



(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Sentiment Analysis about Restaurant Reviews

A Report for the Evaluation 3 of Project 2

Submitted by

POOJA TYAGI

(1613101483 / 16SCSE101670)

in partial fulfilment for the award of the degree

of

Bachelor of Technology

IN

COMPUTER SCIENCE

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Dr. Kavita

APRIL / MAY- 2020

Table of Content:

CHAPTERS:

1. Introduction
2. Literature Survey
3. Approach
4. Implementation
5. Result and analysis
6. Conclusion and future work
7. References

CHAPTER 1

INTRODUCTION

1.1 Background

Artificial intelligence is introduced in the last few years, it is going to change our lives. So, if you've been keeping up with the latest technology trends, then you know that artificial intelligence has the potential to be the most disruptive technology ever. Today, we can ask Siri or Google or Cortana or Alexa to help us with simple questions or tasks, but much of their actual potential is still untapped. The reason why involves language.

This is where natural language processing (NLP) comes into play in artificial intelligence applications. Without NLP, artificial intelligence only can understand the meaning of language and answer simple questions, but it is not able to understand the meaning of words in context. Natural language processing applications allow users to communicate with a computer in their own worlds, i.e. in natural language.

Today, there are many examples of natural language processing systems in artificial intelligence already at work.

1.2. Levels of Natural Language Processing

Natural Language Processing (NLP) is a fundamental element of artificial intelligence for communicating with intelligent systems using natural language. NLP helps computers read and respond by simulating the human ability to understand the everyday language that people use to communicate. In fig.1.2.1 a broad view of classification of natural language processing is illustrated.

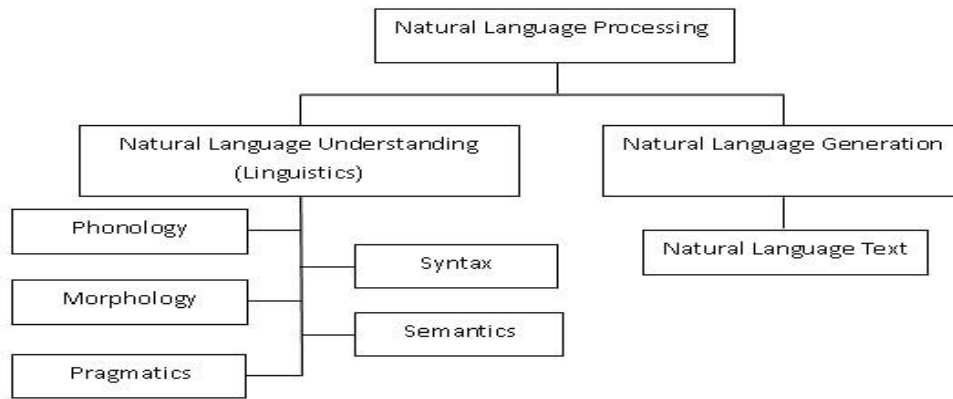


Fig. 1.1 Broad Classification of NLP

1.2.1 Natural Language understanding (Linguistics)

The various important terminologies of Natural Language Processing are: -

1.2.1.1 Phonology

Phonology is the part of Linguistics which refers to the systematic arrangement of sound. The term phono- which means voice or sound, and the suffix -logy refers to word or speech. Phonology is “the study of sound pertaining to the system of language”., whereas it could be explained as, "phonology proper is concerned with the function, behaviour and organization of sounds as linguistic items. Phonology include semantic use of sound to encode meaning of any Human language.

1.2.1.2. Morphology

The different parts of the word represent the smallest units of meaning known as Morphemes. Morphology which comprise of Nature of words, are initiated by morphemes. The interpretation of morpheme stays same across all the words, just to understand the meaning humans can break any unknown word into morphemes. Grammatical morphemes can be divided into bound morphemes and derivational morphemes.

1.2.1.3. Lexical

In Lexical, humans, as well as NLP systems, interpret the meaning of individual words. Sundry types of processing bestow to word-level understanding – the first of these being a part-of-speech tag to each word. In this processing, words that can act as more than one partof-speech are assigned the most probable part-of speech tag based on the context in which they occur. At the lexical level, Semantic representations can be replaced by the words that have one meaning. In NLP system, the nature of the representation varies according to the semantic theory deployed.

1.2.1.4. Syntactic

This level emphasizes to scrutinize the words in a sentence so as to uncover the grammatical structure of the sentence. Both grammar and parser are required in this level. The output of this level of processing is representation of the sentence that divulge the structural dependency relationships between the words. There are various grammars that can be implemented, and which in turn, whack the option of a parser. Not all NLP applications require a full parse of sentences, therefore the abide challenges in parsing of prepositional phrase attachment and conjunction audit no longer impede that plea for which phrasal and clausal dependencies are adequate.

1.2.1.5. Semantic

In semantic most people think that meaning is determined, however, this is not it is all the levels that bestow to meaning. Semantic processing determines the possible meanings of a sentence by pivoting on the interactions among word-level meanings in the sentence. This level of processing can incorporate the semantic disambiguation of words with multiple senses; in a cognate way to how syntactic disambiguation of words that can errand as multiple parts-of-speech is adroit at the syntactic level.

1.2.1.6. Discourse

While syntax and semantics travail with sentence-length units, the discourse level of NLP travail with units of text longer than a sentence i.e, it does not interpret multi sentence texts as just sequence sentences, apiece of which can be elucidated singly. Rather, discourse focuses on the properties of the text as a whole that convey meaning by making connections between component sentences. The two of the most common levels are *Anaphora Resolution* - Anaphora resolution is the replacing of words such as pronouns, which are semantically stranded, with the pertinent entity to which they refer.

1.2.1.7. Pragmatic

Pragmatic is concerned with the firm use of language in situations and utilizes nub over and above the nub of the text for understanding the goal and to explain how extra meaning is read into texts without literally being encoded in them. This requisite much world knowledge, including the understanding of intentions, plans, and goals.

1.2.2 Natural Language Generation

Natural Language Generation (NLG) is the process of producing phrases, sentences and paragraphs that are meaningful from an internal representation. It is a part of Natural Language Processing and happens in four phases: identifying the goals, planning on how goals maybe achieved by evaluating the situation and available communicative sources and realizing the plans as a text Fig.1.2.

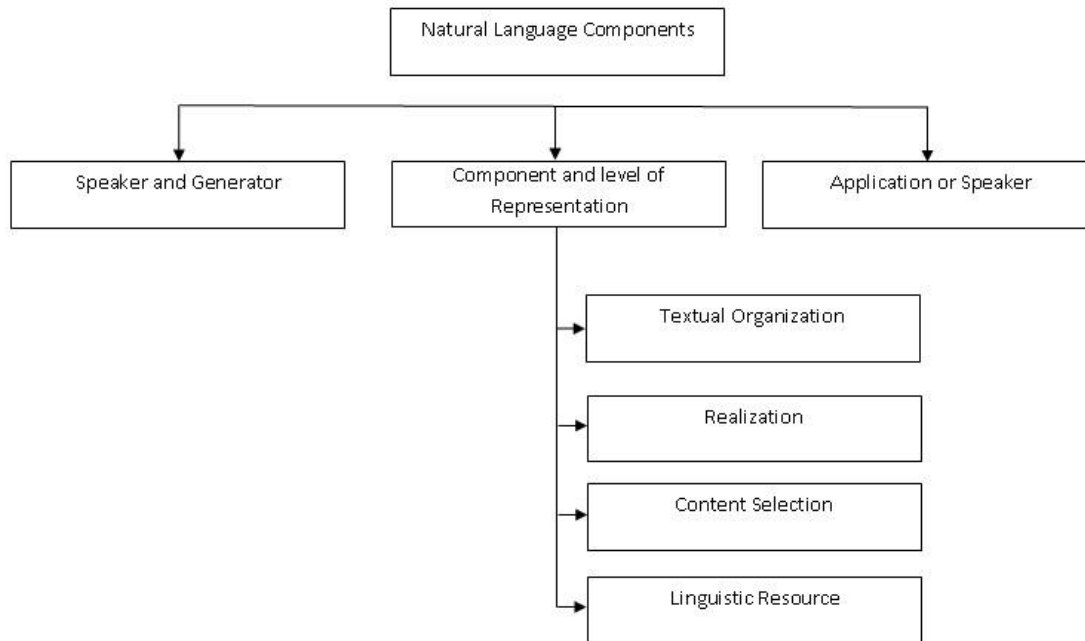


Fig. 1.2 Components of NLG

Components of NLG are as follows:

1.2.2.1. Speaker and Generator

To generate a text we need to have a speaker or an application and a generator or a program that renders the application's intentions into fluent phrase relevant to the situation.

1.2.2.2. Components and Levels of Representation

The process of language generation involves the following interweaved tasks. *Content selection*: Information should be selected and included in the set. Depending on how this information is parsed into representational units, parts of the units may have to be removed while some others may be added by default. *Textual Organization*: The information must be textually organized according the grammar, it must be ordered both sequentially and in terms of linguistic relations like modifications. *Linguistic Resources*: To support the information's realization, linguistic resources must be chosen. In the end these resources will come down to choices of particular words, idioms, syntactic constructs

etc. *Realization*: The selected and organized resources must be realized as an actual text or voice output.

1.2.2.3. Application or Speaker

This is only for maintaining the model of the situation. Here the speaker just initiates the process doesn't take part in the language generation. It stores the history, structures the content that is potentially relevant and deploys a representation of what it actually knows. All these form the situation, while selecting subset of propositions that speaker has. The only requirement is the speaker has to make sense of the situation.

1.2 Examples of natural language processing systems

1.2.1 Communication

Many communication applications such Facebook Messenger are already using artificial intelligence. On the whole, Facebook looks very interested in AI. Some months ago, Facebook announced its M service that promises to become your personal assistant (with the public launch date. "M can do anything a human can." When you request something that M can't do on its own, it sends a message to a Facebook worker and, as they work with the software, the AI begins to learn. Another interesting application of natural language processing is Skype Translator, which offers on-the-fly translation to interpret live speech in real time across a number of languages. Skype Translator uses AI to help facilitate conversation among people who speak different languages. This is great news! Without language barriers, people can communicate using the language they are comfortable with, which will in turn speed up a range of businesses processes.

1.2.2 Faster diagnosis

Natural language processing systems in artificial intelligence are also in hospitals that use natural language processing to indicate a specific diagnosis from a physician's unstructured notes. For example, NLP software for mammographic imaging and mammogram reports support the extraction and analysis of data for clinical decisions, as a study published in Cancer affirms. The software is able to determine breast cancer risk more efficiently, decrease the need for unnecessary biopsies and facilitate faster treatment through earlier diagnosis. According to the study, artificial intelligence reviewed 500 charts in but a few hours, saving over 500 physician hours.

1.2.3 Virtual digital assistants:

Virtual digital assistants: Thanks to smartphone, virtual digital assistant (VDA) technologies are currently the most well-known type of artificial intelligence. Many companies are

understanding the importance of the VDAs for their businesses and are investing significant resources to stay up to date.

If these simple examples of natural language processing applications in artificial intelligence are any indication, the next Artificial Intelligence software and applications will improve our ability to transform unstructured data into valuable business insight and make smart automated decision-making part of our everyday lives.

1.2.4 Customer Review

Natural language processing in artificial intelligence applications makes it easy to gather product reviews from a website and understand what consumers are actually saying as well as their sentiment in reference to a specific product. Companies with a large volume of reviews can actually understand them and use the data collected to recommend new products or services based on customer preferences. This application helps companies discover relevant information for their business, improve customer satisfaction, suggest more relevant products or services and better and understand the customer's needs.

1.3 Aspect Category

Aspect based category detection is necessary because many approaches are there to work on sentence, paragraph or document level. It can give the opinion about the document only. But on what basis the opinion is generated on the sentence, paragraph or on document, it never says that these are the term on that basis the opinion or sentiment analysis is positive or negative. Aspect category detection done this work.

The aspect category of a document cannot be a unique aspect of the same. Because a document or paragraph have multiple sentences, and each sentence can have multiple words, and more than one word can have a unique aspect, so for aspect category analysis, we need to study each and every word except stop word, my personal opinion about stop words is that we need to remove them for the betterment of machine.

Work is done in this paper is taken from the subtask of the SemEval-2014[3], the work is about to identify aspect category hidden in the text. There is a set of five aspect categories, which is set by the authorities. The data is taken from the website source, that is in the form of customer's review. The data need to classified into those given five aspect categories i.e., given the set of aspect categories (price, service, ambience, food, and anecdotes/miscellaneous).

Three example sentences are as follows.

1. ['Even though its good seafood, the prices are too high.'] →(Food and Price)
2. [' Add to that great service and great food at a reasonable price and you have yourself the beginning of a great evening.'] →(Service, Food and Anecdotes/Miscellaneous)
3. ['I have been coming to this neighbourhood spot for years.'] → (Anecdotes/Miscellaneous)

As shows in the above sentences, sentence 1 root word “food, seafood”, ”prices” directly defined the aspect category food and price, while in sentence 2 “Add to that great service and great food at a reasonable price” half of the review will show that there is food, service, Price available but in rest part of the review “you have yourself the beginning of a great evening.” shows that it is belong to default category, in sentence 3 there are not even explicit aspect available so it is also comes under default category.

1.4 Basic approach for sentiment analysis

In fig 1.4 Basic approach for sentiment analysis in natural language processing, in the textual data could be a sentence, a paragraph or it could be a document. There need to be two data sets, first one to train the machine and second one for test the trained machine. In pre-processing of reviews (train data), tokenization of sentence, spelling checks, stop words removal, and part of speech tagging. In pre-processing we need to find positive and negative expressions, incrementors and decrementors expressions(words)and inverters and polarity flips words. Once train data is pre-processed and the machine is trained using the train data, then test data will be test on the trained machine, where all the dictionaries are placed to find sentiment of the data. And then it will give according the machine is trained. It would be good if we use more data to train the machine the test data.

