewpoints expressed in a set of articles and how users perceive bias in the presented point of views. It is

difficult to define an objective measure which can be calculated automatically. Therefore any evaluation is likely to be subjective. For example, one approach might be to compare the various point of views produced by the system against those selected from the same set of articles by users. The robustness of the approach should also be tested to ensure that the system works well in a wide variety of cases.

Appendix A

Pre-requisites

A.1 Opinion Mining & Sentiment Analysis

What other people think is naturally important for human guidance. Through opinions, humans can flux together diverse approaches, experiences, wisdom and knowledge of people for decision making. Humans like to take part in discussions and present their points of view. The point of view about something can either be positive (shows goodness) or negative (shows badness), which is called the polarity of the opinion. An opinion has three main components i.e. the opinion holder or source of opinion, the object about which the opinion is expressed and the evaluation, view or appraisal which is called the opinion. For opinion identification, all these components are important.

They use opinions to express their points of view based on experience, observation, concept, beliefs, and perceptions. The point of view about something can either be positive (shows goodness) or negative (shows badness), which is called the polarity of the opinion. Opinion target identification is basically a classification problem which is defined as: to classify noun phrase or term as opinion target or not.

Sentiment analysis of natural language texts is a large and growing field. We define sentiment to be "a personal positive or negative feeling." Previous work particularly relevant to our task falls naturally in two groups. The first relates to techniques to automatically generate sentiment lexicons. The second relates to systems that analyze sentiment (on a global or local basis) for entire documents.

Several systems have been built which attempt to quantify opinion from product reviews. [Pang and Lee, 2008] perform sentiment analysis of movie reviews.This work has served as a baseline and many authors have used the techniques provided in their paper across different domains. Some related work on determining sentiments on short texts or tweets is [Go et al., 2009]. Stanford Sentiment Analysis tool [Socher et al., 2013] is used as it is capable of computing document-level sentiment, certain other Sentiment toolkit like Alchemy API is also used for user-specified sentiment targeting, entity-level sentiment, emoticons and keyword-level sentiment.

A.2 Detecting Similar Text

The problem of computing similarity, is the principal basis of clustering. The most substantive decision we need to make in clustering a set of strings or in our case list of quotations, is which underlying similarity metric to use. There are myriad string similarity metrics available, and choosing the one that's most appropriate for your situation largely depends on the nature of your objective.

Although these similarity measurements are not difficult to define and compute ourselves, I'll take this opportunity to introduce NLTK (the Natural Language Toolkit), a Python toolkit that you'll be glad to have in your arsenal for mining the social web.

Here are a few of the common similarity metrics that might be helpful in comparing texts that are implemented in NLTK:

Edit distance

Edit distance, also known as Levenshtein distance, is a simple measure of how many insertions, deletions, and replacements it would take to convert one string into another. For example, the cost of converting dad into bad would be one replacement operation (substituting the first d with a b) and would yield a value of '1'. NLTK provides an implementation of edit distance via the nltk.metrics.distance.edit_distance function. Using Levenshtein's original operations, the edit distance between $a = a_1 \dots a_n$ and $b = b_1 \dots b_m$ is given by d_{mn}, defined by the recurrence

$$d_{i0} = \sum_{k=1}^{i} w_{\text{del}}(b_k), \qquad \text{for } 1 \leq i \leq m \tag{A.1}$$

$$d_{0j} = \sum_{k=1}^{j} w_{\text{ins}}(a_k), \qquad \text{for } 1 \leq j \leq n \tag{A.2}$$

$$d_{ij} = \begin{cases} d_{i-1,j-1} & \text{for } a_j = b_i \\ \min \begin{cases} d_{i-1,j} + w_{\text{del}}(b_i) \\ d_{i,j-1} + w_{\text{ins}}(a_j) & \text{for } a_j \neq b_i \\ d_{i-1,j-1} + w_{\text{sub}}(a_j, b_i) \end{cases} \end{cases} \qquad \text{for } 1 \leq i \leq m, 1 \leq j \leq n. \tag{A.3}$$

The actual edit distance between two strings is quite different from the number of operations required to compute the edit distance; computation of edit distance is usually on the order of M*N operations for strings of length M and N. In other words, computing edit distance can be a computationally intense operation, so use it wisely on nontrivial amounts of data.

