Qualitative Analysis For Well Being Management

Prepared by

M P Fousiya

Declaration of Authorship

I, M.P. Fousiya, declare that this thesis titled, 'Qualitative Analysis For Well Being Management' and the work presented in it, is my own. I confirm that this work, which is being submitted for assessment, is my own and is expressed in my own words. Any uses of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at the point of their use and a list of those references is also included.

Signed: *M.P.Fousiya*

Date: *Thursday 15th August 2019*

"If you can't explain it simply, you don't understand it well enough."

Albert Einstein

Abstract

Addressing the overall well-being of underserved communities has been a missing part of puzzle in Big Data Science. Every individuals should have an equal opportunity to reach their full potential in every sphere of their life, but reality is far behind this. Since so many individuals lack the opportunity to improve their lifestyle, the expulsion of health disparity has been emerged as a major worldwide public health objective. ***This project focuses on the analysis of the reasons for the prevalence of these problems in a rural community in India and thus preventing those at low cost and high speed.***

To unveil this, the content of the text data that has been obtained by conducting surveys and informal interviews, are analyzed using two different approaches: (i) Using Word2Vec which helps in understanding the relevant related keywords from the data and (ii) A Text Classification approach using a simple Artificial Neural Network (ANN), whose result is then visualised using a WordCloud. Textual data contains abundant qualitative information that are not easy to undergo a statistical analysis unlike quantitative data. ***The findings says that, for our data, the combination of Text Classification with Word2Vec provides more efficient results than using those modeling approaches individually, as it can find niche topics and associated vocabularies from the interview data.*** This project report provides an overview on the qualitative research, the techniques that are used to analyze our textual data for obtaining meaningful information, the limitations of those approaches and also suggests some possible ways for further study.

Acknowledgements

Thank you, Dr.Muhammed Hamdan for supervising , helping and guiding me throughout the duration of this dissertation.

Thank you, Dr.Ali M S Zalzala from Community Tracks, for giving me the opportunity to work with them.

Thank you, Mrs. Kinnari Bhasker, for your help and support in translating the Gujarati videos into English language.

I would also like to thank my family for their everlasting love and support. I'm grateful to them for believing in me, even at times when I myself didn't.

Contents

List of Figures

Abbreviations

ANN	**A**rtificial **N**eural **N**etwork
CBOW	**C**ontinuous **B**ag **O**f **W**ords
EDA	**E**xploratory **D**ata **A**nalysis
FFNNLM	**F**eed **F**orward **N**eural **N**etwork **L**anguage **M**odel
IoT	**I**nternet **o**f **T**hings
LSTM	**L**ong **S**hort **T**erm **M**emory
NLP	**N**atural **L**anguage **P**rocessing
NLTK	**N**atural **L**anguage **T**ool **K**it
POS	**P**arts **O**f **S**peech
RNN	**R**ecurrent **N**eural **N**etwork
RNNLM	**R**ecurrent **N**eural **N**etwork **L**anguage **M**odel
SDG	**S**ustainable **D**evelopment **G**oals
WHO	**W**orld **H**ealth **O**rganization

Dedicated to my family and friends, thank you for all your support…

Chapter 1

Introduction

1.1 Overview

Addressing the well-being among underserved communities has been a missing part of puzzle in Big Data Science. Technology has made an enormous expansion and it has helped to improve the lifestyle of the global population. However, not all people are having the same access to this technology. Majority of the people who are still not able to manage a good lifestyle are from the underserved communities, the community that is being made up of individuals with low socioeconomic status and poor literacy [Kreps, 2005]. This lack of progress suggests that new approaches are inevitable, if we want to achieve a meaningful, impartial, and enduring results.

1.2 Motivation

According to the recent report from the World Bank and WHO [WHO, 2017], about half of the world population lacks access to essential health services and 100 million are pushed into extreme poverty because of the health expenses. The report also mentions that even though, in 21st century, there is an increase in the number of people who have access to the health services, the progress is still uneven and the people who are being affected by these are mainly from the rural and underserved communities.

Nowadays, the business management for the mass markets like healthcare, pharmaceutical and insurance, are mostly driven and influenced by big data, cloud and mobile technologies. Even though there had occurred many technological advancements in the healthcare sector, in the form of Internet of Things (IoT) and wearable devices, it still remains formidable because of its lack of reach to the masses, especially to the rural and underserved communities which are being characterized by low-income and low-literacy. Technologies would become feasible only when it is designed according to the needs and specifications of the end users within a sustainable social business framework.

The United Nations Sustainable Development Goals (SDG) are a call for action by all countries - poor, rich and middle income - to achieve a better and more sustainable future for all [UnitedNations, 2015]. They address the challenges that are being faced by the people from all over the world, including those that are related to inequality, poverty, prosperity, environmental degradation, climate, peace and justice. Their target is to achieve all of their goals by 2030 without leaving anyone behind. One of their goal, **"Good Health and Well Being"**, is to ensure healthy lives and promote well being for all in order to achieve sustainable development.

This project focuses on employing machine learning to address the problems, in terms of overall well-being, that is being faced by the rural and underserved communities in Ahmedabad, India, and to identify the reasons behind that.

1.3 Aim

Using machine learning in qualitative research has its own challenges, like inaccurate or inadequate data, data preprocessing and identifying the right technique for dealing with the qualitative data. Considering the above challenges our requirement is to develop an algorithm that can efficiently identify the challenges and the reasons that are affecting their overall well-being, by analysing the textual interview data.

This project explores how different algorithms like Artificial Neural Networks(ANN) and Word2Vec performs in understanding the themes or extracting useful knowledge from an unstructured interview data.

1.4 Objectives

This project compares different machine learning algorithms that takes the unstructured textual data as its input, performs thematic analysis and extracts useful information from those data. The objectives of this project are as follows:

- Translate and transcribe the data into English language, as the interview data is in Gujarati and Hindi language.
- Prepare the training data.