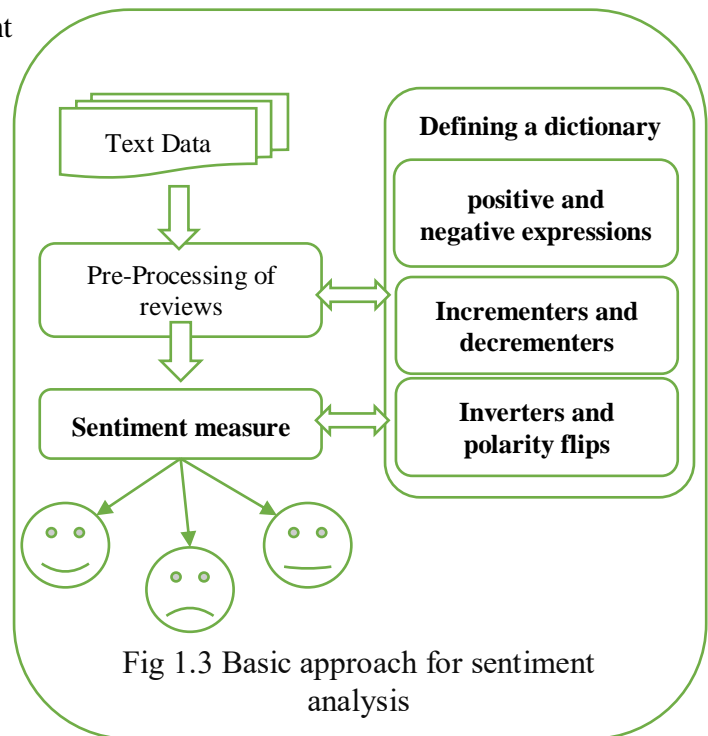


Fig 1.3 Basic approach for sentiment analysis

1.5 Motivation

As Natural language processing is used in many artificial intelligence applications, and it is a area where lots of work already done, and lots of word need to be done in future. Natural language processing can be beneficial for both consumer and business. The work given by SemEval2014[3], it is similar kind of work, where customers and owners of the restaurant can use our application to find their beneficial information. Earlier applications using unsupervised method are not able to give a usable system, which cannot be used to find right information. I found myself that I can contribute to this method in positive manner. Unsupervised method is usable, where the labelled data is not available to use supervised learning methods for classifying row data.

1.6 Thesis Outline

1.6.1 Chapter 1 – Introduction

In the chapter 1- A Background note on natural language processing and its applications. The chapter also give a brief explanation about the motivation of this work.

1.6.2 Chapter 2 – Literature Survey

Most recent work done on the Natural Language Processing and the result of each work.

1.6.3 Chapter 3 – Approach

Module of the proposed approach are explained in this chapter, details of data source, data collected from and data description, steps in the next module of the approach called pre-processing of data. The next module is about further steps taken to prepare a system, that will use various technique and algorithms. By creating occurrence and co-occurrence matrixes a digraph is created of all words available in the entire data. There are two algorithms are used Algorithm 1 is spreading activation to generate activation value for each word, and most close words to aspect category are added into the fire word list. Algorithm 2 is used as a association rule to get aspect category for each review.

1.6.4 Chapter 4 – Implementation

All modules of propose approach are explained by showcasing the real time input and output of the work.

1.6.5 Chapter 5 – Result and Analysis

Various techniques are used to evaluate the result of the system. Which technique is used to show accuracy of the system is explained. Graphical representation of the result data and graphical

representation of comparison of the labelled data and proposed approach result data. The result of the approach in terms of accuracy is 85.89%.

1.6.6 Chapter 6 – Conclusion and Future work

We introduced an unsupervised method for aspect category detection from the reviews of restaurant. It would be hard to get aspect category if the target aspects are not given. And the same word will need to be done on other language. Our future work will be on Hindi Language.

CHAPTER 2

LITERATURE SURVEY

Many researchers introduced supervised and unsupervised learning methods for detecting aspect category for sentiment analysis. In such field of research, aspect category detection for sentiment analysis that encouraged this paper is explained below.

2.1 Paper Titled: Supervised and unsupervised aspect category detection for sentiment analysis with co-occurrence Data

Authors: Kim Schouten, Onne van der Weijde, Flavius Frasinca, and Rommert Dekker

Publication: IEEE Transactions on Cybernetics, 2017

In this paper, the team used two methods to solve the problem as title is mentioned. One is Unsupervised method and Supervised method. The data is taken from SemEval2014 task. Data is text reviews about Laptops and Restaurants.

2.1.1 Unsupervised Methods

Data in the form of text reviews available in the xml format. The total number of reviews 3000. This method, introduced seed words to categorized the reviews into their aspects. Another technique to find closely related words to target aspect that is co-occurrence technique, how a word is strongly connected to word b? to get this information the team use spreading activation algorithms with co-occurrence data.

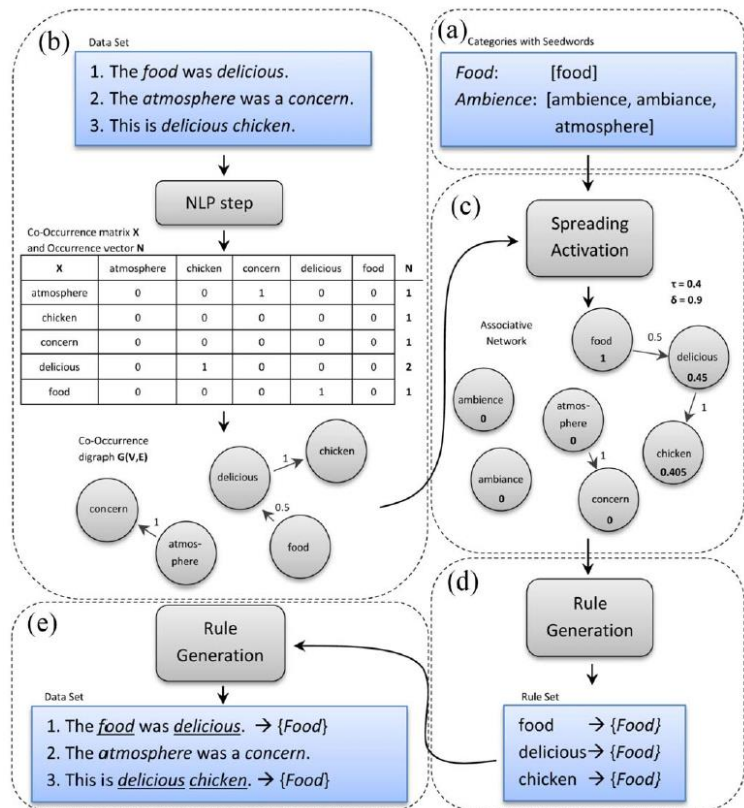


Fig. 2.1 Unsupervised method

In given unsupervised method, the approach is divided into five major steps, as shown in the fig 2.1, (a) is dictionary is created of seed words for aspect category, For example, the *ambience* category has the seed set {*ambience, ambience, atmosphere*}. (b) showing the data, the reviews. The reviews further

processed with NLP steps, like tokenization, removal of stop words and low frequency words. After that they have created co-occurrence matrix and occurrence matrix. The approach created a graph using both matrix, the edge weight = X_{ij}/N_j , where X_{ij} is occurrence value of X_j word after X_i Word and N_j is occurrence value of N_j word. (c) To get activation value for each vertex in the weighted graph created in previous step, is done using seed word dictionary and spreading activation algorithm. All the seed words available in the digraph will get its activation value = 1 and rest of the words get the value using $\min\{A_{c,j} + A_{c,i} \cdot W_{ij} \cdot \delta, 1\}$. δ with a decay factor of 0.9.

$A_{c,j}$ word vertex, which is not in the seed words, $A_{c,i}$ word belongs to seed word vertex for respective category, W_{ij} = edge weight value between $A_{c,i} \rightarrow A_{c,j}$ words vertex.

Association rules are mined when a strong relation between a notional word and one of the aspect categories exists, with the strength of the relation being modeled using the co-occurrence frequency between category and notional word.

Once each category has its own associative network, rules can be mined of the form [*notional word* \rightarrow *category*] from vertices in these networks, based on the activation value of the vertex. Since the same word can be present in multiple associative networks, one word might trigger multiple aspect categories. Based on the words in the

sentence, a set of rules is triggered and their associated aspect categories are assigned to the sentence. The result is explained in Table 2.1, the f1 score of the unsupervised method is 67.0%,

Category	TP's	FP's	FN's	τ_c	precision	recall	F_1
food	313	103	105	0.22	75.1%	74.4%	74.8%
service	100	4	72	0.19	96.2%	58.1%	72.5%
ambience	41	10	77	0.09	80.4%	34.8%	48.5%
price	52	16	31	0.09	79.0%	54.2%	64.3%
misc.	163	159	71	-	50.6%	70.9%	59.1%
all	852	157	173	-	70.0%	64.7%	67.0%

Table 2.1 Result of Unsupervised method

2.1.2 Supervised methods

The supervised method employs co-occurrence association rule mining to detect categories. Method count co-occurrence frequencies between lemmas and the annotated categories of a sentence. However, low frequency words are not taken into account in order to prevent overfitting. This is achieved using a parameter α_L , similar to the unsupervised method. Furthermore, stop words are also removed.

In addition to counting the co-occurrences of lemmas and aspect categories, the co-occurrences between grammatical dependencies and aspect categories are also counted. Similar to lemmas, low frequency dependencies are not taken into account to prevent overfitting, using the parameter α_D . Dependencies, describing the grammatical relations between words in a sentence, are more specific

than lemmas, as each dependency has three components: 1) governor word; 2) dependent word; and 3) relation type. For each dependency, the following three forms: 1) {dependency relation, governor, dependent} (D_1); 2) {dependency relation, dependent} (D_2); and 3) {dependency relation, governor} (D_3).

The co-occurrence frequencies provide the information needed to find good indicators (i.e., words or dependencies) for the categories. To determine the strength of an indicator, the conditional probability $P(B|A)$ is computed from the co-occurrence frequency, where category B is implied when lemma or dependency form A is found in a sentence. These conditional probabilities are easily computed by dividing the co-occurrence frequency of (B,A) by the occurrence frequency of A . The higher this probability, the more likely it is that A implies B . If this value exceeds a trained threshold, the lemma or dependency form indicates the presence of the corresponding category.

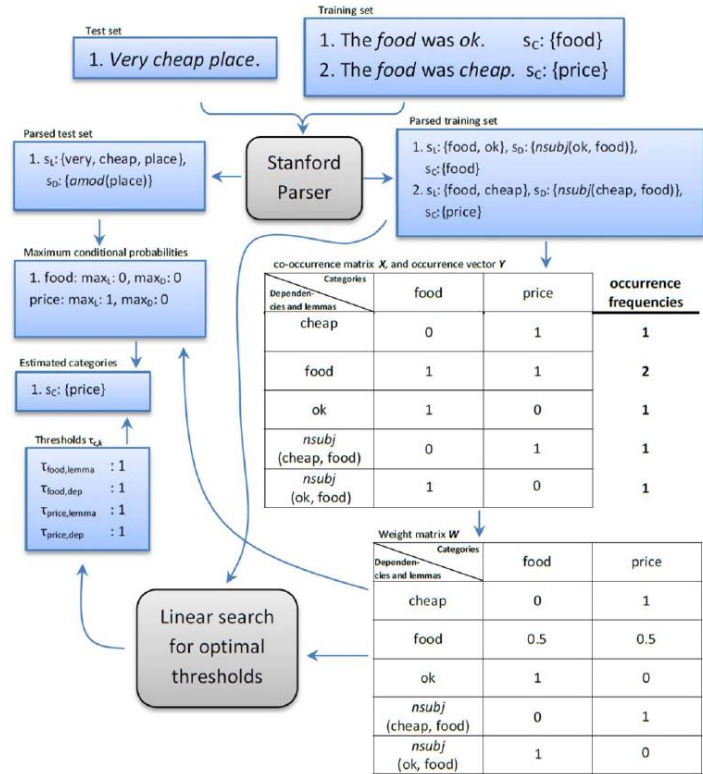


Fig. 2.2 Supervised method

This threshold that the conditional probability has to pass is different for each category. It also depends on whether a dependency form or lemma is involved, since dependency forms generally have a lower frequency, requiring a lower threshold to be effective. To find these thresholds a simple linear search is performed, picking the performing (i.e., on the training data) value from a range of values for each different threshold.

Category	TP's	FP's	FN's	precision	recall	F_1
food	371	51	47	87.9%	88.8%	88.3%
service	159	32	13	83.2%	92.4%	87.6%
ambience	83	28	35	73.8%	70.3%	72.5%
price	74	8	9	90.2%	89.2%	89.7%
anecdotes/misc.	165	38	69	81.3%	70.5%	75.5%
all	852	157	173	84.4%	83.1%	83.8%

Table 2.2: Result of Supervised method F1 Score 83.8%

Once the conditional probabilities are computed and the thresholds are known, unseen sentences from the test set are processed. For each unseen sentence The paper check whether any of the lemmas or dependency forms in that sentence have a conditional probability greater than its

corresponding threshold, in which case the corresponding category is assigned to that sentence. In table 2.2 the supervised method able to score the F1 Score is 83.8%

2.2 Paper Title: NRCCananda-2014: Detecting aspects and sentiment in customer reviews

Authors: S. Kiritchenko, X. Zhu, C. Cherry, and S. M. Mohammad

Publication: in Proc. 8th Int. Workshop Semantic Eval. (SemEval), Dublin, Ireland, 2014

The datasets can be downloaded from <http://metashare.ilsp.gr:8080/>

- 3814 reviews on Restaurants. Fix set of five aspect categories is defined for the restaurant is Food, service, price, ambiance, and anecdotes/miscellaneous.

SemEval-2014 had a shared task (Task 4) on aspect-level sentiment analysis, with over 30 teams participated. The paper describes our submissions, which stood first in detecting aspect categories, first in detecting sentiment towards aspect categories, third in detecting aspect terms, and first and second in detecting sentiment towards aspect terms.

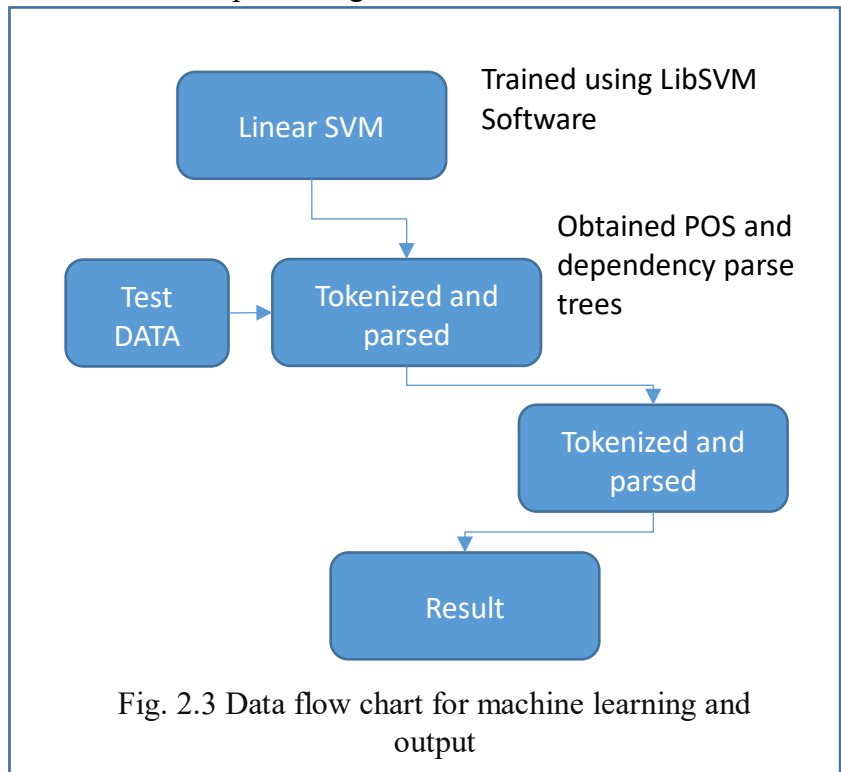


Fig. 2.3 Data flow chart for machine learning and output

The paper trains five liner SVM for each aspect category for the restaurant data. To train the machine, train data is used. The each SVM is trained to find a respective aspect in the review.