N-gram similarity

An n-gram is just a terse way of expressing each possible consecutive sequence of n tokens from a text, and it provides the foundational data structure for computing collocations. There are many variations of n-gram similarity, but consider the straightforward case of computing all possible bigrams (two-grams) for the tokens of two strings, and scoring the similarity between the strings by counting the number of common bigrams between them. The measures are defined as the ratio of the number of n-grams that are shared by two strings and the total number of n-grams in both strings :

$$\frac{2|n-grams(X) \bigcup n-grams(Y)|}{|n-grams(X)| + |n-grams(Y)|} \tag{A.4}$$

NLTK provides a fairly comprehensive array of bigram and trigram (three-gram) scoring functions via the BigramAssociationMeasures and TrigramAssociationMeasures classes defined in its nltk.metrics.association module.

Jaccard Similarity

More often than not, similarity can be computed between two sets of things, where a set is just an unordered collection of items. The Jaccard similarity metric expresses the similarity of two sets and is defined by the intersection of the sets divided by the union of the sets. Mathematically, the Jaccard similarity is written as:

$$J(A,B) = \frac{|A \bigcap B|}{|A \bigcup B|} \tag{A.5}$$

(If A and B are both empty, we define J(A,B) = 1.) $0 \leq J(A,B) \leq 1$. which is the number of items in common between the two sets (the cardinality of their set intersection) divided by the total number of distinct items in the two sets (the

cardinality of their union). The intuition behind this ratio is that calculating the number of unique items that are common to both sets divided by the number of total unique items is a reasonable way to derive a normalized score for computing similarity. In general, you'll compute Jaccard similarity by using n-grams, including unigrams (single tokens), to measure the similarity of two strings.

The **MinHash** min-wise independent permutations locality sensitive hashing scheme may be used to efficiently compute an accurate estimate of the Jaccard similarity coefficient of pairs of sets, where each set is represented by a constant-sized signature derived from the minimum values of a hash function.

The Jaccard distance, which measures dissimilarity between sample sets, is complementary to the Jaccard coefficient and is obtained by subtracting the Jaccard coefficient from 1, or, equivalently, by dividing the difference of the sizes of the union and the intersection of two sets by the size of the union:

$$d_J(A, B) = 1 - J(A, B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}. \tag{A.6}$$

Vector Space Models and Cosine Similarity

While it has been emphasized that TF-IDF (Term Frequency and Inverse Document Frequency) models documents as unordered collections of words, another convenient way to model documents is with a model called a vector space. The basic theory behind a vector space model is that you have a large multidimensional space that contains one vector for each document, and the distance between any two vectors indicates the similarity of the corresponding documents.

$$\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \cos\theta \tag{A.7}$$

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}} \tag{A.8}$$

Given that it's possible to model documents as term-centric vectors, with each term in the document represented by its corresponding TF-IDF score, the task is to determine what metric best represents the similarity between two documents. As it turns out, the cosine of the angle between any two vectors is a valid metric for comparing them and is known as the cosine similarity of the vector. Intuitively, it might be helpful to consider that the closer two vectors are to one another, the smaller the angle between them will be, and thus the larger the cosine of the angle between them will be. Two identical vectors would have an angle of 0 degrees and a similarity metric of 1.0, while two vectors that are orthogonal to one another would have an angle of 90 degrees and a similarity metric of 0.0.

A.3 Clustering Algorithms

Greedy Clustering

If the distance between any two sentences as determined by a similarity heuristic is "close enough", set by some threshold based on data, we greedily group them together. In this context, being "greedy" means that the first time we are able to determine that an item might fit in a cluster, we go ahead and assign it without further considering whether there might be a better fit, or making any attempt to account for such a better fit if one appears later. Although incredibly pragmatic, this approach produces very reasonable results. Clearly, the choice of an effective similarity heuristic is critical to its success, but given the nature of the nested loop, the fewer times we have to invoke the scoring function, the faster the code executes (a principal concern for nontrivial sets of data).

A nested loop that compares every single item to every single other item for clustering purposes is not a scalable approach for a very large value of n. The crux of an $\mathcal{O}(n^2)$ algorithm is that the number of comparisons required to process an input set increases exponentially in proportion to the number of items in the set.