- Prepare the transcribed interview data to make it ready for analysis.
- Performing advanced simulations on the resulting data:
  - Using Word2Vec : Convert each and every word in the data into vector format which will then help to identify the semantic similarities between them.
  - Using a Neural Network for text classification : Classify the responses based on the questions which is already been categorized into different groups and identify the codewords from each groups using a Word Cloud.
- Compare the developed models and identify the approach that would help to define our theme **"Well Being Management"** really well.

Chapter 2

Background and Literature Review

2.1 Introduction

This chapter provides an overview about the qualitative research methodology and explores why is it appropriate for our research. This is then followed by a brief description on the general steps involved in Text Analytics, which is also called as Natural Language Processing (NLP) which helps to deal with the unstructured textual data.

2.2 Qualitative Research

Research is an organised, systematic and disciplined approach to answer the questions about our observations and experiences about the world. It is a structured approach to gather and interpret information that allows us to understand, theorise about or explain experience.

Doing only the quantitative research might make the effort of understanding about the overall well-being of the population as void, since it emphasises only on the objective measurements without having more in depth and systematic examination of the underlying cause. On the other hand, qualitative research focuses on generating meanings and understandings through rich description obtained by diving deeper into the problem [Rose Marie Nieswiadomy, 2012]. Most researchers who rely on qualitative approach would agree with the observation that the numbers obtained from quantitative research are impressive, but unfortunately, it conceals more than they reveal [Suter, 2012].Typically the qualitative approach can address several problems that arises from different philosophical view of the world, by using different methods and design. The design characteristics of this approach is flexible, evolving and emergent in nature. But at the same time it should be emphasised that the qualitative approach, in no way suggests that it is less disciplined or easier to implement.

Steps in qualitative research are [Rose Marie Nieswiadomy, 2012]:

- Identify the Problem of the Study : In a qualitative study the problem to be examined indicate the general nature of the phenomenon need to be studied and the group or community that will be studied.
- State the Purpose and Select the Research Design : The research design depends on the phenomenon needed to be solved. It can be grounded theory approach, action research approach, etc.
- Select the Sample : One of the criticism of qualitative study is that the sample size is smaller when compared to quantitative study and there are no set rules for this. Sandelowski's (1995b) article clearly states that the quality of the information obtained from the participants is more important than the quantity of the data [Sandeloweski, 1995].
- Gain Entry to the Research Site : Qualitative researchers need to conduct the research from the place where the participants of the research are living or working.
- Protect the Rights of Participants : Due to the close relation with the researcher, the research participants may disclose their very personal or private information with the researcher. Therefore ethical issues is very important in this case.
- Collect the Data : Qualitative data collection approaches may vary by using unstructured or semi-structured techniques, which includes methods like conducting individual or group interviews, group discussions or participant observations.
- Coding : After collecting the qualitative data (we may have text, audio or visual representation of the data like drawings, photos, etc.), the next process is to do coding. Coding is the process of symbolically assigning an evocative attribute (code) for a portion of qualitative data [Elliott, 2018]. During coding we look at the patterns of similarity or differences in our data and assign codes according to these identified patterns.
- Analyze and Interpret the Data : Unlike quantitative data, analysing and interpreting the qualitative data is much more challenging due to its unstructured format.

- Communicate and Utilize the Study Results : Utilize the obtained study results to make decisions or recommendations for solving the problem that is identified during the first step of qualitative research.

There are different approaches [Grove et al., 2012, Rose Marie Nieswiadomy, 2012] for conducting a qualitative research and the purpose of our project determines which approach do we need to use in our research design. Our survey (interviews, observations, and focus groups) is conducted using a grounded theory qualitative research approach, as it focuses on developing a new theory or expanding an existing theory. It uses constant comparative method to analyse the qualitative data. The survey was conducted with selected stakeholders who are from within or are associated with the target community and without having any prior knowledge about the investigations. The theory, which is grounded in the data, is developed once the data is collected and analyzed. When theory generation is more critical than theory testing, constructs and concepts are grounded in the data and the hypotheses are tested as they arise from the research. The theory which we would implement will be closely associated with a knowledge translation framework, that aims to bridge the significant gap between what is known (evidence) and what is implemented by the stakeholders (practice) [Manns, 2015].

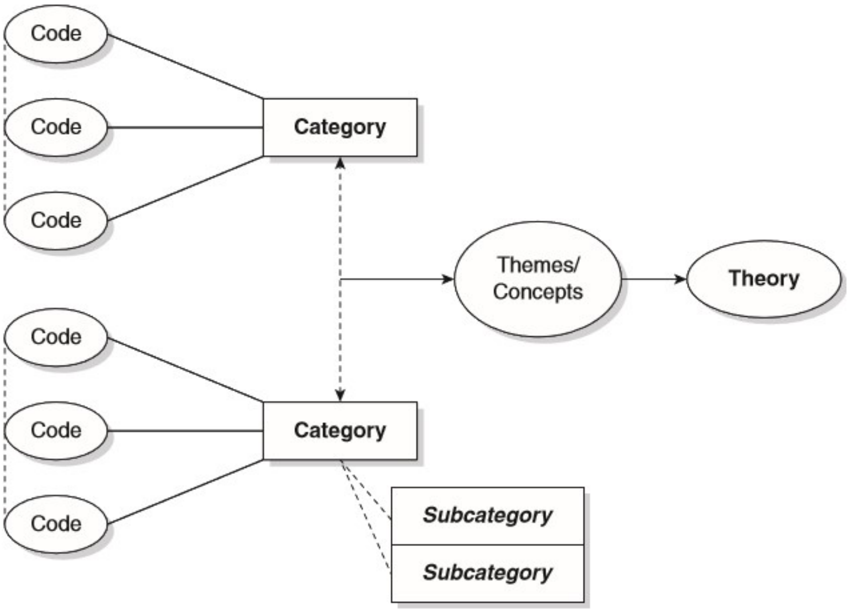


Figure 2.1: A streamlined codes-to-theory model for qualitative inquiry .[Saldaña, 2015]

The above figure shows the visual representation of how the codes and categories obtained from the provided data can lead to theory formation.