Result

The table 2.2.1 shows the result of the proposed approach F1 Score 80.19 %

System	P	R	F1
NRC-Canada (All)	84.41	76.37	80.19

2.3 Paper Title: SemEval-2014 Task 4: Aspect based sentiment analysis

Authors: M. Pontiki et al.

Publication: in Proc. 8th Int. Workshop Semantic Eval. (SemEval), Dublin, Ireland,

The paper introduced 4 subtasks for Restaurant and Laptop review data. The aspect category

for restaurant data is given but for the laptop data, there are not such categories are given. As for restaurant data [service, food, price, ambience and miscellaneous], all the four tasks are applicable for the restaurant data. But for laptop data only two tasks are open. The details explanation given below.

2.3.1 Task Description: For the first two subtasks (SB1, SB2), datasets on both domains (restaurants, laptops) were provided. For the last two subtasks (SB3, SB4), datasets only for the restaurant reviews were provided.

Aspect term extraction (SB1): in this task the machine will work to find terms available in the review. For example: “the staff was horrible” in this review staff is the term. In restaurant data set. A train data set are given where terms are already extracted. Based on the train data set. The machine has to predict Aspect term in the review. Multiple terms can be predicted in a single review.

Aspect term polarity (SB2): in this task, the polarity of the term need to be predict. So, the base data to solve this data is depends on the task 1. The polarity of the sentence in terms of positive, negative, neutral or conflict. For example, in the above sentence “the staff was horrible” the terms in the sentence is “staff” but the polarity is “negative” because “horrible” word is a negative polarity word.

Aspect category extraction (SB3): if we consider previous example. “the staff was horrible” based on terms the machine need to be predict the aspect category of the review. The implicit aspect of the review is “service”, because staff is related to service. Let see another example: “the food was delicious and service was very fast.” In this example there are two terms are available food and service, unfortunately both terms are aspect category also. So this is call explicit terms. So a review can have multiple aspect category also.

Aspect category polarity (SB4): if aspect category is extraction is done. After that the machine need to get the polarity of the aspect. if we consider Ist example. “the staff was horrible”, the aspect of the review is “service”, the polarity of the term is negative so the aspect category polarity also be negative. Let see another example: “the food was delicious and service was ok.” two aspect categories are available food and service, food has positive polarity and service has neutral polarity, so the sentence will get food as positive aspect category and service as a neutral polarity.

The labelled data is in xml format.

```
<sentence id="3121">
  <text>But the staff was so horrible to us.</text>
  <aspectTerms>
    <aspectTerm to="13" from="8" polarity="negative" term="staff"/></aspectTerms>
  <aspectCategories>
```


<aspectCategory polarity="negative" category="service"/></aspectCategories>
</sentence>

<Sentence id> is to identify unique tuple in the database file. <text> review. <aspectTerms> is showing that the total number of terms are available in the review for example in the above example there is only one term i.e. “staff”.<aspectTerm> is tag is used to mention term of the review and its polarity in the review. <aspectCategories> is used to mention total number of category are available in a particular review. And last tag <aspectCategory> is used to describe only one aspect category and its polarity based on the available terms in the review.

2.4 Paper Title: Survey on aspect-level sentiment analysis

Authors: K. Schouten and F. Frasincar

Publication: IEEE Transactions Knowledge and Data Engineering, March, 2016

In the fig. 2.4.1, the paper described all the possible methods for aspect level sentiment analysis.

2.4.1 Aspect Detection:

Aspect detection the subject of the text is called aspect detection. The author explains Frequency Based, Syntax- Based, Supervised Machine Learning and hybrid, there are possible solutions of aspect-based category detection.

2.4.2 Sentiment Analysis:

weather the text is positive or negative or neutral. Finding these polarities is called sentiment analysis. The authors mentions about dictionary-based, supervised machine learning and unsupervised machine learning.

2.4.3 Joint Aspect Detection and Sentiment Analysis:

When we do both aspect category detection and sentiment analysis task jointly the joint aspect detection and sentiment analysis is called. The authors explains four methods for this task, syntax-based, supervised machine learning, unsupervised machine learning and hybrid machine learning.

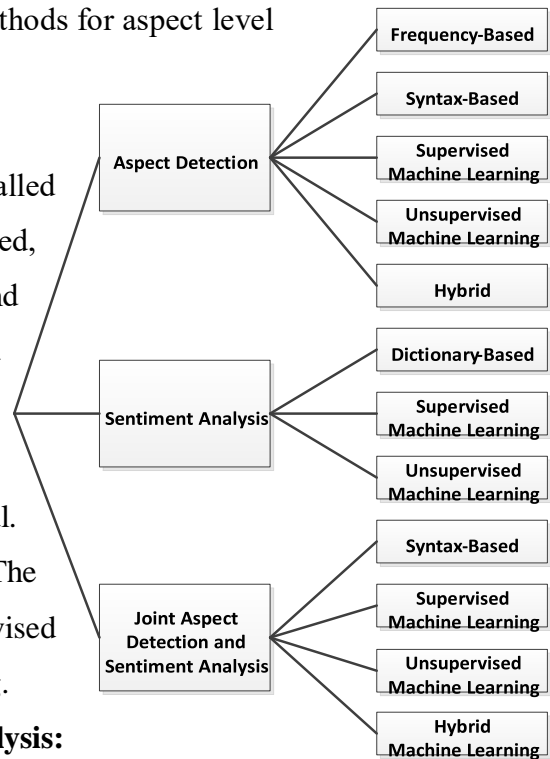


Fig. 2.4. Taxonomy for aspect-level sentiment analysis approaches using the main characteristic of the proposed algorithm.

2.5 Conclusion of literature survey

An early work, a supervised method for category detection is proposed in [1]. The author uses unsupervised method; all the words are associated in the form of a network with the category's set of seed words. With a reported F1-score of 67%. The authors use a co-occurrence technique to find closely related words, and that is used to create a co-occurrence directed graph. Each vertex is connected with another vertex based on occurrence after that vertices. Spreading activation algorithm is used to assign a value that will be calculated using various parameter defined in the paper [1]. To find, for each category the association rule is [seed word/national word \rightarrow aspect category] if the sentence does not have any seed word/national word, it will not get any aspect category, and that sentence will take default aspect category. The paper said the result of this method was F1-score of 84%.

In [2], the paper uses the supervised learning method and uses SVMs. The create five SVMs one SVM for one aspect category, it means there are five SVMs are trained. The SVM has many types of classification rule like (unigram, bi-gram and n-gram) another one is information from a word using clustering, both learned from YELP data. The lexicon explicitly derived aspect categories. The sentences, which is not comes under any target aspect category it will be sent to the post-processing steps, where the sentence will be labelled with the aspect category according to available posterior probability. The F1-score for supervised method was 88.6%

SemEval-2014[3] task aimed to adoptive research in the field of aspect-based sentiment analysis, where the goal is to identify the aspects category of given aspect category and the sentiment expression for each aspect available in the review. The task provided datasets containing manually labelled reviews of restaurants and laptops, as well as a baseline procedure. It attracted 163 submissions from 32 teams.

There are many methods are introduced in the field of NLP Machine learning. For Aspect detection in the textual data, there are supervised, unsupervised and hybrid machine learning methods are used. For Sentiment Analysis, dictionary-based, supervised and unsupervised machine learning are used. The paper [4] explain each and every method and evaluation process and its content in depth. The paper explained about Joint aspect detection and sentiment analysis methods, there are syntax-based method, supervised, unsupervised machine learning and hybrid machine learning. It also explains problems in all methods for aspect detection and sentiment analysis. The major problems are occurred by comparative opinions, conditional sentences, negations modifiers and also explained aggregation methods in brief.

For sentiment analysis from twitter data [5], authors learn the machine by the set of features they find in the test data, for example emoticon, latin alphabets, negative words, slangs, positive words, dictionary words, hashtags, targets, POS tagging, prior polarity scores of all words, by using all these features they use the combinations of the features to find best features for sentiment analysis on twitter data, the paper introduced these features as senti-features. It performs better with the accuracy of 75.39%

Another high performing sentiment analysis is proposed by pooja. Instead of a using supervised learning, it is kind of artificial intelligent application, without any classifier, the author uses a sentiwordnet3.0 corpus from natural language toolkit, to get the sentiment from words that are available in the sentiwordnet3.0. The blogger forced to use spelling correction before get the sentiment from sentiwordnet3.0. Blogger also explains the steps to setup environment to use python 3.6 with natural language toolkit (NLTK). After pre-processing steps completed then get the sentiment of each word available in the data (after preprocessing), like “tokenization, spelling correction, stop words removal, part of speech tagging etc.

CHAPTER 3

APPROACH

A system that can classify the textual data(3044 restaurant’s reviews) under given aspect categories, Service, Food, Ambience, Price and Anecdotes/Miscellaneous need to be very accurate because many systems are already been introduced and last best one is[1] the work is published in IEEE Transaction paper, April 2017. The idea to solve this aspect category detection problem by unsupervised machine learning, that the paper [1] used a method called unsupervised, its accuracy

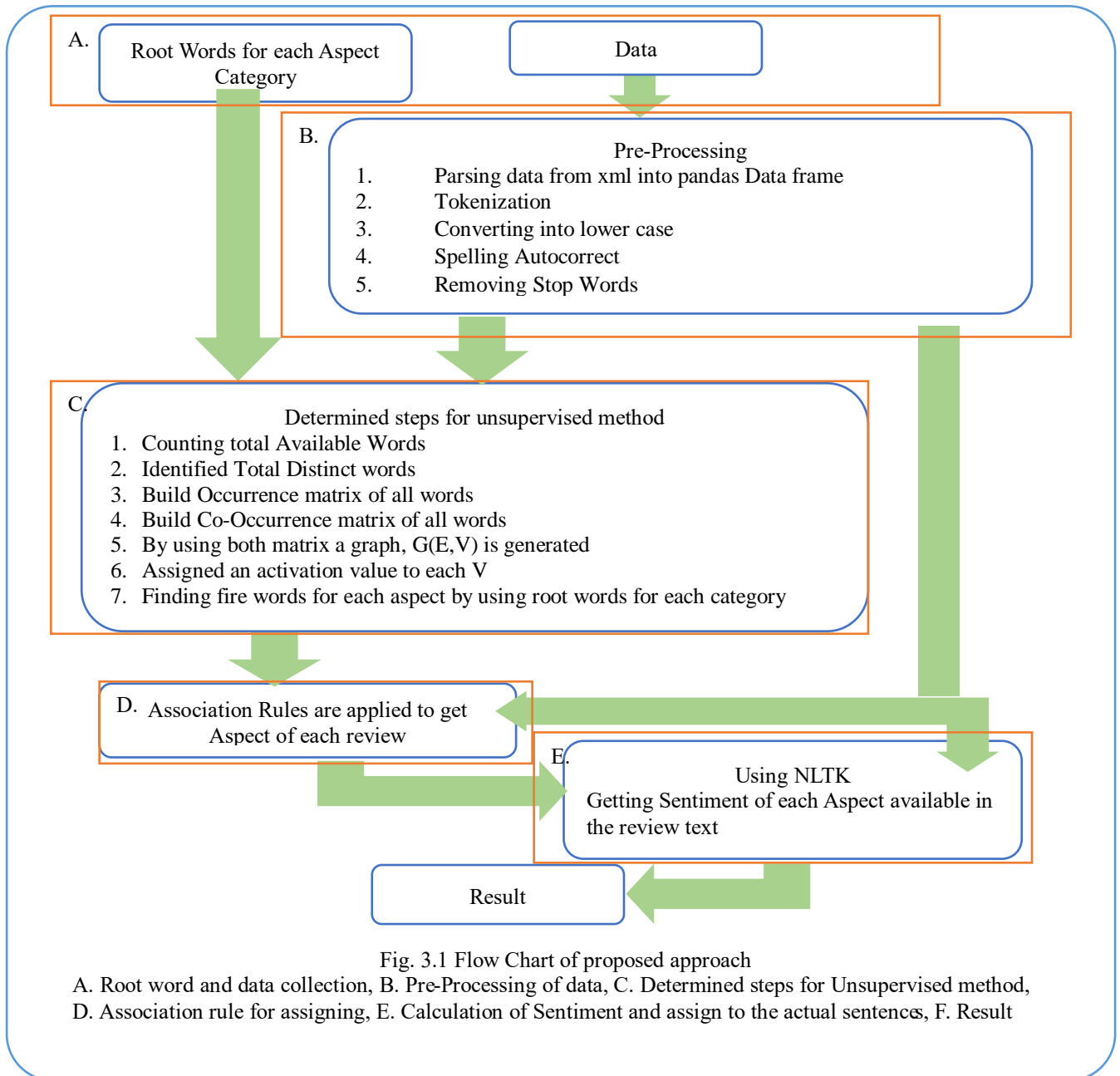


Fig. 3.1 Flow Chart of proposed approach

A. Root word and data collection, B. Pre-Processing of data, C. Determined steps for Unsupervised method, D. Association rule for assigning, E. Calculation of Sentiment and assign to the actual sentences, F. Result

speaks about the work, that it is not up to the mark, and if anyone get used this work, it will not give a respective result. So, our intention is to make the work useable, where the data is not available to make a supervised machine that can process a natural language for aspect category detection. We should have a technique that the data can be categorized by todays AI techniques. So now the proposed approach result is up to the mark, which has better accuracy then the accuracy of my base paper [1].

3.1 Proposed Approach

The proposed approach (fig.3.1) is a better solution for the same data set used in paper [1], data is a xml format which has set of reviews of restaurant, there are 3044 reviews which are already categorised under all five aspect categories [food-1233, anecdotes/miscellaneous – 1133, service – 597, Ambience – 432 and price – 319] all 3044 reviews after categorising will become 3714 reviews. Because it is possible to have multiple aspects in a review.

The proposed method is an unsupervised method [3] and co-occurrence relation and spreading activation algorithm [1], it will totally depend on root words, which can be called from Wordnet for each aspect category [1] and some special {for example:-food→ lassi, dosa, salads etc.} words that are not available in the Wordnet, that are closely related to the food, and likewise for other aspect categories.

The root words are the major part of this approach, these are not easy to put into the root words list, because a single wrong word can fail the system. These words are helpful to find fire words for each and every aspect category. So, we need to be very careful to select root words for particular aspect. After collection of all root words for each category. The data will be pre-processed and all pre-processing steps need to be performed, determined steps for Unsupervised method, to build occurrence matrix (X) need to find total distinct words are in the data, after that co-occurrence matrix (Y) is designed using all distinct words available in the entire reviews.