Hierarchical clustering

Hierarchical clustering is superficially similar to the greedy heuristic we have been using, while k-means clustering is radically different. We'll primarily focus on k-means for rest of this section, but it's worthwhile to briefly introduce the theory behind both of these approaches since you're very likely to encounter them during literature review and research. Hierarchical clustering is a deterministic technique in that it computes the full matrix of distances between all items and then walks through the matrix clustering items that meet a minimum distance threshold. It's hierarchical in that walking over the matrix and clustering items together produces a tree structure that expresses the relative distances between items. In the literature, you may see this technique called agglomerative because it constructs a tree by arranging individual data items into clusters, which hierarchically merge into other clusters until the entire data set is clustered at the top of the tree. The leaf nodes on the tree represent the data items that are being clustered, while intermediate nodes in the tree hierarchically agglomerate these items into clusters. The computation of a full matrix implies a polynomial runtime. For agglomerative clustering, the runtime is often on the order of $\mathcal{O}(n^3)$. But clever use of dynamic programming and memoization is used to reduce the complexity to $\mathcal{O}(n^2)$.

k-means clustering

Whereas hierarchical clustering is a deterministic technique that exhausts the possibilities and is often an expensive computation on the order of $\mathcal{O}(n^3)$, k-means clustering

generally executes on the order of $\mathcal{O}(k * n)$ times. For even small values of k, the savings are substantial. The savings in performance come at the expense of results that are approximate, but they still have the potential to be quite good.

Given a set of observations $(x1, x2, ..., xn)$, where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into $k(\leq n)$ sets $S = S1, S2, ..., Sk$ so as to minimize the within-cluster sum of squares (WCSS). In other words, its objective is to find:

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 \tag{A.9}$$

where i is the mean of points in Si.

The idea is that you generally have a multidimensional space containing n points, which you cluster into k clusters through the following series of steps:

1. Randomly pick k points in the data space as initial values that will be used to compute the k clusters: K1, K2,..., Kk.

2. Assign each of the n points to a cluster by finding the nearest Kneffectively creating k clusters and requiring k*n comparisons.

3. For each of the k clusters, calculate the centroid, or the mean of the cluster, and reassign its Ki value to be that value. (Hence, youre computing "k-means"during each iteration of the algorithm.)

4. Repeat steps 2-3 until the members of the clusters do not change between iterations. Generally speaking, relatively few iterations are required for convergence.

Elbow Method

Clustering consist of grouping objects in sets, such that objects within a cluster are as similar as possible, whereas objects from different clusters are as dissimilar

as possible. Thus, the optimal clustering is somehow subjective and dependent on the characteristic used for determining similarities, as well as on the level of detail required from the partitions. For the purpose of our clustering experiment we use clusters derived from Gaussian distributions, i.e. globular in nature, and look only at the usual definition of Euclidean distance between points in a two-dimensional space to determine intra and inter-cluster similarity. The following measure represents the sum of intra-cluster distances between points in a given cluster C_k containing n_k points:

$$D_k = \sum_{\mathrm{x}_i \in C_k} \sum_{\mathrm{x}_j \in C_k} ||\mathrm{x}_i - \mathrm{x}_j||^2 = 2n_k \sum_{\mathrm{x}_i \in C_k} ||\mathrm{x}_i - \mu_k||^2.$$

Adding the normalized intra-cluster sums of squares gives a measure of the compactness of our clustering:

$$W_k = \sum_{k=1}^{K} \frac{1}{2n_k} D_k.$$

This variance quantity W_k is the basis of a naive procedure to determine the optimal number of clusters: the elbow method. The gap statistic was developed by Stanford researchers Tibshirani, Walther and Hastie in their 2001 paper. The idea behind their approach was to find a way to standardize the comparison of $\log W_k$ with a null reference distribution of the data, i.e. a distribution with no obvious clustering. Their estimate for the optimal number of clusters K is the value for which $\log W_k$ falls the farthest below this reference curve. This information is contained in the following formula for the gap statistic:

$$\mathrm{Gap}_n(k) = E_n^*\{\log W_k\} - \log W_k.$$

To obtain the estimate $E_n^*\{\log W_k\}$ we compute the average of B copies $\log W_k^*$ for $B = 10$, each of which is generated with a Monte Carlo sample from the reference distribution. Those $\log W_k^*$ from the B Monte Carlo replicates exhibit a standard deviation $\mathrm{sd}(k)$ which, accounting for the simulation error, is turned into the quantity

$$s_k = \sqrt{1 + 1/B}\,\mathrm{sd}(k).$$

Finally, the optimal number of clusters K is the smallest k such that $\mathrm{Gap}(k) \geq \mathrm{Gap}(k+1) - s_{k+1}$.