2.3 Text Analytics

Text Analytics involves a set of techniques and approaches that is being carried out for bringing the unstructured textual content to a point where it is represented as structured data and then mined for finding insights, trends or patterns. It can be also described as a set of statistical, linguistic and machine learning methodologies that helps to model and structure the content of the textual sources for exploratory data analysis (EDA), business intelligence systems or for research purposes. Text Analytics helps to save our time, by efficiently answering different queries we are having within the text data. The book 'Practical Text Analytics: Maximizing the Value of Text Data' [Anandarajan et al., 2018] introduces and explains about text analytics as a valuable method for converting qualitative data into quantitative data for deriving insights from it.

2.3.1 History

Natural Language Processing (NLP) has a much longer history which has been started in 1950s with the objective of making the computers understand the human language. During this decade, the content analysis in social sciences has emerged as a means for analysing the contents, including text and other media. In late 1980s and early 1990s Latent Semantic Analysis (LSA) were introduced as a method for dimension reduction and identifying the latent factor in the text [Deerwester et al., 1990]. Data Mining has been emerged and machine learning has gained more importance in 1990s. The inclination towards the Big Data Analytics has gained more importance by 2010 and the foundational concepts that were used till now, has been started to apply for this as well. Figure 2.1 shows the timeline for text analytics.

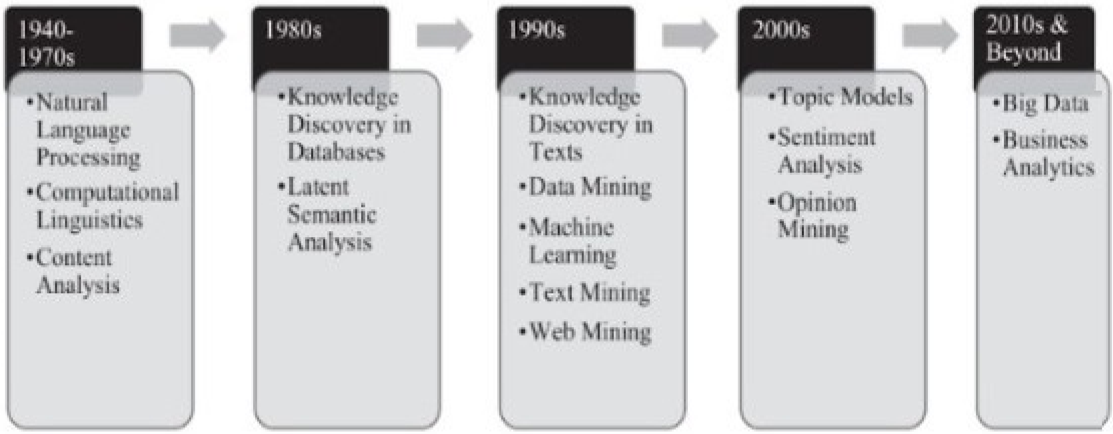


FIGURE 2.2: Timeline for Text Analytics.[Anandarajan et al., 2018]

The Text Analytics steps can be broadly divided into four:(i) Planning, (ii) Text Preparation, (iii) Analysis and (iv) Reporting. Following is a brief overview about these steps.

2.3.2 Planning

The planning step determines the quality of the entire process of text analytics because here we are setting the foundation for the process of analysis. Qualitative data analysis can be very frustrating when we are dealing with very large amount of data, but following the right processes will not only help to carry out the process effectively but also can have fun while doing it. The initial planning considerations for the text analytics approach includes [Anandarajan et al., 2018] :

- Drivers : The text analytics process depends on the initial motivation of our project. There are three drivers for content analysis applications:(1) the text,(2) the problem and (3) the method.
- Objectives : It is important to have an accurate idea about the objectives before starting the project. These well defined objectives for the research gives good understanding about the data, methods and resources that are needed for the completion of our analysis.
- Choosing Right Data : Choosing the right data is really very important, especially when the analysis is driven by the problems, as the data need to be in exact sync with our objective.

The planning process involves several major tasks like framing the problem, data generation and making decisions about analysis which will help the analyst in reaching the goal in a more efficient way.

- Problem Framing : This involves the identification and description of the problem and deciding which coding approach we need to follow to start our analysis. Coding is the process of indexing or tagging unstructured text data to provide an overview of the disparate data and this allow the researcher to have a sense out of them, in connection with the research question. Coding is very important because, *"Text data are dense data, and it takes a long time to go through them and make sense of them"* [Creswell, 2015]. Two different types of coding approaches are:
  - Inductive Coding Approach : If our goal is to deduce something new from the text data, we need to use the inductive or data driven coding approach

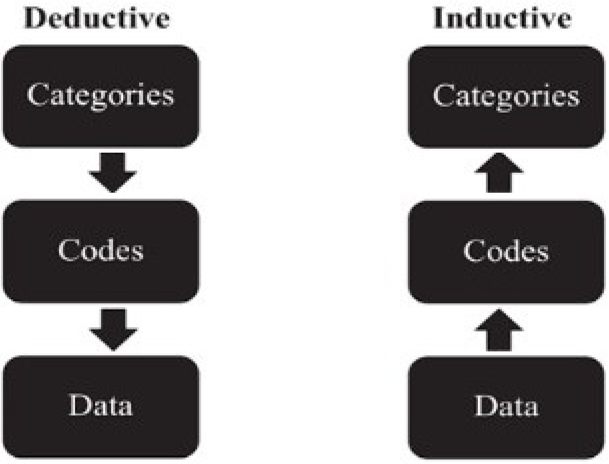


FIGURE 2.3: Inductive and Deductive Coding Approach.[Anandarajan et al., 2018]

by keeping the external influences like background and theory to a minimum. Grounded theory follows an inductive coding approach as the theory is grounded in the underlying data and the researcher will not be having any prior knowledge about that theory.