The co-occurrence directed graph $G(V, E)$ will be designed based on co-occurrence matrix of words, if Y_i and Y_j are occurred in the same review and the occurrence of Y_i is before Y_j then, the Y_j will have an edge directed from Y_i . The same method will be applicable to entire data. The weight of each edge between two vertices will be decided by $(W_{i,j}) = \frac{Y_{i,j}}{x_j}$, Generating the activation value to each V will be calculated by applying spreading activation algorithm[1], After getting activation value it will be decided that which word is close word to the root words and of the which aspect category, those words activation value will be used to sum-up and if the value crosses the threshold value, then the aspect category will be assign for that sentence by implicitly[1]. After assign the aspect category

the plain text and pre-processed data that is called comes under that particular aspect will be used to calculate for sentiment analysis for that particular aspect category. The sentiment score will be called from sentiwordnet3.0. and assigned to the sentence that is already have aspect.

Spreading activation algorithm, in this algorithm we have four input \rightarrow root words, graph(V,E), decay factor, threshold value and we will get two output \rightarrow list of fire-words and activation value for each word of graph. For each vertices of graph $V(G) \leftarrow 0$ will be assigned. After that for each $G(V_i)$ that belong to Root Words (R_c) $\leftarrow 1$ will be assigned. Rest of the vertices will get the activation value $A_{vj} \leftarrow \text{Min}(A_{vj} + A_{vi} * W_{ij} * dc, 1)$, if the value of A_{vj} is greater than threshold value i.e. 0.1, the word will be added into the fire word list for the respective aspect category. This process will be done for each aspect category.

Association rule algorithm will be used to get aspect category of each sentence and for each aspect category. In this we have four inputs for each aspect category (Review, Graph(V,E), Threshold Value \rightarrow A_{tc} and FireWordList \rightarrow F_c and one output Aspect Category for the text \rightarrow T . 1. For each word in sentence T_i . 2. If T_i is available in the graph and its value is 1 than 3. The sentence will get the aspect category for respective input aspect category. If not 4. Find the co-occurrence words in the graph and sum-up the activation values and compare with Threshold value aspect category ($A_{tc} \rightarrow 1.0$) if sum-up value is greater than or equal to A_{tc} , the text will get the respect category if less than it will be assigned default aspect category.

Algorithm1: Activation Spreading Algorithm

```

Input root_words  $\rightarrow R_c$ 
Input graph  $\rightarrow G(V,E)$ 
Input decay  $\rightarrow dc=0.9$ 
Input Threshold Value  $tc \leftarrow 0.1$ 
Output FireWordList  $\rightarrow F$ 
Output Activation Value  $\rightarrow A_v$ 
1 For each  $i \in G(V,E)$  do
2    $A_{v,i} \leftarrow 0$ 
3 For each  $i \in R_c$  do
4    $A_{v,r} \leftarrow 1$ 
5   add  $i$  to  $F$ 
6   end
7  $A_{vj} \leftarrow \text{Min}(A_{vj} + A_{vi} * W_{ij} * dc, 1)$ 
8 foreach  $i \in V \setminus F$  do
9   if  $A_{v,i} > tc$  then
10    add  $j$  to  $F$ 

```

Association Rule Algorithm 2 (ARA)

```

Input Review -  $T$ 
Input graph-  $G(V,E) \in C$ 
Input Threshold Value -  $A_{tc} = 1.0$ 
Input FireWordList -  $F_c$ 
Output  $A_c$  for each  $T$ 
1 For each  $i \in T$  do
2   if  $i == 1$  in  $V$ 
3     [ $T = \text{category } c$ ]
4     if  $i$  in  $F_c$ 
5        $A_{v,i} += A_{v,i}$ 
6 If  $A_{v,i} \geq A_{tc}$ 
7   [ $T = \text{category } c$ ]
8 else
9   [ $T = \text{category Default}$ ]

```

CHAPTER 4

IMPLEMENTATION

The code

```
#import networkx as nx
from xml.dom import minidom
#from nltk.stem import PorterStemmer
from autocorrect import spell

import pandas as pd
import nltk
from nltk.corpus import wordnet as wn
#from stemming.porter2 import stem

#from textblob import TextBlob
#import jgraph
#import numpy as np

#mydoc = minidom.parse('restaurants-trial.xml')
mydoc = minidom.parse('Restaurants_Train.xml')
text = list(mydoc.getElementsByTagName('text'))
data=[]
tokenword=[]
countwords=[]
distinct_words=[]
clean=[]

not_fired_word=[]
sentiwordsinreview=[]
sentiment=[]
tc=0.7
dt=0.9
for elem in text:
    data.append([elem.firstChild.data])

del(text)
df_data=pd.DataFrame(data)

stop_words = ["mightn't", 'or', 'doesn', 'y', 'should', 'doing', 'after',
'this', 'those', 'with', 'his', 'herself', 'aren', 'which', 'won', 'itself',
'wouldn', 'it', 'has', 'at', 'is', 'further', 'some', 'myself', "you'd",
've', 'was', 'to', 'until', 'no', "shouldn't", 'couldn', 'whom', 'of',
'between', 'from', 'her', 'me', 'its', 'hasn', 'these', 'through', "wasn't",
"don't", 'theirs', 'hadn', 'into', 'been', 'again', 'yours', 'she', 'mightn',
'under', 'while', 'just', 'had', 'about', 'below', 'the', 'll', 'hers',
"didn't", 'how', 'm', 'such', "hadn't", 'what', 'be', 'did', 'but', 'each',
't', 'by', 'didn', 'before', 'off', 'where', 'most', 'there', 'ourselves',
'our', 'over', "it's", 'will', 'own', 'he', 'why', 'if', 'weren', 'during',
'have', 'ours', 'more', 'other', 'very', 're', 'above', 'are', 'down',
'shan', 'shouldn', 'an', 'too', 'isn', 'mustn', 'you', 'o', "doesn't", 'not',
'haven't", 'they', 'both', 'don', 'does', 'only', 'himself', "aren't",
```

```
'your', "hasn't", 'haven', 'all', "that'll", "she's", 'out', "needn't", 'on',
'nor', "shan't", 'needn', 'then', 'having', 'so', "wouldn't", 'who',
"you're", "isn't", 'as', 'for', 'them', 'i', "you'll", 'my', 'do', 'here',
's', 'ain', 'themselves', 'ma', 'yourselves', 'we', 'being', "mustn't",
"you've", 'that', 'now', 'their', 'few', 'up', 'a', 'in', 'can', 'against',
'same', "couldn't", "won't", 'because', 'than', 'd', 'were', 'once', 'him',
'and', 'any', "weren't", 'yourself', 'am', "should've", 'when',
'wasn't', '@', '#', '$', '(', '.', ',')']
```

```
for line in data:
    for word in line:
        words = nltk.word_tokenize(word.lower())
        tokenword.append(words)
```

```
df_data['tokenized_words']=tokenword
```

```
autocorrect=[]
```

```
for lines in tokenword:
    linesword=[]
    for word in lines:
        linesword.append(spell(word))
    autocorrect.append(linesword)
```

```
df_data['Autocorrect']=autocorrect
```

```
words=df_data['Autocorrect']
```

```
clean = words.apply(lambda x: [word for word in x if word not in stop_words])
```

```
df_data['clean_text']=clean
```

```
for words in clean:
    for word in words:
        countwords.append(word)
```

```
del(words,word)
#clean_words =list([text.strip().lower() for text in countwords if not
text in stop_words])
```

```
for w in countwords:
    if not w in distinct_words:
        distinct_words.append(w)
```

```
del(w)
```

```
df_occurrence=pd.DataFrame(distinct_words)
```

```
oc_table=[0 for i in range(1) for j in range(len(distinct_words))]
```

```
for w1 in range(len(distinct_words)):
    for w2 in range(len(countwords)):
        if distinct_words[w1]==countwords[w2]:
            oc_table[w1]+=1
```

```
del(w1,w2)
```



```

df2=pd.DataFrame(data=0,index=distinct_words[0:],columns=distinct_words[0:])

df2['occurrence']=oc_table

for i in range(len(clean)):
    for j in range(len(clean[i])-1):
        for k in range(j+1, len(clean[i])):
            a=int(i)
            b=int(j)
            c=int(k)
            w1=clean[a][b]
            w2 =clean[a][c]
            if w1 != w2:
                df2.loc[w1, w2] += 1
                #com[w1][w2] += 1
del (w1,w2,i,j,k,a,b,c)

df_occurrence['Occurrence']=oc_table
df2['occurrence']=oc_table

graph_df2=pd.DataFrame(data=0,index=distinct_words[0:],columns=distinct_words
[0:])
for i in range((len(distinct_words))-1):
    for j in range(i+1,len(distinct_words)):
        w1=distinct_words[i]
        w2=distinct_words[j]
        if w1 != w2:
            graph_df2.loc[w1,w2]=df2[w2][w1]/df2['occurrence'][w2]
            #edg_weight.append(weight)
del (w1,w2,i,j)

root_words_food=['bagels', 'cake', 'potato', 'onion', 'garlic', 'burger',
'dishes', 'dish', 'salads', 'food', 'eating', 'eat', 'perks', 'lime', 'ice',
'beer', 'dosa', 'tomato', 'sausage', 'chicken', 'meats', 'traditional',
'ingredients', 'sardines', 'cheese', 'omelet', 'drinks', 'cuisine', 'pizza',
'eaten', 'tasty', 'salmon', 'rice', 'pickles', 'seafoods', 'sandwich',
'tuna', 'fish', 'soup', 'seafoods', 'seafood', 'sushi', 'pastry', 'goat',
'salad', 'pork', 'delicious', 'noodles', 'meal', 'desert', 'nutrient',
'solid_food', 'breads', 'lamb', 'tapas', 'bread', 'egg', 'eggs']
root_words_price=['deal', 'high', 'prices', 'priced', 'budget', 'afford',
'inexpens', 'cheap', 'expense', 'price', 'term', 'damage', 'bargain', 'cost',
'bill', 'bills', 'buy', 'cash', 'discount', 'invoice', 'off', 'overprice',
'Purchas', 'toll']
root_words_services=['menu', 'server', 'waitress', 'order', 'staff',
'deliver', 'friend', 'avail', 'help', 'serv', 'service']
root_words_ambience=['ambience', 'floor', 'place', 'crowd', 'block',
'ambient', 'atmosphere', 'room', 'area']

fired_word_food=[]
fired_word_price=[]
fired_word_services=[]
fired_word_ambience=[]
food=[]
price=[]

```

```

services=[]
ambience=[]
sentiment=[]

def set_aspect1(root_words_food):

activation_val=pd.DataFrame(data=0,index=distinct_words[0:],columns=distinct_
words[0:])
    for w1 in root_words_food:
        if w1 in distinct_words:
            activation_val[w1][w1]=1
            if w1 not in fired_word_food:
                fired_word_food.append(w1)

    for i in range(len(clean)):
        for j in range(len(clean[i])-1):
            for k in range(j+1, len(clean[i])):
                a=int(i)
                b=int(j)
                c=int(k)
                w1=clean[a][b]
                w2 =clean[a][c]
                if w1 != w2:

activation_val.loc[w1,w2]=min((activation_val[w2][w1])+(activation_val[w1][w1
])* (graph_df2.loc[w1,w2])*(dt),1)
                    if activation_val.loc[w1,w2]>tc:
                        if w2 not in fired_word_food:
                            fired_word_food.append(w2)

    return activation_val

def set_aspect2(root_words_price):

activation_val=pd.DataFrame(data=0,index=distinct_words[0:],columns=distinct_
words[0:])
    for w1 in root_words_price:
        if w1 in distinct_words:
            activation_val[w1][w1]=1
            if w1 not in fired_word_price:
                fired_word_price.append(w1)

    for i in range(len(clean)):
        for j in range(len(clean[i])-1):
            for k in range(j+1, len(clean[i])):
                a=int(i)
                b=int(j)
                c=int(k)
                w1=clean[a][b]
                w2 =clean[a][c]
                if w1 != w2:

activation_val.loc[w1,w2]=min((activation_val[w2][w1])+(activation_val[w1][w1
])* (graph_df2.loc[w1,w2])*(dt),1)
                    if activation_val.loc[w1,w2]>tc:
                        if w2 not in fired_word_price:

```

```

        fired_word_price.append(w2)

    return activation_val
def set_aspect3(root_words_services):

activation_val=pd.DataFrame(data=0,index=distinct_words[0:],columns=distinct_
words[0:])
    for w1 in root_words_services:
        if w1 in distinct_words:
            activation_val[w1][w1]=1
            if w1 not in fired_word_services:
                fired_word_services.append(w1)

    for i in range(len(clean)):
        for j in range(len(clean[i])-1):
            for k in range(j+1, len(clean[i])):
                a=int(i)
                b=int(j)
                c=int(k)
                w1=clean[a][b]
                w2 =clean[a][c]
                if w1 != w2:

activation_val.loc[w1,w2]=min((activation_val[w2][w1])+(activation_val[w1][w1
])* (graph_df2.loc[w1,w2])*(dt),1)
                    if activation_val.loc[w1,w2]>tc:
                        if w2 not in fired_word_services:
                            fired_word_services.append(w2)

    return activation_val
def set_aspect4(root_words_ambience):

activation_val=pd.DataFrame(data=0,index=distinct_words[0:],columns=distinct_
words[0:])
    for w1 in root_words_ambience:
        if w1 in distinct_words:
            activation_val[w1][w1]=1
            if w1 not in fired_word_ambience:
                fired_word_ambience.append(w1)

    for i in range(len(clean)):
        for j in range(len(clean[i])-1):
            for k in range(j+1, len(clean[i])):
                a=int(i)
                b=int(j)
                c=int(k)
                w1=clean[a][b]
                w2 =clean[a][c]
                if w1 != w2:

activation_val.loc[w1,w2]=min((activation_val[w2][w1])+(activation_val[w1][w1
])* (graph_df2.loc[w1,w2])*(dt),1)
                    if activation_val.loc[w1,w2]>tc:
                        if w2 not in fired_word_ambience:
                            fired_word_ambience.append(w2)