Metric

Euclidean Distance :	$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
Squared Euclidean distance :	$\|a-b\|_2^2 = \sum_i (a_i - b_i)^2$
Manhattan distance :	$\|a-b\|_1 = \sum_i \|a_i - b_i\|$
maximum distance :	$\|a-b\|_\infty = \max_i \|a_i - b_i\|$
Mahalanobis distance :	$\sqrt{(a-b)^\top S^{-1}(a-b)}$ $\quad S : Covariance\,Matrix$

Linkage criteria

Maximum or complete linkage clustering :	$\max\{\, d(a,b) : a \in A,\, b \in B \,\}$.
Minimum or single-linkage clustering :	$\min\{\, d(a,b) : a \in A,\, b \in B \,\}$.
Mean or average linkage clustering :	$\frac{1}{\|A\|\|B\|} \sum_{a \in A} \sum_{b \in B} d(a,b)$.
Centroid linkage clustering :	$\|\|c_s - c_t\|\|$ $\quad c_s\, and\, c_t\, are\, cluster\, centroid.$

A.4 Document Summarization

Document summarization creates information reports that are both concise and comprehensive. With different opinions being put together & outlined, every topic is described from multiple perspectives within a single document. While the goal of a brief summary is to simplify information search and cut the time by pointing to the most relevant source documents, comprehensive multi-document summary should itself contain the required information, hence limiting the need for accessing original files to cases when refinement is required. Automatic summaries present information extracted from multiple sources algorithmically, without any editorial touch or subjective human intervention, thus making it completely unbiased.There are numerous possibilities and approaches, but one of the simplest to get started with dates all the way back to the April 1958 issue of IBM Journal. In the seminal article entitled "The Automatic Creation of Literature Abstracts," H.P. Luhn (See [Luhn, 1958] .) The basic premise behind Luhns algorithm is that the important sentences in a document will be the ones that contain frequently occurring words. First, not all frequently occurring words are important; generally speaking, stopwords are filler and are hardly ever of interest for analysis. The first approach uses a statistical threshold to filter out sentences by computing the mean and standard deviation for the scores obtained, while the latter simply returns the top N sentences. Depending on the nature of the data, your mileage will vary, but you should be able to tune the parameters to achieve reasonable results with either.

Appendix B

Required Tools

B.1 NLTK

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, and an active discussion forum `http://www.nltk.org`. It is an Extensible Toolkit for Computational Semantics.

B.2 NY Times API

NYTimes.com is now not only an unparalleled source of news and information. But now it's a premier source of data. With the Article Search API, you can search New York Times articles from Sept. 18, 1851 to today, retrieving headlines, abstracts, lead paragraphs, links to associated multimedia and other article metadata. API monitors functionality which includes breaking news, categorisation of news according to predefined categories or user-defined search words, linking of related news over time, extraction and display of meta-information such as references to locations, persons

and organisations, or quotations.The link is `http://developer.nytimes.com/docs/read/article_search_api_v2`. You can get useful results from the Article Search API with a simple query.

B.3 Stanford Log-linear Part-Of-Speech Tagger

A Part-Of-Speech Tagger (POS Tagger) is a piece of software that reads text in some language and assigns parts of speech to each word (and other token), such as noun, verb, adjective, etc., although generally computational applications use more fine-grained POS tags like 'noun-plural' [Toutanova et al., 2003].This assigns part-of-speech (POS) information to each token. For example, 'NNP' indicates that the token is a noun that is part of a noun phrase, 'VBD' indicates a verb thats in simple past tense, and 'JJ' indicates an adjective. The Penn Treebank Project provides a full summary of the POS tags that could be returned. With POS tagging completed, it should be getting pretty apparent just how powerful analysis can become. For example, by using the POS tags, well be able to chunk together nouns as part of noun phrases and then try to reason about what types of entities they might be (e.g., people, places, or organizations).

B.4 Stanford Named Entity Recognizer (NER)

Named Entity Recognition (NER) labels sequences of words in a text which are the names of things, such as person and company names, or gene and protein names. It comes with well-engineered feature extractors for Named Entity Recognition, and many options for defining feature extractors. Stanford NER is also known as CRF-Classifier. The software provides a general implementation of (arbitrary order) linear chain Conditional Random Field (CRF) sequence models [Finkel et al., 2005]

B.5 Stanford Sentiment Analysis

Stanford Sentiment Analysis tool is based on a new type of Recursive Neural Network that builds on top of grammatical structures. Its new deep learning model actually builds up a representation of whole sentences based on the sentence structure. It computes the sentiment based on how words compose the meaning of longer phrases [Socher et al., 2013].