- Deductive Coding Approach : On the other hand, if we need to expand the current knowledge base, we can use deductive or theory driven coding approach which uses existing data, theory and information for performing hypothesis testing and model building.

- Data Decision : While doing text analytics it is important to decide on the granularity of the unit of text for analysis. The unit can be characters, words, sentences or documents and it depends on the purpose of the project. The result of text data analysis depends highly on the scope chosen and quality of the data given as input. Figure 2.4 shows some important dimensions of quality for text data [Pipino et al., 2002].

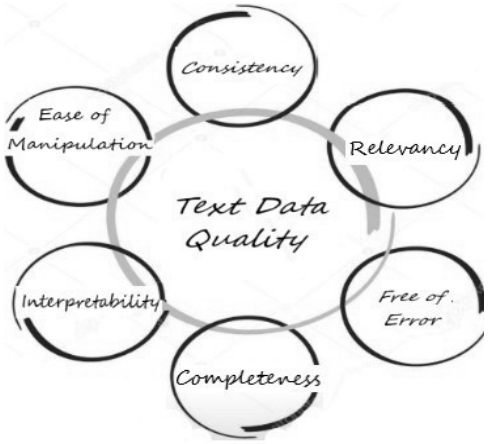


FIGURE 2.4: Quality Dimensions of Text Data

- Method and Implementation Selection : The selection of algorithm is highly dependant on the purpose of our project.

2.3.3 Text Preparation

By the end of the planning stage, the goal of our analysis should have been well defined. Next we have to proceed with the process of text preparation. It is the process of making the text data ready for analysis. There are a variety of methods to restructure and clean the unstructured textual data. It includes unitization and tokenization, standardizing and cleaning by removing stop words, stemming or lemmatization, parts-of speech tagging, named-entity recognition and then final representation in which we need to convert the text data into a format that is understandable by a computational model. These step comprises the majority of the tasks involved in Natural Language Processing (NLP). Following chapter describes about each and every steps involved in text preparation in detail.

2.3.4 Analysis and Reporting

After preparing the text data, to gain insights or extract useful information, we need to employ different text analysis techniques like Text Classification, Summarization, etc [Anandarajan et al., 2018]. Inorder to achieve a desired result, depending upon our problem, either we can implement a single method or a combination of multiple methods.

After analysis, finally, we need to find an efficient way to share the obtained results. Visualizing the results of analysis is an important part of the storytelling strategy and it helps to convey the obtained findings. Powerful visualizations helps the audience to understand the results more easily and effectively. The visualization methods like Word Cloud or Connected Graphs can be used to present the extracted information from the text data [Barrasso, 2018].

Chapter 3

Requirement Analysis and Evaluation

This project focuses on understanding the problems and the lifestyle of a particular community in Ahmedabad, India. The motivation for this research came from the United Nations Sustainable Development Goals that focuses on achieving a better future for all by 2030. This chapter describes about the different requirements needed for the successful completion of our project and methods for evaluation.

3.1 Experimental Environment Requirements

3.1.1 Dataset

The dataset provided by **Community Tracks : Tech Solutions for a Better Life** has been captured by conducting surveys (interviews, observations, and focus groups) using a grounded theory qualitative research approach, with selected stakeholders from within and those who are associated with the target community.

The survey was rooted in understanding the overall well-being management in a slum community in India. Specifically, the survey tried to (i) understand the status of health within the target community, (ii) define the influence of poverty and employment on their lifestyle, and (iii) gain insights into technology designs induced within the target community. It was assumed of any prior knowledge about these investigations, so the theory (which is grounded in the data) can only be generated once the data has been analyzed as suggested in the inductive approach.

Observations, interviews and focus groups were conducted for seven target communities for capturing seven relevant study categories. The questions asked during the survey belongs to these seven different categories as follows :

- **Self Management** : Dealt with understanding how the people in the community manage themselves individually. This includes questions that enquires about their food pattern and daily activities.
- **Social Support** : Tried to understand how the members of a family, different families or the people in a community support and advises each other.
- **Chronic Diseases** : Investigated about different chronic diseases (mainly focused on the case of diabetes) prevailing in the community and understand the reasons behind that.
- **Communications** : Tried to understand what is the main source they used for communicating with each other or with the outside world and how do they benefit from each. It also included questions about the language that they use in general, also about the social activities and celebrations.
- **Job and Income** : Dealt with understanding about their common jobs, the way they try to get jobs (through employment agencies or NGOs), the services they use (banks, insurance, government schemes) and how they use technology for generating their income.
- **Poverty** : In general this category of questions tried to investigate about their education, condition of their house, fuel they use for cooking and also about the possessions that they have.
- **Tools and Design** : Tried to understand about how different technologies like glucose monitoring devices and strips, M-pesa, RFID card, mobile phones, etc, are used currently by the people in the community. Also investigated about what is the probability to use those by them in the future.

3.1.2 Research Methodology

This section describes about the methods followed to achieve the desired goal of our project. It includes the details about the general approach that has been taken, including the steps followed like data evaluation, data translation and transcription and the experimental setup.

1. **Data Evaluation:**
The data evaluation process has been split into three investigations. Specifically, checked for whether

    - the status of health within the target community is included
    - the influence of poverty and employment on their lifestyle are defined and
    - gained insights into the technology designs induced within the target community.

2. **Translating and Transcribing the Data:**
As the data was gathered from rural communities in Ahmedabad, India, they were in Gujarati and Hindi language (with majority in Gujarati language). For making the process of analysis easier, the video interview data are translated and then transcribed into English language, so that we will have the text data for proceeding with the text analytics processes.

3. **Experimental Setup:**
Python is a programming language which have many packages that are specialised for data science tasks. As Python contains many interesting libraries like nltk, pandas, numpy, etc, all the tasks involved in text data preprocessing and analysis could be done easily and this helps to makes the Natural Language Processing an easier and fun task. The Jupyter Notebook is an open-source web application which supports over 40 programming languages including Python. The whole project has been done by using Python in Jupyter Notebook.