```

```

return activation_val

food=set_aspect1(root_words_food)
price=set_aspect2(root_words_price)
services=set_aspect3(root_words_services)
ambience=set_aspect4(root_words_ambience)
df = pd.DataFrame(columns=['Sentances','Category','Sentiment'])

Sentances=[]
category=[]
final_sentiment=""

def sentiment_analysis(review):

    import nltk
    from nltk.corpus import sentiwordnet as swn

# Part of Speach tagging to each word
    tagged_texts = nltk.pos_tag(review)
    final_sentiment=""
# creating an empty variable for storing the senti words and their POS score
    ss_set = []
    pos_score=neg_score=token_count=final_score=0
# loop for identifing and storing POS score into impty set
    for word, tag in tagged_texts:
        if 'NN' in tag and list(swn.senti_synsets(word, 'n')):
            ss_set = list(swn.senti_synsets(word, 'n'))[0]
        elif 'VB' in tag and list(swn.senti_synsets(word, 'v')):
            ss_set = list(swn.senti_synsets(word, 'v'))[0]
##
        print(ss_set)
        elif 'JJ' in tag and list(swn.senti_synsets(word, 'a')):
            ss_set = list(swn.senti_synsets(word, 'a'))[0]
##
        print(ss_set)
        elif 'RB' in tag and list(swn.senti_synsets(word, 'r')):
            ss_set = list(swn.senti_synsets(word, 'r'))[0]
##
        print(ss_set)
# calculating the full score of the review
    if ss_set:
        pos_score += ss_set.pos_score()
        neg_score += ss_set.neg_score()

# Counting the total words available in the set for sentiment score
    token_count += 1
    final_score = pos_score - neg_score
##
    print(final_score)
##
    print(token_count)
    norm_final_score = round(float(final_score) / token_count, 2)
    if norm_final_score>0:
        if neg_score>=1:
            final_sentiment = 'negative'
        if neg_score>1:
            final_sentiment = 'conflict'
        else:
            final_sentiment = 'positive'

```

```

elif norm_final_score<0:
    if pos_score>=1:
        final_sentiment = 'Positive'
    if pos_score>1:
        final_sentiment = 'conflict'
    else:
        final_sentiment = 'negative'
elif norm_final_score==0:
    if pos_score>neg_score:
        final_sentiment = 'Positive'
    if neg_score>pos_score:
        final_sentiment = 'negative'
    else:
        final_sentiment = 'neutral'
return final_sentiment

def aspectfortext(plaintext, data):
    sentiment_food=[]
    sentiment_ambience=[]
    sentiment_price=[]
    sentiment_service=[]

    for i in range(len(plaintext)):
        word=plaintext[i]
        if ambience.loc[word][word]==1:
            a=1
            sentiment_ambience.append(word)
            for j in range(i+1,len(plaintext)):
                word2=plaintext[j]
                if word2 in fired_word_ambience:
                    sentiment_ambience.append(word2)
                    a+=float(ambience.loc[word][word2])
            if a>=1:
                Sentances.append(data)
                category.append("ambience")
                sentiment.append(sentiment_analysis(sentiment_ambience))
        elif ambience.loc[word][word]!=1:
            a=ambience.loc[word][word]
            if word in fired_word_ambience:
                sentiment_ambience.append(word)

            for j in range(i+1,len(plaintext)):
                word2=plaintext[j]
                if word2 in fired_word_ambience:
                    a+=float(ambience.loc[word][word2])
                    sentiment_ambience.append(word2)
            if a>=1:
                Sentances.append(data)
                category.append("ambience")
                sentiment.append(sentiment_analysis(sentiment_ambience))

    if food.loc[word][word]==1:
        a=1
        sentiment_food.append(word)
        for j in range(i+1,len(plaintext)):
            word2=plaintext[j]

```

```

        if word2 in fired_word_food:
            a+=float(food.loc[word][word2])
            sentiment_food.append(word2)
            a+=float(food.loc[word][word2])

    if a>=1:
        Sentances.append(data)
        category.append("food")
        sentiment.append(sentiment_analysis(sentiment_food))
elif food.loc[word][word]!=1:
    a=food.loc[word][word]
    if word in fired_word_food:
        sentiment_food.append(word)
    for j in range(i+1,len(plaintext)):
        word2=plaintext[j]
        if word2 in fired_word_food:
            a+=float(food.loc[word][word2])
            sentiment_food.append(word2)
    if a>=1:
        Sentances.append(data)
        category.append("food")
        sentiment.append(sentiment_analysis(sentiment_food))

if price.loc[word][word]==1:
    a=1
    sentiment_price.append(word)
    for j in range(i+1,len(plaintext)):
        word2=plaintext[j]
        if word2 in fired_word_price:
            a+=float(price.loc[word][word2])
            sentiment_price.append(word2)
    if a>=1:
        Sentances.append(data)
        category.append("price")
        sentiment.append(sentiment_analysis(sentiment_price))
elif price.loc[word][word]!=1:
    a=price.loc[word][word]
    if word in fired_word_price:
        sentiment_price.append(word)
    for j in range(i+1,len(plaintext)):
        word2=plaintext[j]
        if word2 in fired_word_price:
            a+=float(price.loc[word][word2])
            sentiment_price.append(word2)
    if a>=1:
        Sentances.append(data)
        category.append("price")
        sentiment.append(sentiment_analysis(sentiment_price))

if services.loc[word][word]==1:
    a=1
    sentiment_service.append(word)
    for j in range(i+1,len(plaintext)):
        word2=plaintext[j]
        if word2 in fired_word_services:
            sentiment_service.append(word2)

```

```

        a+=float (services.loc[word] [word2])
    if a>=1:
        Sentances.append(data)
        category.append("service")
        sentiment.append(sentiment_analysis(sentiment_service))
    elif services.loc[word] [word]!=1:
        a=services.loc[word] [word]
        if word in fired_word_services:
            sentiment_service.append(word)
        for j in range(i+1,len(plaintext)):
            word2=plaintext[j]
            if word2 in fired_word_services:
                sentiment_service.append(word2)
                a+=float (services.loc[word] [word2])
        if a>=1:
            Sentances.append(data)
            category.append("service")
            sentiment.append(sentiment_analysis(sentiment_service))
    else:
        Sentances.append(data)
        restset=[]
        for j in range(i+1,len(plaintext)):
            word2=plaintext[j]
            restset.append(word2)
        category.append("anecdotes/miscellaneous")
        sentiment.append(sentiment_analysis(restset))
    if data not in Sentances:
        Sentances.append(data)
        category.append("anecdotes/miscellaneous")
        sentiment.append(sentiment_analysis(plaintext))

for line in range(len(clean)):
    sentence=data[line]
    plaintext=clean[line]
    aspectfortext(plaintext,sentence)

df['Sentances']=Sentances
df['Category']=category
df['Sentiment']=sentiment

writer = pd.ExcelWriter('Approachoutput.xlsx')
df.to_excel(writer,'Sheet1')
writer.save()

def result():
    import pandas as pdr

    resultdata = pdr.ExcelFile("Approachoutput.xlsx")
    resultdata.sheet_names
    [u'Sheet1']
    dfnew = resultdata.parse("Sheet1")

    dfnew["is_duplicate"]= dfnew.duplicated()

```

```

NewData=pd.DataFrame(dfnew.loc[dfnew['is_duplicate']==False])
finalreseult_df=pd.DataFrame(NewData)
new=finalreseult_df.drop('is_duplicate',axis=1)

finalreseult_df=pdr.DataFrame(new)

writer = pdr.ExcelWriter('FinalResult17051811.xlsx')
finalreseult_df.to_excel(writer,'Sheet1')
writer.save()

result()

```

4.1 Data description

There are total 3044 sentences in the data set. The set of files is downloaded from <http://alt.qcri.org/semEval2014/task4/index.php?id=data-and-tools>, and baseline method is also available to download. The description of the data is, out of 3044 reviews food = 1233, anecdotes/miscellaneous= 1113, Service=597, Ambience= 432 and Price = 319.

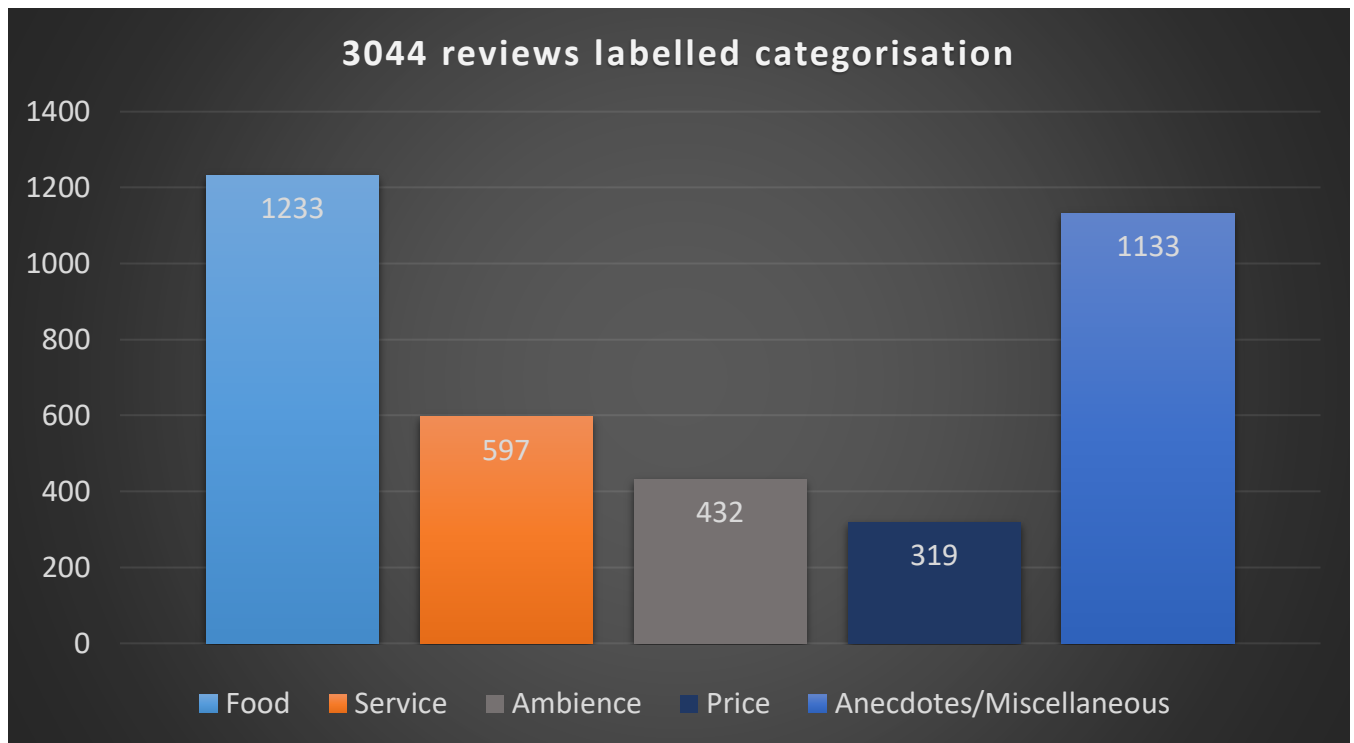


Fig. 4.1 3044 reviews labelled categorisation

4.2 Root Words

For target aspect categories, some special words are identified from the WordNet, wordnet is nothing but a corpus where we can find the relation between two words. How much close a word x is related to word y or word z. The wordnet is freeware available, and easily can be attached to the project,

my project is implemented in the python 3.6 and a library can be called into the project. Some special word, which is not extracted from the wordnet, that are included by data study. These words are available in the description of the entity, entity is nothing but the restaurant. Restaurant's menu can give us the food related words like {salad, paratha, lassi etc.}. There are four target aspect categories and if the review is not related to them, then it will go to default (anecdotes/miscellaneous) category. There is not any wordlist for default category.

4.3 Preprocessing

In our approach the following steps are done on the data.

4.3.1 Parsing data from xml into pandas data frame

The data in xml format, xml database file is encoded in some special tags. It is very easy to store the data and no need to install any other software to read this file in any Operating System. It can be easily open in notepad type text related application software. It can easily open in the excel application software.

```
<sentence id="3121">
  <text>But the staff was so horrible to us.</text>
  <aspectTerms>
    <aspectTerm to="13" from="8" polarity="negative" term="staff"/></aspectTerms>
  <aspectCategories>
    <aspectCategory polarity="negative" category="service"/></aspectCategories>
</sentence>
```

<Sentence id> is to identify unique tuple in the database file. <text> is to read the review. <aspectTerms> is showing that the total number of terms are available in the review for example in the above example there is only one term i.e. "staff". <aspectTerm> is tag is used to mention term of the review and its polarity in the review. <aspectCategories> is used to mention total number of category are available in a particular review. And last tag <aspectCategory> is used to describe only one aspect category and its polarity based on the available terms in the review. So there are total 3044 reviews.

We have to parse only <text> for our unsupervised approach. So we have used “xml” python library and its “xml.dom” function to parse <text> tag in to our code and then to make further operation on the total reviews, it is stored into pandas dataframe.

Pandas is a library for python language, and it is very good for manipulation in the data.

The text will come into this format

Example :-

```
[Best of all is the warm vibe, the owner is super friendly and service is fast.]  
[The food was delicious but do not come here on a empty stomach.]  
[Go here for a romantic dinner but not for an all out wow dining experience.]  
[The menu is limited but almost all of the dishes are excellent.]
```

4.3.2 Tokenization

Tokenization is done by natural language toolkit predefined function nltk.word_tokenize. NLTK is a platform for python language, it offers over 50 corpuses and lexical resources for example sentiwordnet3.0, Wordnet etc. NLTK provides suite to process textual data, like tokenization, tagging, semantic reasoning, stemming for strengthen NLP libraries.

Example output :-

```
['Best', 'of', 'all', 'is', 'the', 'warm', 'vibe', ',', 'the', 'owner', 'is', 'super', 'friendly', 'and', 'service', 'is', 'fast', '.']  
['The', 'food', 'was', 'delicious', 'but', 'do', 'not', 'come', 'here', 'on', 'a', 'empty', 'stomach', '.']  
['Go', 'here', 'for', 'a', 'romantic', 'dinner', 'but', 'not', 'for', 'an', 'all', 'out', 'wow', 'dining', 'experience', '.']  
['The', 'menu', 'is', 'limited', 'but', 'almost', 'all', 'of', 'the', 'dishes', 'are', 'excellent', '.']
```

4.3.3 Converting into lower case

The main reason behind to convert text into lower case is, when we are going to remove stop word from the actual data, may be can the reviews have multiple uppercase and lowercase combinations to remove all stop words from the data there are only two ways, first one, is to identify each and every word that are present in the data and then remove from the data, which will be difficult and time consuming effort. And another way to just use the corpus of stop words from NLTK and convert them similar case and same can be removed from the data.

The second way to remove the stop words are easy and less time consuming then first one.

Example output: -

['best', 'of', 'all', 'is', 'the', 'warm', 'vibe', ',', 'the', 'owner', 'is', 'super', 'friendly', 'and', 'service', 'is', 'fast', '.']
['the', 'food', 'was', 'delicious', 'but', 'do', 'not', 'come', 'here', 'on', 'a', 'empty', 'stomach', '.']
['go', 'here', 'for', 'a', 'romantic', 'dinner', 'but', 'not', 'for', 'an', 'all', 'out', 'wow', 'dining', 'experience', '.']
['the', 'menu', 'is', 'limited', 'but', 'almost', 'all', 'of', 'the', 'dishes', 'are', 'excellent', '.']

4.3.4 Spelling Autocorrect

By correcting the spelling of each available word, it is helpful to find sentiment score from sentiwordnet3.0. If a term available in the review is misspelling then it would not be mapped from sentiwordnet3.0, then the sentiment of the word will not be considered in our approach. The main reason to use it, because we are not learning machine, we are using AI technique to get aspect with its sentiment.