B.6 Alchemy API

AlchemyAPI uses natural language processing technology and machine learning algorithms to extract semantic meta-data from content, such as information on people, places, companies, topics, facts, relationships, authors, and languages. API endpoints are provided for performing content analysis on Internet-accessible web pages, posted HTML or text content. To use AlchemyAPI, you need an access key here `http://www.alchemyapi.com/api`. It has several functionalities like Entity extraction, Sentiment analysis, Keyword extraction, Concept tagging, Relation extraction, Taxonomy Classification, Language detection, Text extraction.

B.7 TextRazor

TextRazor offers a complete cloud or self-hosted text analysis infrastructure. Also involves creating a key here `https://www.textrazor.com`. Properties include Entity Extraction, Disambiguation and Linking,Keyphrase Extraction, Automatic Topic Tagging and Classification. Deep analysis of your content to extract Relations, Typed Dependencies between words and Synonyms, enabling powerful context aware semantic applications.

Bibliography

[Baeza-Yates et al., 1999] Baeza-Yates, R., Ribeiro-Neto, B., et al. (1999). *Modern information retrieval*, volume 463. ACM press New York.

[Balahur and Steinberger, 2009] Balahur, A. and Steinberger, R. (2009). Rethinking sentiment analysis in the news: from theory to practice and back. *Proceeding of WOMSA*, 9.

[Balahur et al., 2009] Balahur, A., Steinberger, R., Goot, E. v. d., Pouliquen, B., and Kabadjov, M. (2009). Opinion mining on newspaper quotations. In *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT'09. IEEE/WIC/ACM International Joint Conferences on*, volume 3, pages 523–526. IET.

[Barzilay and Lee, 2003] Barzilay, R. and Lee, L. (2003). Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 16–23. Association for Computational Linguistics.

[Barzilay and McKeown, 2005] Barzilay, R. and McKeown, K. R. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

[Bendersky and Croft, 2009] Bendersky, M. and Croft, W. B. (2009). Finding text reuse on the web. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 262–271. ACM.

[Bogers and Van den Bosch, 2007] Bogers, T. and Van den Bosch, A. (2007). Comparing and evaluating information retrieval algorithms for news recommendation. In *Proceedings of the 2007 ACM conference on Recommender systems*, pages 141–144. ACM.

[Chen et al., 2002] Chen, G., Jaradat, S. A., Banerjee, N., Tanaka, T. S., Ko, M. S., and Zhang, M. Q. (2002). Evaluation and comparison of clustering algorithms in analyzing es cell gene expression data. *Statistica Sinica*, 12(1):241–262.

[Das and Smith, 2009] Das, D. and Smith, N. A. (2009). Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference*

on Natural Language Processing of the AFNLP: Volume 1-Volume 1, pages 468–476. Association for Computational Linguistics.

[Feng et al., 2008] Feng, J., Zhou, Y.-M., and Martin, T. (2008). Sentence similarity based on relevance. In *Proceedings of IPMU*, volume 8, pages 832–839.

[Fernando and Stevenson, 2008] Fernando, S. and Stevenson, M. (2008). A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics*, pages 45–52. Citeseer.

[Finkel et al., 2005] Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics.

[Go et al., 2009] Go, A., Bhayani, R., and Huang, L. (2009). Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1:12.

[Godbole et al., 2007] Godbole, N., Srinivasaiah, M., and Skiena, S. (2007). Large-scale sentiment analysis for news and blogs. *ICWSM*, 7:21.

[Harb et al., 2008] Harb, A., Plantié, M., Dray, G., Roche, M., Trousset, F., and Poncelet, P. (2008). Web opinion mining: How to extract opinions from blogs? In *Proceedings of the 5th international conference on Soft computing as transdisciplinary science and technology*, pages 211–217. ACM.

[Ho et al., 2010] Ho, C., Murad, M. A. A., Kadir, R. A., and Doraisamy, S. C. (2010). Word sense disambiguation-based sentence similarity. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 418–426. Association for Computational Linguistics.

[Jing et al., 1998] Jing, H., Barzilay, R., McKeown, K., and Elhadad, M. (1998). Summarization evaluation methods: Experiments and analysis. In *AAAI symposium on intelligent summarization*, pages 51–59.

[Kaji and Kitsuregawa, 2007] Kaji, N. and Kitsuregawa, M. (2007). Building lexicon for sentiment analysis from massive collection of html documents. In *EMNLP-CoNLL*, pages 1075–1083.