    - **Preparing the Training Data :** To start with the process of text classification, we need to have a training data. This is prepared from the list of categorized questions that has been used for conducting the survey.
    - **Dataset Preparation :** Initially we need to preprocess the unstructured text data to make it ready for analysis by using following methods:
        - Data Conversion
        - Standardizing and cleaning
        - Tokenizing
        - POS tagging
        - Recognizing the named entities
        - Removing stopwords

  - Lemmatizing

  Both the training data and the transcribed interview data has been passed through the same preprocessing steps during the data preparation stage. After preprocessing we need to represent the data in a desirable format so that the developed models can take it as its input and provide necessary results as output.

- **Information Retrieval :** The preprocessed dataset is given as input to a model and it should provide the themes within the data as the output.

- **Interpreting and Visualising the Results :** Interpret the result or the keywords (as suggested by the inductive approach) obtained from the proposed automated analysis. Different relevant keywords or information obtained as output from the model can then be visualised using different approaches like word cloud or graphs.

4. Building the Framework:
The framework consists of three different models :

- A text classification model using a simple neural network with only one hidden layer, which classifies the data into seven classes. The model is initially trained on the prepared training data, which are already been classified into seven classes. After training, the preprocessed interview data is fed into the model for assigning it into its respective classes. Word Cloud is used to visualize the obtained result and understand the themes.

- A Word2Vec model that converts each word into vector format and then identifies the related keywords from the data.

- Combination of the above mentioned models, so that we can extract more useful information than obtained by using those methods individually.

3.2 Evaluation

Evaluation of the results obtained can be done using following approaches:

- Compare the results:

  - Identify the general keywords that ususally represents the lifestyle of the respective communities, by specifically considering the case of health, food, job, social activities, etc, through a secondary research.

  - Compare the interpreted results obtained from the automated analysis with the dimensions derived from the secondary research.

- Compare the results obtained using automated analysis with selected random documents and check whether the obtained results are accurate by manually scanning through those documents.

Chapter 4

Dataset Preparation

4.1 Introduction

It is undebatable that majority of the time of a data scientist is spent in preparing the data, since the collected data can be in various sizes and different formats [Dumais et al., 1998]. To make the data usable and to extract most out of it, it is very important to reshape and cleanse the data. This chapter provides a detailed description about the steps that has been followed to prepare the data inorder to leverage it for analysis.

4.2 Translation

As the available data was in Gujarati and Hindi language, and Community Tracks were not familiar with that language, the translation of the data into English language was inevitable as specified in the requirements section. The available data was comprised of 81 videos whose altogether duration was about 18 hours. We need to take care of a variety of factors while performing the translation. Following provides a detailed description of such factors that came across while translating the video into English language.

- Presence of Multiple Languages : People often assume that if a person is well educated and knows multiple languages, they would have the skill to produce good translations. In fact, only a very few multilinguals will have the right skills to become a good translator. As our data contains interview in both Gujarati and Hindi languages, to translate well, just having a good knowledge of the source language is not enough. For achieving a good translation, we need to have an excellent and advanced knowledge of both the languages, which means that we need a near native ability. Because without that, subtle or even not so subtle nuances of meaning would be missed and we won't ever be able to translate it accurately.

- Glossary of Alien Words : Some of the colloquial words used in the video are used as such without translating it into English. Following table provides the glossary of such words and the category which it represents.

Category	Words
Gujarati Dishes	dal, bati, sabji, roti, rotlo, kichdi, shira, shiro, puri, chana, ghatiya, laapsi, bajra, dosa
Gujarati Snacks	poha, fafada, pakodi, paplu, pappad, mamri, bhungla, sav murmura, dhokla
Addictions	gutkha, miraj, supari, chikni, charas, ganja
Relatives	phua, mama, mami, bhabhie, bhai, baapu, amma, jeit, bhaiya
Persons doing rituals	bhuva, kitlaji, pujari, baba, vibhuti (scared powder), dhaana (ritual with seeds), bajan (ritual songs)
Dons	gundaraj, dhadha
Greetings	namaste, jay ambay
Methods of cooking	sigdi, chula
Residential area	chaul, sarpanch (head of the area)
Stores	raddhie, pitta

Table 4.1 : Glossary of Alien Words

- Multi-Actor Scenario : Even though most of the conversation is in a one-to-one format (between interviewer and respondent), in some of the videos where there are more than one respondents sitting as a group, sometimes they use to speak in between other's conversations. Capturing all those dialogues in a systematic way is really very important and it needed extra care and attention.

- Choosing the Right Word : To be a good translator, being an adequate writer in the target language is not enough. We need to be an excellent writer, a wordsmith, and should have a way with words to write with flair. During translation, choosing the right word according to the context is very important and this involves a complex mental processing. For example, in Gujarati, the word "Ben" can mean "teacher" or "sister". The meaning changes according to the context in which they have been used.

- Sentence Framing : During translation, we need to simultaneously juggle meaning, vocabulary choices and differing grammar systems, inorder to frame the sentence correctly in English language. And this helped to convey the required meaning and to produce well-worded text, with phrase after phrase, without any failure. Since the sentence format followed in Hindi, Gujarati and English is entirely different, we need to take extra care while translating and framing the sentences in English language, else the meaning would end up as entirely different.

- Audio Clarity : Even though majority of the conversations were very clear, there were still many dialogues which were not clear enough to capture and translate, as those videos were captured from outdoor areas like street-sides in Gujarat where there were so much of background noises of vehicles and people passing by. Unfortunately, because of this reason, minimal amount of such dialogues were missed out.

- No Interpretation : As our data is an interview data, sometimes during the conversation most of the actors were speaking or asking about the same thing repeatedly. Each and every such dialogues are translated without skipping or interpreting anything. Keeping the dialogues as such is very important, as our project explores on how to extract useful information from such a raw unstructured data.