4.3.5 Removing Stop Words

According to standford.edu, some particularly common words, those word's appearance have little value in favoring select documents matching, a user need are excluded from the vocabulary entirely. Those words are stop words.

The following 179 stop words are used in our approach.

["mightn't", 'or', 'doesn't', 'y', 'should', 'doing', 'after', 'this', 'those', 'with', 'his', 'herself', 'aren', 'which', 'won', 'itself', 'wouldn't', 'it', 'has', 'at', 'is', 'further', 'some', 'myself', "you'd", 've', 'was', 'to', 'until', 'no', "shouldn't", 'couldn't', 'whom', 'of', 'between', 'from', 'her', 'me', 'its', 'hasn', 'these', 'through', "wasn't", 'don't', 'theirs', 'hadn', 'into', 'been', 'again', 'yours', 'she', 'mightn', 'under', 'while', 'just', 'had', 'about', 'below', 'the', 'll', 'hers', "didn't", 'how', 'm', 'such', "hadn't", 'what', 'be', 'did', 'but', 'each', 't', 'by', 'didn', 'before', 'off', 'where', 'most', 'there', 'ourselves', 'our', 'over', "it's", 'will', 'own', 'he', 'why', 'if', 'weren', 'during', 'have', 'ours', 'more', 'other', 'very', 're', 'above', 'are', 'down', 'shan', 'shouldn', 'an', 'too', 'isn', 'mustn', 'you', 'o', "doesn't", 'not', "haven't", 'they', 'both', 'don', 'does', 'only', 'himself', "aren't", 'your', "hasn't", 'haven', 'all', "that'll", "she's", 'out', "needn't", 'on', 'nor', "shan't", 'needn', 'then', 'having', 'so', "wouldn't", 'who', "you're", "isn't", 'as', 'for', 'them', 'i', "you'll", 'my', 'do', 'here', 's', 'ain', 'themselves', 'ma', 'yourselves', 'we', 'being', "mustn't", "you've", 'that', 'now', 'their', 'few', 'up', 'a', 'in', 'can', 'against', 'same', "couldn't", "won't", 'because', 'than', 'd', 'were', 'once', 'him', 'and', 'any', "weren't", 'yourself', 'am', "should've", 'when', 'wasn', '!', '@', '#', '"', '\$', '(, !,)']

After removal of stop words the processed data will be look like :

Example of output:

```
[ 'Best', 'warm', 'vibe', 'owner', 'super', 'friendly', 'service', 'fast' ]
[ 'food', 'delicious', 'come', 'empty', 'stomach' ]
[ 'Go', 'romantic', 'dinner', 'out', 'wow', 'dining', 'experience' ]
[ 'menu', 'limited', 'dishes', 'excellent' ]
```

Index	0	tokenized_words	Autocorrect	clean_text
0	But the staff was so horrible to us.	['but', 'the', 'staff', 'was', 'so', 'horrible', 'to', 'us', '.']	['but', 'the', 'staff', 'was', 'so', 'horrible', 'to', 'us', 'a']	['staff', 'horrible', 'us']
1	To be completely fair, the only redeeming factor was ...	['to', 'be', 'completely', 'fair', 'the', 'only', 'redeeming', 'factor', 'wa...']	['to', 'be', 'completely', 'fair', 'a', 'the', 'only', 'redeeming', ...]	['completely', 'fair', 'redeeming', 'factor', 'food...']
2	The food is uniformly exceptional, with a very c...	['the', 'food', 'is', 'uniformly', 'exceptional', 'with', 'a', 'very', ...]	['the', 'food', 'is', 'uniformly', 'exceptional', 'a', 'with', 'a', ...]	['food', 'uniformly', 'exceptional', 'capable', 'k...']
3	Where Gabriela personally greets you and recommends ...	['where', 'gabriela', 'personally', 'greet', 'you', 'and', 'recommends', 'you...']	['where', 'Gabriela', 'personal', 'greet', 'you', 'and', 'recommen...']	['Gabriela', 'personal', 'greet', 'recommends', 'eat...']
4	For those that go once and don't enjoy it, all I can ...	['for', 'those', 'that', 'go', 'once', 'and', 'do', 'n't', 'enjoy', 'it', 'all', 'I', 'c...']	['for', 'those', 'that', 'go', 'once', 'and', 'do', 'not', 'enjo...']	['go', 'enjoy', 'say', 'get']
5	Not only was the food outstanding, but the litt...	['not', 'only', 'was', 'the', 'food', 'outstanding', 'but', 'the', 'little...']	['not', 'only', 'was', 'the', 'food', 'outstanding', 'a', 'but...']	['food', 'outstanding', 'little', 'perks', 'great']
6	It is very overpriced and not very tasty.	['it', 'is', 'very', 'overpriced', 'and', 'not', 'very', 'tasty', '.']	['it', 'is', 'very', 'overpriced', 'and', 'not', 'very', 'tasty', 'a...']	['overpriced', 'tasty']
7	Our agreed favorite is the orrechiete with sausage an...	['our', 'agreed', 'favorite', 'is', 'the', 'orrechiete', 'with', 'sausage', 'and', '...']	['our', 'agreed', 'favorite', 'is', 'the', 'orrechiete', 'with...']	['agreed', 'favorite', 'orrechiete', 'sausage', 'ch...']
8	The Bagels have an outstanding taste with a t...	['the', 'bagels', 'have', 'an', 'outstanding', 'taste', 'with', 'a', 'ter...']	['the', 'bagels', 'have', 'an', 'outstanding', 'taste', 'with', ...]	['bagels', 'outstanding', 'taste', 'terrific', 'textur...']
9	Nevertheless the food itself is pretty good.	['nevertheless', 'the', 'food', 'itself', 'is', 'pretty', 'good', '.']	['nevertheless', 'the', 'food', 'itself', 'is', 'pretty', 'good', ...]	['nevertheless', 'food', 'pretty', 'good']
10	I've noticed megadeath, mr. scruff, st. germain, tradi...	['i', 've', 'noticed', 'megadeath', 'mr.', 'scruff', 'st.', 'germain', 'tradi...']	['i', 've', 'noticed', 'megadeath', 'a', 'mrs', 'scruff...']	['ve', 'noticed', 'megadeath', 'mrs', 'scruff...']
11	They did not have mayonnaise, forgot our toa...	['they', 'did', 'not', 'have', 'mayonnaise', 'forgot', 'our', 'toas...']	['they', 'did', 'not', 'have', 'mayonnaise', 'a', 'forgot', 'our...']	['mayonnaise', 'forgot', 'toast', 'left', 'ingredient...']
12	It took half an hour to get our check, which was perfe...	['it', 'took', 'half', 'an', 'hour', 'to', 'get', 'our', 'check', 'which', 'was', 'perfe...']	['it', 'took', 'half', 'an', 'hour', 'to', 'get', 'our', 'chec...']	['took', 'half', 'hour', 'get', 'check', 'perfect', ...]
13	The design and atmosphere is just as good.	['the', 'design', 'and', 'atmosphere', 'is', 'just', 'as', 'good', '.']	['the', 'design', 'and', 'atmosphere', 'is', 'just', 'as', ...]	['design', 'atmosphere', 'good']
14	He has visited Thailand and is quite expert on the cui...	['he', 'has', 'visited', 'thailand', 'and', 'is', 'quite', 'expert', 'on', 'th...']	['he', 'has', 'visited', 'thailand', 'and', 'is', 'quite', ...]	['visited', 'thailand', 'quite', 'expert', 'cuisine']
15	I would definitely recommend Mary's and am ma...	['i', 'would', 'definitely', 'recommend', 'mary', 's', 'and', 'am', 'ma...']	['i', 'would', 'definitely', 'recommend', 'mary', 'is', 'and', ...]	['would', 'definitely', 'recommend', 'mary', 'making...']
16	The pizza is the best if you like thin crusted pizz...	['the', 'pizza', 'is', 'the', 'best', 'if', 'you', 'like', 'thin', 'crusted', 'pizz...']	['the', 'pizza', 'is', 'the', 'best', 'if', 'you', 'like', 'thi...']	['pizza', 'best', 'like', 'thin', 'crusted', 'pizza']

Fig. 4.2 Data frame showing different stage of data.

We can see, in column '0' the actual reviews are present, in column 'tokenized_words' the text is tokenized, in the column 'Autocorrect' spelling has been done. And in the last column 'clean_text' it is showing that the stop words are removed from the processed reviews.

4.4 Determined steps for unsupervised method

For unsupervised method we need to manipulate the data so that we can get desired output from the text. So in our approach we have used these techniques.

4.4.1 Counting total Available Words

After removal of stop words from the reviews we have got approx. 20786 words, now we have a list of 20786 words, we have to gain the relation between each and every word. So we moved to the next step.

4.4.2 Identified Total Distinct words

From total 20786 words we have got approx. 4192 words, these words will be our future vertex of the co-occurrence digraph.

Total **distinct words** are : - ['staff', 'horrible', 'us', 'completely', 'fair', 'redeeming', 'factor', 'food', 'average', 'could', 'make', 'deficiencies', 'Teodora', 'uniformly', 'exceptional', 'capable', 'kitchen', 'proudly', 'whip', 'whatever', 'feel', 'like', 'eating', 'whether', 'menu', 'Gabriela', 'personal', 'greet', 'recommends', 'eat', 'go', 'enjoy', 'say', 'get', 'outstanding', 'little', 'perks', 'great', 'overpriced', 'tasty', 'agreed', 'favorite', 'orrechiete', 'sausage', 'chicken', 'usually', 'waiters', 'kind', 'enough', 'split',]

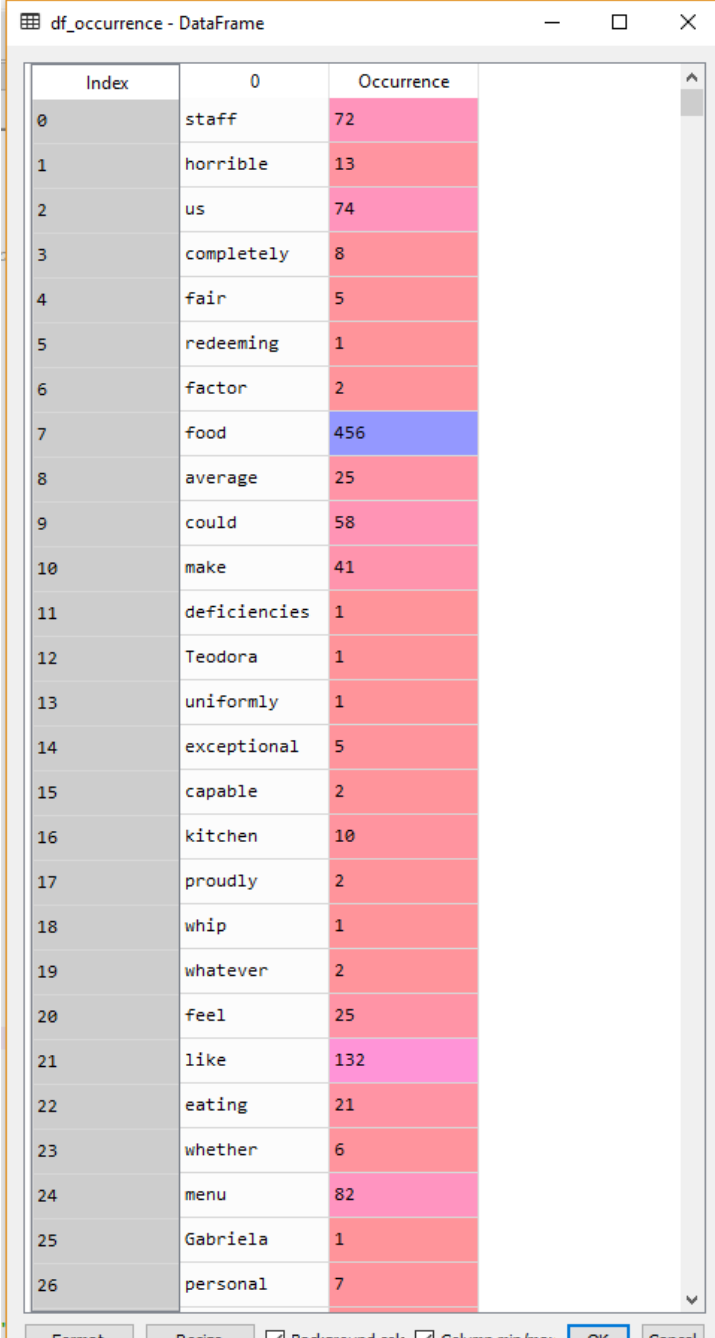


Index	Type	Size	Value
0	str	1	staff
1	str	1	horrible
2	str	1	us
3	str	1	completely
4	str	1	fair
5	str	1	redeeming
6	str	1	factor
7	str	1	food
8	str	1	average
9	str	1	could
10	str	1	make
11	str	1	deficiencies
12	str	1	Teodora
13	str	1	uniformly
14	str	1	exceptional
15	str	1	capable
16	str	1	kitchen
17	str	1	proudly
18	str	1	whip
19	str	1	whatever
20	str	1	feel
21	str	1	like
22	str	1	eating

Fig. 4.3 data frame where all the distinct word stored

4.4.3 Build Occurrence matrix of all words

Total distinct words are 4192 and how many times each word appeared in the data. This information is stored in occurrence matrix X for further evaluation.



Index	0	Occurrence
0	staff	72
1	horrible	13
2	us	74
3	completely	8
4	fair	5
5	redeeming	1
6	factor	2
7	food	456
8	average	25
9	could	58
10	make	41
11	deficiencies	1
12	Teodora	1
13	uniformly	1
14	exceptional	5
15	capable	2
16	kitchen	10
17	proudly	2
18	whip	1
19	whatever	2
20	feel	25
21	like	132
22	eating	21
23	whether	6
24	menu	82
25	Gabriela	1
26	personal	7

Fig. 4.4 Screenshot of occurrence matrix “X”

4.4.4 Build Co-Occurrence matrix of all words

This co-occurrence matrix “Y” shows the relation between word x and y. i.e. how many time a word ‘y’ is occurred after the word ‘x’.