[Kobayashi et al., 2005] Kobayashi, N., Inui, K., Matsumoto, Y., Tateishi, K., and Fukushima, T. (2005). Collecting evaluative expressions for opinion extraction. In *Natural Language Processing–IJCNLP 2004*, pages 596–605. Springer.

[Li et al., 2006] Li, Y., McLean, D., Bandar, Z., O'shea, J. D., Crockett, K., et al. (2006). Sentence similarity based on semantic nets and corpus statistics. *Knowledge and Data Engineering, IEEE Transactions on*, 18(8):1138–1150.

[Lin, 2004] Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8.

[Liu, 2010] Liu, B. (2010). Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2:627–666.

[Luhn, 1958] Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165.

[MacCartney and Manning, 2007] MacCartney, B. and Manning, C. D. (2007). Natural logic for textual inference. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 193–200. Association for Computational Linguistics.

[Manku et al., 2007] Manku, G. S., Jain, A., and Das Sarma, A. (2007). Detecting near-duplicates for web crawling. In *Proceedings of the 16th international conference on World Wide Web*, pages 141–150. ACM.

[Mohler and Mihalcea, 2009] Mohler, M. and Mihalcea, R. (2009). Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 567–575. Association for Computational Linguistics.

[Pang and Lee, 2008] Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135.

[Papineni et al., 2002] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

[Park et al., 2011] Park, S., Ko, M., Kim, J., Liu, Y., and Song, J. (2011). The politics of comments: predicting political orientation of news stories with commenters' sentiment patterns. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, pages 113–122. ACM.

[Patwardhan and Pedersen, 2006] Patwardhan, S. and Pedersen, T. (2006). Using wordnet-based context vectors to estimate the semantic relatedness of concepts. In *Proceedings of the EACL 2006 Workshop Making Sense of Sense-Bringing Computational Linguistics and Psycholinguistics Together*, volume 1501, pages 1–8. Citeseer.

[Pedersen et al., 2004] Pedersen, T., Patwardhan, S., and Michelizzi, J. (2004). Wordnet:: Similarity: measuring the relatedness of concepts. In *Demonstration papers at hlt-naacl 2004*, pages 38–41. Association for Computational Linguistics.

[Pulman and Sukkarieh, 2005] Pulman, S. G. and Sukkarieh, J. Z. (2005). Automatic short answer marking. In *Proceedings of the second workshop on Building Educational Applications Using NLP*, pages 9–16. Association for Computational Linguistics.

[Qu et al., 2004] Qu, Y., Shanahan, J., and anyce Wiebe (2004). *Exploring Attitude and Affect in Text, Theories and Applications: Papers from the 2004 AAAI Symposium: March 22-24, Stanford, California*. AAAI Press.

[Resnik, 1998] Resnik, P. (1998). Wordnet and class-based probabilities. *WordNet: An electronic lexical database*, pages 239–263.

[Rousseeuw, 1987] Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

[Seo and Croft, 2008] Seo, J. and Croft, W. B. (2008). Local text reuse detection. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 571–578. ACM.

[Singhal, 2001] Singhal, A. (2001). Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, 24(4):35–43.

[Socher et al., 2013] Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer.

[Tanev, 2007] Tanev, H. (2007). Unsupervised learning of social networks from a multiple-source news corpus. *MuLTISOuRcE, MuLTILINguAL INfORMATION ExTRAc-TION ANd SuMMARIzATION*, page 33.

[Tibshirani et al., 2001] Tibshirani, R., Walther, G., and Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423.

[Tintarev and Masthoff, 2006] Tintarev, N. and Masthoff, J. (2006). Similarity for news recommender systems. In *Proceedings of the AH06 Workshop on Recommender Systems and Intelligent User Interfaces*. Citeseer.

[Toutanova et al., 2003] Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics.

[Wiebe, 1994] Wiebe, J. M. (1994). Tracking point of view in narrative. *Computational Linguistics*, 20(2):233–287.

[Wilson et al., 2009] Wilson, T., Wiebe, J., and Hoffmann, P. (2009). Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational linguistics*, 35(3):399–433.

[Wu and Palmer, 1994] Wu, Z. and Palmer, M. (1994). Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics.

[Zanzotto et al., 2009] Zanzotto, F., Pennacchiotti, M., and Moschitti, A. (2009). A machine learning approach to textual entailment recognition. *Natural Language Engineering*, 15(04):551–582.