- Time Spent : As we need the accurate data, the translation were performed manually without using any tools, like a voice translator. Therefore for translating a 20 minute of video data itself, it were taking about 4 hours, as we need to pick each and every word spoken by everyone in the video and need to accurately frame it correctly in the target language.

4.3 Transcription

As our project explores how to extract useful information from the unstructured textual data, doing only the translation was not enough to proceed forward with the process of analysis. The translated data was needed to be transcribed into textual format. As translation, several factors need to be kept in mind while performing the data transcription as well. Such factors are:

- Structure : As the transcribed data will be used as an input for further analysis, it need to follow a standard structure. During transcription, we need to keep in mind that in future, to leverage it for analysis, this data is to be converted into a machine readable format. So, maintaining a standard structure throughout the transcription process will help to ease the data preprocessing step.

- Actor Specification : In our data, we need to specify which dialogues are spoken by whom and this need to follow the above mentioned standard structure throughout, so that during analysis, identification of the actors who had spoken about a particular theme can be done very easily.
- Connected Words : The group of words that convey a single idea need to be connected using hyphens. For example, the word ”well wisher” need to be written as ”well-wisher”, so that we can keep those words together even after undergoing the data preprocessing step. Otherwise, the preprocessing step would separate that word into two, and thus the collective meaning would be lost.
- Punctuation : During transcription, we need to take care of punctuation marks, with giving more care for fullstops, so that the machine can identify where is the start and end of each and every sentences.
- Proper Nouns : In our data, there comes a lot of proper nouns like names of a person (Eg. Meenakshi Ben), place (Eg. Ranip Crossroad) or organization (Eg. Manav Sadhna). These need to be represented in its proper format with the initial letter as capital. So that the algorithm can correctly identify and process it accordingly.
- Background Information : As the transcription does not provide any visual information, inorder to understand which dialogues are said by whom and to whom, we need to include those background information within the transcribed files. This need to follow a standard structure, as during preprocessing the machine should be able to identify those from the other data.

4.4 Data Conversion

4.4.1 Training Data

For preparing the training data the categorised questions are labelled with their respective categories and was stored in a tabular format which is then saved into a text file. Following table shows the category distribution across the training dataset. The table shows that there is class imbalance and also there is only less training data. So, during model selection we need to deal with these issues.

Labels	Number of Questions
Self Management	10
Social Support	11
Chronic Diseases	16
Communications	5
Job and Income	9
Poverty	10
Tools and Design	5

Table 4.2 : No. of questions with respective labels.

4.4.2 Transcribed Data

As the transcribed files of 81 videos were in 81 different word files, we need to aggregate and convert all those data into a tabular format for proceeding with the text preprocessing and analysis. Doing this aggregation manually will consume a huge amount of time, so this is done by using the Python programming language with the help of libraries like pandas, docx2txt, re and os. Extra care need to be taken while converting these data files into a single tabular format, since each questions and responses should be aligned with each other as in the data we already have. Since we had followed a standard structure while performing the transcription itself, it was not that much tough to perform the required conversion. The columns or the features that we need to have in our tabular data are:

- ”AskedBy” : Represents the person who has asked the question. There are 3 interviewers (Interviewer, Interviewer1 and Interviewer2) and we need to specify each of them in their respective scenarios.
- ”Question” : Represents the question asked by the specified interviewer.
- ”AnsweredBy” : As the interview is conducted from different locations in Ahmedabad, also in cases like focus groups there are multiple respondents, there are many values which comes under this category. In most of the cases, wherever possible, we had managed to provide their actual names as well.
- ”Answer” : Represents the answer or response provided by the respondents.

AskedBy	Question	AnsweredBy	Answer
		Lady1	We have 26 to 27 anganwadis from one corner to another corner.
Interviewer	That means there is one anganwadi for every 200 houses?	Lady1	Yes.
Interviewer	That means there are 26000 people here.	Lady1	Yes. That much is for sure, from one corner to another corner there are that much.
Interviewer	That means this is a huge community.	Lady1	There is a lot in the interiors. How much ever population is there in the interior, that much is there in the exterior also.
Interviewer	When we come from Gandhi Ashram, we have seen some near the bridge and here.	Lady1	There is still a lot in the interiors. Till Ranip Crossing there is lot of them.

Table 4.3 : First five rows of aggregated tabular data.

The above table shows the first five rows of the aggregated tabular data. For the first row there is no question asked, as the first video starts with the dialogue said by the lady.

4.5 Data Preprocessing

Natural Language refers to the language that is being written and spoken by humans and Natural Language Processing attempts to extract information from these human language using various algorithms. Before getting deeper into the algorithmic details, let us now try to understand what it means for a computer to store the human language like the sentence you are reading now. While these words echo in our mind with the full meaning and energy, for computers these are simply patterns of pixels or bits which are known as strings in any programming language [Collobert et al., 2011]. Most of the computer processing that is being applied to these human language is just shuffling and skating over these strings or symbols, without actually understanding what they means or signifies. For example, if we feed the computer a sentence like "A little house by the lakeside, with a shining sun spreading light." does not evoke any nostalgia as us, since the system is not able to identify its underlying meaning.

The decisions taken during preprocessing of the data greatly influences the text document representation which is being given as an input for analysis. Efficient and detailed text data preprocessing helps to reduce the dimension of the data and makes the process of analysis smoother. The preparation of the text is broadly divided into two: (i) Text Preprocessing and (ii) Term Document Representation. Following provides a brief discussion about the steps followed for processing our text data.