Index	staff	horrible	us	completely	fair	redeeming	factor	food	average	could	make	deficiencies
staff	0	1	5	0	0	0	0	5	0	0	3	0
horrible	0	0	1	0	0	0	0	1	0	1	0	0
us	1	0	0	0	0	0	0	5	0	0	0	0
completely	0	0	0	0	1	1	1	2	1	1	1	1
fair	0	0	0	0	0	1	1	1	1	1	1	1
redeeming	0	0	0	0	0	0	1	1	1	1	1	1
factor	0	0	0	0	0	0	0	1	1	1	1	1
food	2	7	3	0	0	0	1	0	8	5	3	1
average	0	0	1	0	0	0	0	2	0	2	1	1
could	0	1	6	0	0	0	0	2	0	0	2	1
make	1	0	0	0	0	0	0	1	0	2	0	1
deficiencies	0	0	0	0	0	0	0	0	0	0	0	0
Teodora	0	0	0	0	0	0	0	0	0	0	0	0
uniformly	0	0	0	0	0	0	0	0	0	0	0	0
exceptional	0	0	0	0	0	0	0	0	0	0	0	0
capable	0	0	0	0	0	0	0	0	0	0	0	0
kitchen	0	0	1	0	0	0	0	0	0	0	0	0
proudly	0	0	0	0	0	0	0	0	0	0	0	0
whip	0	0	0	0	0	0	0	0	0	0	0	0
whatever	0	0	0	0	0	0	0	0	0	0	0	0
feel	0	0	0	0	0	0	0	2	0	1	0	0
like	0	0	1	0	0	0	1	12	1	1	1	0
eating	0	0	0	0	0	0	0	2	1	0	0	0
whether	0	0	0	0	0	0	0	0	0	0	0	0
menu	0	1	1	0	0	0	0	12	0	1	1	0
Gabriela	0	0	0	0	0	0	0	0	0	0	0	0
personal	0	0	0	0	0	0	0	1	0	0	0	0

Fig. 4.5 Screen shot of Co-occurrence matrix ‘Y’

4.4.5 By using both matrix a graph, G (E, V) is generated

The weight of each edge between two vertices will be decided by $(W_{i,j}) = \frac{Y_{i,j}}{X_j}$, where $Y_{i,j}$ value is taken from co-occurrence matrix, $Y_{i,j}$ means, the number of time j word occurred after the word of i. and X_j value has been taken from occurrence matrix. The mean of X_j is, how many time j word occurred in the entire data/reviews.

Co-occurrence graph

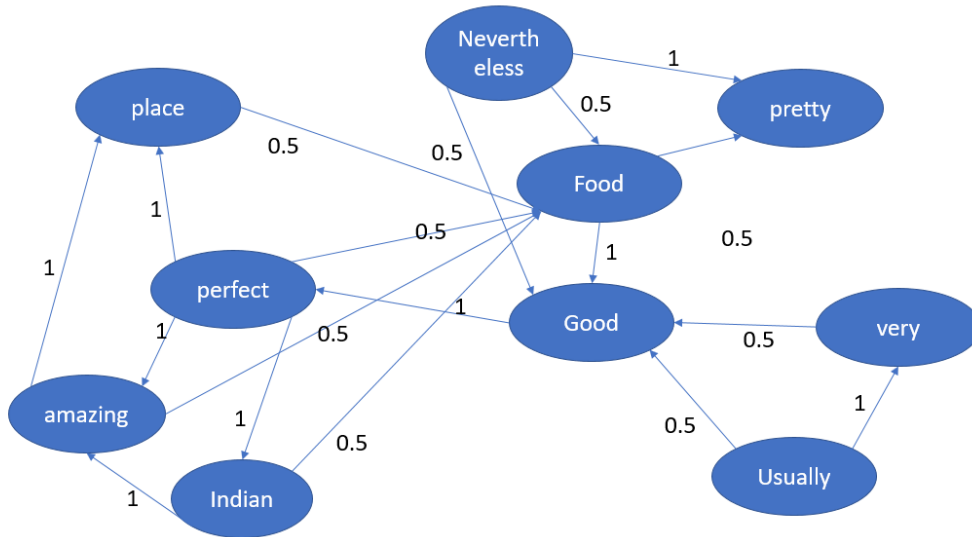


Fig. 4.6 Graph example how the words are connected

4.4.6 Assigned an activation value to each V

To find the most strongly connected words for each aspects we have used an algorithm as it is also used in[1]. Spreading activation algorithm, in this algorithm we have four input \rightarrow root words, graph(V,E), decay factor, threshold value and we will get two output \rightarrow list of fire-words and activation value for each word of graph. For each vertices of graph $V(G) \leftarrow 0$ will be assigned. After that for each $G(V_i)$ that belong to Root Words (R_c) $\leftarrow 1$ will be assigned. Here R_c is Root word for a particular target category, and which ever vertex have the value 1, will be the root for each new sub graph. Rest of the vertices, which are connected to the any root vertex for more than one root vertex of the graph, will get the activation value $A_{vj} \leftarrow \text{Min}(A_{vj} + A_{vi} * W_{ij} * dc, 1)$, if the value of A_{vj} is greater than threshold value i.e. 0.7, the word will be added into the fire word list for the respective aspect category. This process will be done for each aspect category.

Algorithm1: Activation Spreading Algorithm

```

Input root_words  $\rightarrow R_c$ 
Input graph  $\rightarrow G(V,E)$ 
Input decay  $\rightarrow dc=0.9$ 
Input Threshold Value  $t_c \leftarrow 0.1$ 
Output FireWordList  $\rightarrow F$ 
Output Activation Value  $\rightarrow A_v$ 
1 For each  $i \in G(V,E)$  do
2    $A_{v,i} \leftarrow 0$ 
3 For each  $i \in R_c$  do
4    $A_{v,r} \leftarrow 1$ 
5   add  $i$  to  $F$ 
6   end
7  $A_{vj} \leftarrow \text{Min}(A_{vj} + A_{vi} * W_{ij} * dc, 1)$ 
8 For each  $j \in G(V,E)$  do
9   if  $A_{vj} > t_c$  then
10  add  $j$  to  $F$ 
  
```


Index	food	average	could	make	deficiencies	Teodora	uniformly	exceptional	capable	kitchen
completely	0	0	0	0	0	0	0	0	0	0
fair	0	0	0	0	0	0	0	0	0	0
redeeming	0	0	0	0	0	0	0	0	0	0
factor	0	0	0	0	0	0	0	0	0	0
food	1	1	0.387931	0.197561	0.9	0.9	0.9	1	1	0.36
average	0	0	0	0	0	0	0	0	0	0
could	0	0	0	0	0	0	0	0	0	0
make	0	0	0	0	0	0	0	0	0	0
deficiencies	0	0	0	0	0	0	0	0	0	0
Teodora	0	0	0	0	0	0	0	0	0	0

Fig. 4.7 Activation value assigned to each word for food aspect category

Index	guided	allowing	purchase	similarly	svc	jekyll	hyde
words	0	0	0	0	0	0	0
free	0	0	0	0	0	0	0
wine	0	0	0	0	0	0	0
price	0	0.9	0.9	0.9	0	0	0
although	0	0	0	0	0	0	0
poor	0	0	0	0	0	0	0
quantity	0	0	0	0	0	0	0
also	0	0	0	0	0	0	0

Fig. 4.8 Activation value assigned to each word for price aspect category

Index	alluring	five	star	featuring	sommelier	complicated	maze	captain	foods	mommy	person
gotten	0	0	0	0	0	0	0	0	0	0	0
better	0	0	0	0	0	0	0	0	0	0	0
months	0	0	0	0	0	0	0	0	0	0	0
perhaps	0	0	0	0	0	0	0	0	0	0	0
queens	0	0	0	0	0	0	0	0	0	0	0
biggest	0	0	0	0	0	0	0	0	0	0	0
secret	0	0	0	0	0	0	0	0	0	0	0
may	0	0	0	0	0	0	0	0	0	0	0
bit	0	0	0	0	0	0	0	0	0	0	0
packed	0	0	0	0	0	0	0	0	0	0	0
French	0	0	0	0	0	0	0	0	0	0	0
area	9	0.18	0.3	0.9	0.45	0.9	0.9	0.9	0.45	0	0
much	0	0	0	0	0	0	0	0	0	0	0

Fig. 4.9 Activation value assigned to each word for ambience aspect category

Index	impeccable	unobtrusive	present	attend	needs	ess-a-bagel	either	
owner	0	0	0	0	0	0	0	0
super	0	0	0	0	0	0	0	0
friendly	0	0	0	0	0	0	0	0
service	1	0.45	0.9	0.9	0.225	0	0.0692308	0
value	0	0	0	0	0	0	0	0
joint	0	0	0	0	0	0	0	0
fan	0	0	0	0	0	0	0	0
ent	0	0	0	0	0	0	0	0

Fig. 4.10 Activation value assigned to each word for service aspect category

4.4.7 Finding fire words for each aspect by using root words for each category

Fire words for each category mean, there are lots of words are occurred with the root words in each category, in the entire data, to get most closed words from root words, we have set a threshold value i.e. 0.7, if the activation value is greater or equal to threshold value, the same word will be added into the fire word list for the particular aspect category.

fired_word_ambience	list	258	['ambience', 'floor', 'place', 'crowd', 'block', 'ambient', 'atmospher ...
fired_word_food	list	748	['bagels', 'cake', 'potato', 'onion', 'garlic', 'burger', 'dishes', 'd ...
fired_word_price	list	86	['deal', 'high', 'prices', 'priced', 'budget', 'afford', 'cheap', 'pri ...
fired_word_services	list	267	['menu', 'server', 'waitress', 'order', 'staff', 'deliver', 'friend', ...

Fig.4.11 fire word list for each category

In Fig. 4.11 We have got 258 words for ambience, 748 words for food category, for service there are 267 words and very less words are come under price category only 86. The above words will be used in the association rule. So that we can assign the category to particular reviews, where root words are not mentioned.

4.5 Association Rule

Association rule algorithm will be used to get aspect category of each sentence and for each aspect category. In this we have four inputs for each aspect category (Review, Graph(V,E), Threshold Value \rightarrow Atc and FireWordList \rightarrow Fc and one output Aspect Category for the text \rightarrow T. 1) For each

word in sentence T_i . 2) If T_i is available in the graph and its value is 1 than 3) The sentence will get the aspect category for respective input aspect category graph. 4) if T_i available in the graph and its value is not 1, then Find the word in fireword list and if it is available in the fire word of the particular aspect category graph. 5) sum-up the activation values 6) Threshold value for aspect category is ($Atc \rightarrow 1.0$) if sum-up value is greater than or equal to Atc , 7) then the text will get the respective aspect category 8) if less than the value of Atc . 9) the review will get default aspect category.

Association Rule Algorithm 2 (ARA)

```

Input Review - T
Input graph-  $G(V,E) \in C$ 
Input Threshold Value -  $Atc = 1.0$ 
Input FireWordList -  $Fc$ 
Output  $Ac$  for each  $T$ 
1 For each  $i \in T$  do
2     if  $i == 1$  in  $V$ 
3         [ $T = category\ c$ ]
4     if  $i$  in  $Fc$ 
5          $Av,i += Av,i$ 
6 If  $Av,i \geq Atc$ 
7     [ $T = category\ c$ ]
8 else
9     [ $T = category\ Default$ ]

```

4.6 Sentiment Analysis

For sentiment analysis, we have used sentiwordnet3.0, to get the sentiment score for each and every word. The following steps are taken to solve this problem.

4.6.1 POS Tagging

After pre-processing steps, we have got clean words of every sentence, these words are tagged by using natural language toolkit- part of speech tags. There total 45 tags available in natural language toolkit.

Tag : Tag name

- \$: dollar
- (: opening parenthesis
-): closing parenthesis
- ,: comma
- .: sentence terminator
- ": closing quotation mark
- : dash
- :: colon or ellipsis
- ``: opening quotation mark
- CC: conjunction, coordinating
- CD: numeral, cardinal
- DT: determiner
- EX: existential there
- FW: foreign word
- IN: preposition or conjunction, subordinating
- JJ: adjective or numeral, ordinal
- JJR: adjective, comparative
- JJS: adjective, superlative

LS: list item marker
 MD: modal auxiliary
 NN: noun, common, singular or mass
 NNP: noun, proper, singular
 NNPS: noun, proper, plural
 NNS: noun, common, plural
 PDT: pre-determiner
 POS: genitive marker
 PRP\$: pronoun, possessive
 PRP: pronoun, personal
 RB: adverb
 RBR: adverb, comparative
 RBS: adverb, superlative
 RP: particle
 SYM: symbol
 TO: "to" as preposition or infinitive marker
 UH: interjection
 VB: verb, base form
 VBD: verb, past tense
 VBG: verb, present participle or gerund
 VBN: verb, past participle
 VBP: verb, present tense, not 3rd person singular
 VBZ: verb, present tense, 3rd person singular
 WDT: WH-determiner
 WP\$: WH-pronoun, possessive
 WP: WH-pronoun
 WRB: Wh-adverb

The problem occurred, when we got analyzed sentiwordnet3.0, in this corpus we have only 4 tags. That are used to identified words. The following tags are there.

a :- adjectives
 n :- nouns
 r :- adverbs
 v :- verbs

After performing part of speech tagging on the clean text, the output was:-

[('Best', 'JJS'), ('warm', 'NN'), ('vibe', 'NN'), ('owner', 'NN'), ('super', 'VBD'), ('friendly', 'JJ'), ('service', 'NN'), ('fast', 'NN')]
 [('food', 'NN'), ('delicious', 'NNS'), ('come', 'VB'), ('empty', 'JJ'), ('stomach', 'NN')]
 [('Go', 'NNP'), ('romantic', 'JJ'), ('dinner', 'NN'), ('out', 'RP'), ('wow', 'JJ'), ('dining', 'VBG'), ('experience', 'NN')]
 [('menu', 'NN'), ('limited', 'VBD'), ('dishes', 'NNS'), ('excellent', 'JJ')]

4.6.2 Getting Sentiment score

To overcome mapping problem, we got the idea from [6], the blogger posted a logic, where we consider only four major tags from natural language toolkit and getting sentiment score of words.

Natural language toolkit tags	=	Sentiwordnet3.0 tags
'NN'	=	'n'
'VB'	=	'v'
'JJ'	=	'a'
'RB'	=	'r'

If 'NN' letters found in part of speech tags then all the word will be comes under 'n'

If 'VB' letters found in part of speech tags then all the word will be comes under 'v'

If 'JJ' letters found in part of speech tags then all the word will be comes under 'a'

If 'RB' letters found in part of speech tags then all the word will be comes under 'r'

By giving this logic,

For Tag 'NN'

NN: noun, common, singular or mass

NNP: noun, proper, singular

NNPS: noun, proper, plural

NNS: noun, common, plural

Fog Tag 'VB'

VB: verb, base form

VBD: verb, past tense

VBG: verb, present participle or gerund

VCN: verb, past participle

VBP: verb, present tense, not 3rd person singular

For Tag 'JJ'

JJ: adjective or numeral, ordinal

JJR: adjective, comparative

JJS: adjective, superlative

For Tag 'RB'

RB: adverb

RBR: adverb, comparative

RBS: adverb, superlative

According to above logic, all 15 tags are covered by these 4 tags.

The output of this technique is for first example sentence

```
<best.a.01: PosScore=0.75 NegScore=0.0>
<vibration.n.04: PosScore=0.0 NegScore=0.125>
<owner.n.01: PosScore=0.0 NegScore=0.0>
<friendly.a.01: PosScore=0.25 NegScore=0.125>
<service.n.01: PosScore=0.0 NegScore=0.0>
<fast.n.01: PosScore=0.0 NegScore=0.0>
```

Using part of speech tag, each word available in the aspect, will be searched from sentiwordnet3.0 and the positive and negative score are called from the sentiwordnet3.0. that will be sum-up the positive and negative score for each aspect available in the sentence.