4.5.1 Standardization and Cleaning

Standardization and cleaning of the input data is extremely important, as it is going to have a huge impact on the quality of the results we obtain. Following steps are performed while standardising and cleaning our text data:

- Removing Background Information : As our data contains the background information (like to whom they are speaking), since it may affect our analysis adversely, we need to remove it. During transcription, this background information has been included within brackets. So while cleaning, it can be removed easily by using only a single statement.
- Dealing with Hyphenated Words : The hyphenated words like "well-wisher" should be always kept together, even during analysis. So before dealing with other punctuations, we need to deal with this hyphen at first. Those hyphens should be replaced with an empty string. So that the word "well-wisher" would be converted as "wellwisher".
- Removing Punctuation and Numbers : Since in text analytics, we are giving more importance to the content of the text than the numbers and punctuation, it is necessary to clean the textual data by removing those punctuation marks and numbers, and this helps to make the data content simpler and easier to read and analyse.

4.5.2 Unitize and Tokenize

The next step of text preprocessing is the identification of the units in text upon which the analysis need to be done. This is also called as unitization. Tokenization is the process of breaking up a piece of text into many pieces such as words, group of words or sentences according to the units that is been identified during unitization [Anandarajan et al., 2018]. This is done by separating the text units on the basis of occurrence of spaces or punctuations. As the following example, in our project the tokenization is done on the basis of uni-gram concept where each term or unit have only a single word.

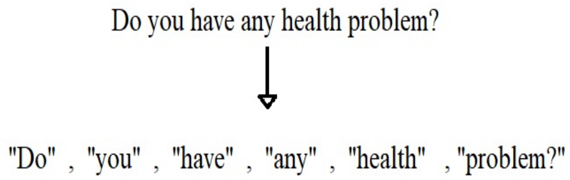

Do you have any health problem?
"Do" , "you" , "have" , "any" , "health" , "problem?"

FIGURE 4.1: Example of Word Tokenization.

Here the sentence ”Do you have any health problem?” has been split into six tokens ”Do”, ”you”, ”have”, ”any”, ”health” and ”problem?”.

4.5.3 Parts-Of-Speech (POS) Tagging

Parts-Of-Speech (POS) Tagging simply means tagging words with their respective Parts-Of-Speech, based on its definition and the context. In the world of NLP most of the models are based on the Bag-Of-Words concept which fails to identify syntactic relations among words [Neale et al., 2018]. For example, the Bag-Of-Words model identifies the word ”like” in ”I like you” where ”like” is a verb representing a positive sentiment and ”I am like you” where ”like” is a preposition having a neutral sentiment, as same. POS tagging helps to build parse trees which are helpful in doing Named Entity Recognition and identifying relation between words [Kumavat and Jain, 2015]. In our case, POS is assigned based on certain rules (for example, the words ending with ”ing” or ”ed” can be assigned as a verb), and it is called as Rule Based Tagging. Inorder to achieve high precision with this approach we will have to use an exhaustive set of hand coded rules. As these rules are already predefined in Python’s Natural Language Tool Kit (NLTK), POS tagging were done with the help of only a single statement.

4.5.4 Named Entity Recognition

The main idea behind Named Entity Recognition (NER) is to search particularly for the named entities like person name, locations, dates or times in text and classify them to an appropriate predefined category for the purpose of information extraction about those entities [Ratinov and Roth, 2009]. Named entities are meant to refer to discrete things in the world which can be referred using a defined name. Some examples of named entities in our data are "Manav Sadhna", "Diwali", "Ranip Crossing" etc. If we are using Bag-Of-Words approach without POS tagging, the word "Manav Sadhna" is taken as two seperate words "Manav" and "Sadhna". For this reason itself POS tagging and NER are really very important in Natural Language Processing. As our project focuses on defining the lifestyle of a particular community, most of the named entities will not be having much importance in this context. Therefore, as part of the text preprocessing stage, the recognised named entities, which has occured less than three times, are removed from our data.

4.5.5 Lowercasing and Stop Word Removal

For machines, the words "Diabetes" and "diabetes" are considered as different, since as mentioned before, the system is based on the concept of Bag-of-Words model, where the underlying meaning is not taken into account [Neale et al., 2018]. But for humans, both means the same. So we need to standardise the words into a single format (like lowercase), so that the system would understand those words as same.

Stopwords are the commonly used words (such as “the”, “a”, “an”, “of”, “in”, "would" etc.) that does not contribute much to the context of the sentences. We do not want these words as part of our data and utilizing our valuable processing time. So as a part of text preprocessing, we need to remove them inorder to limit the number of tokens and thus to improve the efficiency of our analysis.

4.5.6 Lemmatization

Lemmatization is the process of grouping together the inflected forms of a word identified by the word's lemma [Anandarajan et al., 2018]. This process remove inflectional endings of words by returning the dictionary form of the word using vocabulary analysis. Lemmatizing leverages more informed analysis by grouping the words with similar meaning, based on the context around the word, part of speech and other factors. For example, in our data the word "addictions" and "addiction", are lemmatized into its root form "addiction". Because of considering the context of words, lemmatizing is typically more accurate than other methods like stemming, but the downside is that it may be computationally more expensive than stemming. Lemmatizing helps us to take the data preprocessing one step further by providing better data for analysis.

Chapter 5

Word Vectorization

5.1 Introduction

The qualitative data that has been collected through a survey (interviews, focus groups and group discussion) will help to discover far more than what we intended to find. An interview data is comprised of the dialogues between the interviewer and the respondent, in which the respondent might respond with whatever has came into their mind, even it can be out of the scope. So to develop a framework that drill deeply down into such a raw data and retrieve useful information is a big challenge. This requires systematic transformation of the unstructured data into a structured format using a method that is able to capture the underlying themes within the data.

When we deal with a text data, we cannot use the text as such as an input to a model, as the computational systems will not be able to understand their meaning or what it signifies. Instead, we need to convert and represent it in a numerical format, so that the algorithm within the model can do the necessary mathematical transformation and obtain the desired result. Nowadays, many researchers who deals with the textual data are using neural word embeddings, which helps to represent each and every word in the vocabulary as a set of numbers [Fernández-Reyes et al., 2018, Kusner et al., 2015]. In this project we are using the word embedding known as Word2Vec, inorder to extract the keywords that helps to understand about the well being of people in the community.