By using these two parameter {positive score and negative score} we manage to get all four sentiment category by normalizing the sentiment score (Ss)of each aspect, $Ss = \sum_{i=0}^n Ps, i - Ns, i$, where i for first word and n is for number of words in the aspect, Ps,i = positive score of word i and Ns,i= negative score of word i. to normalized the score = (Ss/n)

In the last it will be assigned to the particular review with its aspect category; as shown in the fig.4.12.

	Sentences	Category	Sentiment
0	['We ordered the special, grilled branzino, that was so infused with bone, it was difficult to eat.']	food	negative
1	['The wait staff is friendly, and the food has gotten better and better!']	service	positive
2	['The wait staff is friendly, and the food has gotten better and better!']	food	positive
3	['This place, which is only a few months old, is perhaps Queens' biggest secret!']	ambience	neutral
4	['It may be a bit packed on weekends, but the vibe is good and it is the best French food you will find in the area.']	food	positive
5	['It may be a bit packed on weekends, but the vibe is good and it is the best French food you will find in the area.']	ambience	negative
6	['This place is so much fun.']	ambience	neutral
7	['I was glad I did.']	anecdotes/miscellaneous	neutral
8	['Right off the L in Brooklyn this is a nice cozy place with good pizza.']	ambience	positive
9	['Right off the L in Brooklyn this is a nice cozy place with good pizza.']	food	positive

Fig. 4.12 Example output

In fig. 4.12 index 0 has only one aspect that is 'food' and its sentiment is negative, while in index 1 and 2 have the same review but aspect is different. The half review "The wait staff is friendly" it is related to aspect 'service', while next part of the review "and the food has gotten better and better!" has 'food' aspect and its sentiment is 'positive'. Let see another example in fig 4.12. index number 4 and 5, "It may be a nit packed on weekends" this is related to 'ambience' and its sentiment is 'negative'

while other part of review “but the vibe is good and it is the best French food you will find in the area.” Has ‘food’ aspect category and sentiment are positive.

The task to find total aspect available in the review it perfectly done. The sentiment is taken from sentiwordnet3.0, it is for English language, we can use another way or technique like ‘texblob’ library to get the sentiment of the available.

Limitations of the proposed approach is range of root words. Root words are basically, synonyms and some extra words, that are used in general English language in any restaurant for target aspect, If we are able to list more words under root words, it would lead to get more accurate fire words for the same aspect category, then it is very easy to find aspect of the data, fire words are nothing but the words, which were frequently occurred with the root words. It will help the machine, when any root word is not there but fire words are there then the sum-up the value of each fire word available in the review, will assign the aspect category as implicitly.

4.7 Software/Hardware requirement

For software requirement, the work is implemented on Windows 10 environment with python3.6 application software. The spider python tool under anaconda project navigator software.

Minimum hardware requirement, Intel i5 processor with 2.0 GHz clock speed, minimum hard disk space required 10GB free in the system. As far as concern about random access memory minimum requirement is DDR3 2GB and the best will be 8GB.

CHAPTER 5

RESULT AND ANALYSIS

5.1 Parameters to evaluate the result

- Sentiment analysis machine result are evaluated on accuracy in terms of Positive, Negative, Neutral, and Conflict sentences. We have evaluated the result on accuracy in terms of aspect category detection in the review. Weather the aspect is correctly identified or not. Following parameters are used to analyze the result.

$$\text{ACCURACY \%} = (\text{TI}/\text{TI}+\text{FI}) * 100$$

Where TI = True Identified, FI = False Identified.

5.2 Evaluation

We have total 3044 reviews, but after applying the proposed method it become 3566, the reason of review increment is, a review can have multiple aspect, the result is in Table 5.1, in the second column Lablled data is given by the SemEval 2014 and next in column the result of proposed approach

Table 5.1:- Total Reviews :- 3044					
<i>Categories</i>	Labelled Data	Proposed Approach	TRUE	FALSE	% Accuracy
<i>Food</i>	1233	1097	991	106	90.34
<i>Price</i>	319	244	218	26	89.34
<i>Ambience</i>	432	414	389	25	93.96
<i>Service</i>	597	418	398	20	95.22
<i>Anecdotes/Miscellaneous</i>	1133	1393	1067	326	76.6
Total	3714	3566	3063	503	85.89
Accuracy% =	Data Generated = 96.01		83.58		82.47

for each aspect category. TRUE and FALSE columns are representing the truly and falsely identified by the proposed approach, the proposed approach is performing better than many unsupervised methods, that are used to do the same task.

Data Generated by the proposed approach is 96.01% with respect to labelled data, and the error % is 16.42% by considering (False/True)*100 then accuracy will be 83.58%. The accuracy of the proposed approach over the labelled data is 82.47 =(total true /total labelled data)*100

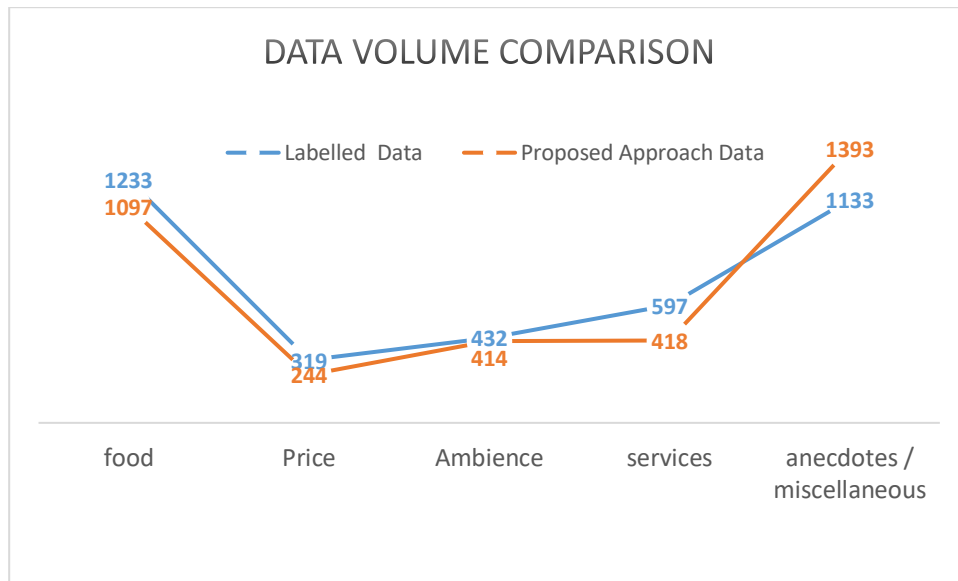


Fig. 5.1 Data volume comparison of labelled data and proposed approach result data

In Fig. 5.1 the orange line is representing the data volume in each aspect category, and blue line is representing actual labelled data, here we can see food, price, ambience and services have less number of reviews but in anecdotes/miscellaneous we have more number of reviews than labelled data. It is because we have less number of reviews in other category, all those reviews, which are not identified by the proposed approach, will transferred into the default category. And default category is anecdotes/miscellaneous.

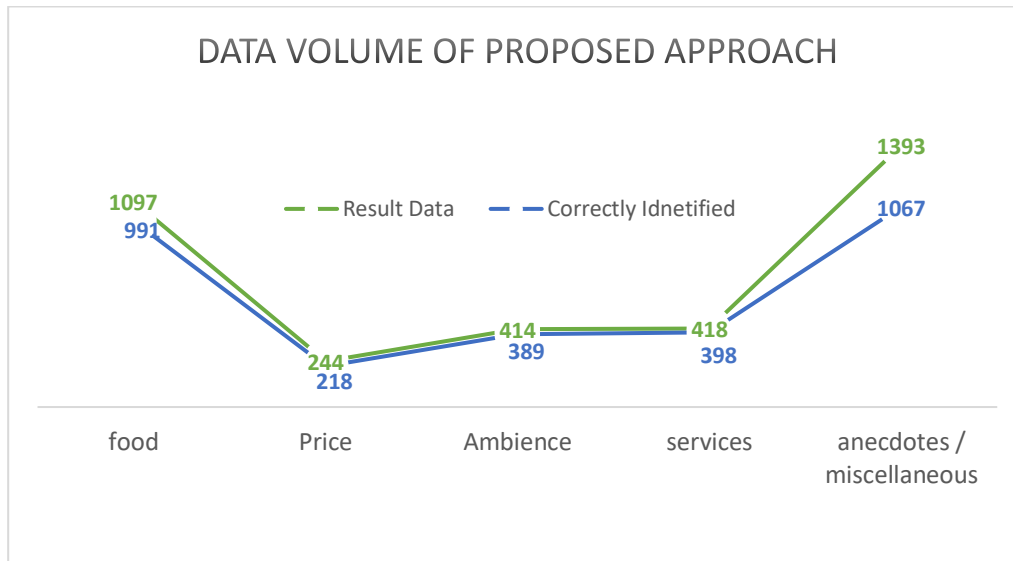


Fig. 5.2 Data volume of proposed approach

In Fig. 5.2 in blue lines are showing correctly identified reviews and green line represents for proposed approach data volume. In proposed data volume we have truly and falsely identified both.

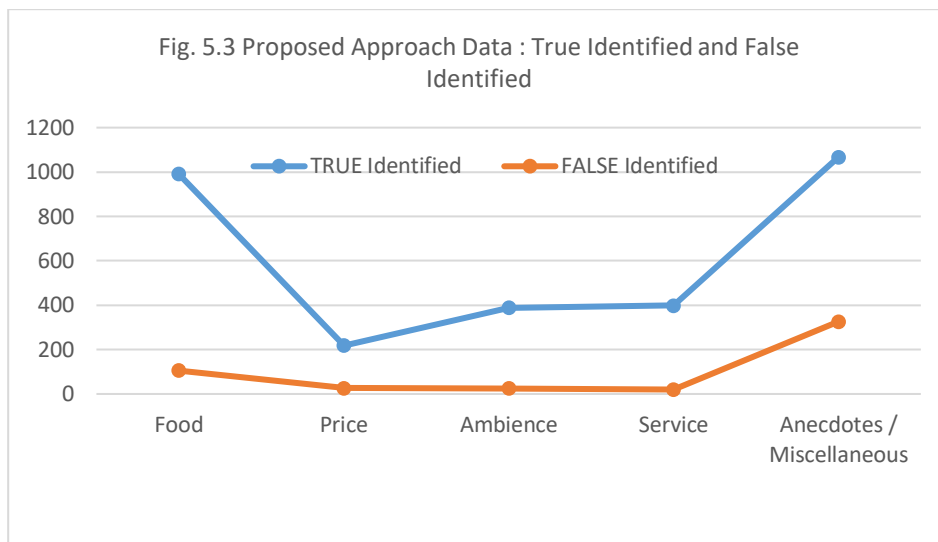


Fig. 5.3 Proposed approach result.

TRUE Identified data is indicated by blue colour line and FALSE Identified data is indicated by orange colour line in the fig.5.3, all aspect categories are listed and their respective result also. True Identified data meaning is the data is correctly identified in the same aspect category by the approach. And False Identified meaning is the data is listed under the aspect category is not true. It means the data is false.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

How we landed to this work, well we were searching for a problem where we can create a model that will be useful for general public, which will help to take decision for purchasing news product or services and it will give them a brief explanation of the product/service so that they need not to waste much time to take decision. Finally, we reached to this SemEval2014 task. In this work we have to do aspect-based sentiment analysis, for the general public and to the owner of the restaurant as well. This will help both parties. For the owner of the restaurant, it will help to find their negative and positive aspects. And they can improve their services/product quality by knowing the status of the respective aspect. Here we have four major aspect [service, food, ambience, price] and one default category [Anecdotes/Miscellaneous] if a review does not have any aspect then it will go to default category.

We have introduced root words and fire words list category to solve the problem in unsupervised method. Using root words are able to get implicit [1] aspect category detection, with direct aspect category detection for any review. Some reviews do not have direct aspect, but the words are occurred with direct aspect, we call them in fire word list, and it is created for each aspect category. When a review does not have root word in it. Our approach search fire words in the same reviews, and it works great. We are able to manage 85.89% accuracy.

In this approach we have two assumptions, threshold value for fire words $t_c=0.7$ and threshold value for aspect $A_{t_c}=1.0$ both values are on assumptions based. We need to be very careful to put these values into the system. If the we increase the value of t_c , it will loss the fire word available in the reviews. That will causes to lost the implicit aspect available in the review. In the same manner if we increase the value of A_{t_c} , it will losses the implicit aspect available in the reviews, because in the association rule we are sum-up the values of the fire word available in the reviews. The sum -up value is less then the value of A_{t_c} , the particular reviews will get default aspect. Not the right one. So t_c value is for fire words and A_{t_c} value is for aspect hidden in the review. Both values are decided after observing many results.

There are total 3044 sentences in the data set. The set of files is downloaded from

<http://alt.qcri.org/semEval2014/task4/index.php?id=data-and-tools>, and baseline method is also available to download. The description of the data is, out of 3044 reviews food = 1233, anecdotes/miscellaneous= 1113, Service=597, Ambience= 432 and Price = 319.

The result by the proposed approach for correct identified in food =991, price = 218, service = 398, ambience = 389 and for anecdotes=1067. total aspect was 3566 created by the proposed approach and out of 3566 the correctly identified aspects are 3063. It gives a good accuracy 85.89%

6.2 Future Work

While calculating the sentiment, the sentiment is taken from sentiwordnet3.0, it is for English language only, we will use different techniques and different library like TextBlob to get the sentiment.

This work is done only in English language, the work can be extended for Indian languages, we have to study more research paper on sentiment analysis on Indian language, and aspect category detection for Indian language, as far as concerned about my knowledge, there are not any sentiwordnet3.0 and Wordnet for Indian language, so we need to implement the work from the scratch. It would be a great experience.

REFERENCES

- [1] Kim Schouten, Onne van der Weijde, Flavius Frasincar, and Rommert Dekker, “Supervised and Unsupervised Aspect Category Detection for Sentiment Analysis with Co-occurrence Data” IEEE Transactions on Cybernetics, Volume: PP, Issue: 99, pp. 1 – 13, April 2017
- [2] S. Kiritchenko, X. Zhu, C. Cherry, and S. M. Mohammad, “NRC Canada-2014: Detecting aspects and sentiment in customer reviews,” in Proc. 8th Int. Workshop Semantic Eval. (SemEval), Dublin, Ireland, pp. 437–442, 2014
- [3] M. Pontiki et al., “SemEval-2014 Task 4: Aspect based sentiment analysis,” in Proc. 8th Int. Workshop Semantic Eval. (SemEval), Dublin, Ireland, pp. 27–35, 2014
- [4] K. Schouten and F. Frasincar, “Survey on aspect-level sentiment analysis,” IEEE Transactions Knowledge and Data Engineering, vol. 28, no. 3, pp. 813–830, Mar. 2016.