5.2 Artificial Neural Network

As Word2Vec is developed using a neural network, this section provides a simple definition about what is a neural network and how it works. Artificial Neural Network is developed by imitating the functioning of neurons in brains. Nowadays most of the efficient complex machine learning models have been developed using this concept [LeCun et al., 2015].

An Artificial Neural Network consists of multiple neurons or nodes arranged in multiple layers:

- One input layer : The number of neurons in the input layer is determined by the number of features we are having in our data.
- One output layer: There can be one or more neurons in the output layer and it depends on the problem we want to solve.
- One or more hidden layers: The neurons in the hidden layers will be interconnected with the neurons in the input layer, output layer, as well as within the hidden layers. The output of neurons in a layer is fed in as the input of neurons in the adjacent layer. The flow of the input and output of hidden layers continues back and forth through the model by the processes called as feed forwarding and backpropagation [LeCun et al., 2015], as long as the correct target word provides higher probability.

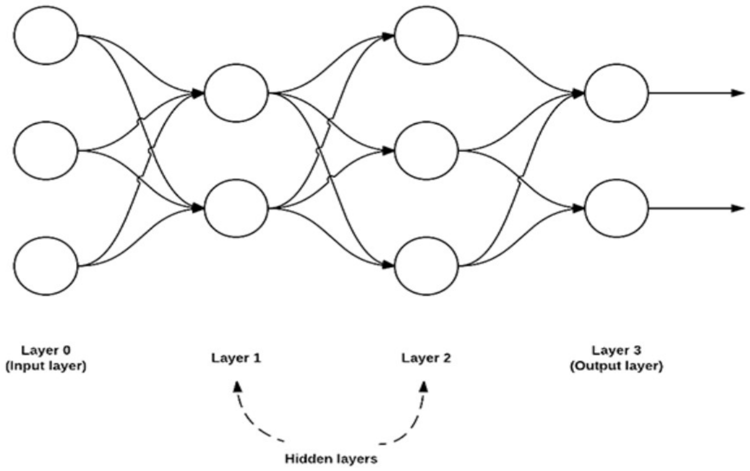


Figure 5.1: Example of a Neural Network [Rosebrock, 2016].

Deep Learning is a phrase that has been used to represent a deeper or complex neural networks [Goodfellow et al., 2016]. The concept as well as working of deep learning is similar to that of a neural network. The complexity in deep learning is determined by the elaborated patterns of flow of information throughout the network.

5.2.1 Feedforwarding

In a feed forward neural network, the flow of input to the output layer will be only in one direction as shown in the above figure and there would be no cycles in the flow of values. The steps followed in feed forwarding is as follows:

- At first the nodes in the hidden layers are initialised with some random weights or values.
- The dot product of the input and hidden layer weights are transformed by using the respective activation function and is passed on to the next layer. The same procedure from one layer to next layer is continued till it reaches the output layer.

5.2.2 Backpropagation

Backpropagation is the process by which the neural network refines the weight values at the hidden layer depending the expected value at the output layer. Backpropagation passes through the following steps:

- When the feedfrowarding process reaches the output layer, it need to check whether the result obtained after applying the activation function is confirming with the expected output value. If not, then the error or loss function is need to be calculated.
- The error is backpropagated and the hidden layer weights are adjusted back to back using a gradient descent algorithm.
- Again the feedforwarding with the adjusted weights are done as similar as before and this process continues till the network completes certain number of epochs or reach a certain level of accuracy.

5.3 Word2Vec

Word2Vec is a three layered neural network - one input layer, one hidden layer and one output layer - that helps to process the input text corpus and provides the vector representation of each word in the provided corpus as its output. Therefore, it is also termed as neural word embeddings. Even though Word2Vec is not trained and built using a deep learning neural network, the produced vector format is understandable by any deep learning model.

The process of converting the text into numbers is called as encoding and the most famous encoding is one-hot-encoding [Ganegedara, 2018]. In this each word in the corpus is encoded by using 0s and 1s and the dimension of a word will be equal to the number of unique words in the corpus. Consider the following text from our data.

"They know Hindi and Gujarati. But in Gujarati and Hindi there are different forms."

So in total, now there is 11 unique words in the example text. From that, after preprocessing the stopwords and punctuations would be removed and all words would be lower cased. So the resulting text will be as follows.

"know hindi gujarati gujarati hindi different form"

Now there is 5 unique words and each word is one-hot-encoded as follows.

Unique Word	Encoding
know	[1 0 0 0 0]
hindi	[0 1 0 0 0]
gujarati	[0 0 1 0 0]
different	[0 0 0 1 0]
form	[0 0 0 0 1]

Table 5.1: Example of One-Hot-Encoding

But the problem with one-hot-encoding is that it does not capture the relation between different words, as it just encodes each word by only considering the number of unique words in the corpus and thus the distance between each words are same to each other.

Each words in the provided text corpora represents a discrete status, and inorder to capture the syntactic and semantic relation between different words, we need to find the probability of co-occurence of those statuses. This is where the Word2Vec algorithm become really useful, as it tries to understand the probability of co-occurence of each and every word and this helps to group the words that were having similar meaning or co-occured together [Mikolov et al., 2013]. Based on the appearances of each word in the provided data, Word2Vec is able to find the words that are related with each other word and thus the algorithm generates its training data by itself.

5.3.1 Training Data Generation

Even though Word2Vec is an unsupervised model, it still need to leverage into a supervised mode and thus need a training data to learn. The training data is produced from within the data without having any auxiliary information like labels. The training data needed to train our neural network is built from the provided text corpora, by considering the neighbors of each and every word present in it [Mikolov et al., 2013]. Window size determines how many words should be considered as a word's neighbors. When window size equals 1, in our example a word's neighbor would be its immediate adjacent words to its left and right, and thus our training data would be as follows.

Word	Neighbor
know	hindi
hindi	know
hindi	gujarati
gujarati	hindi
different	hindi
different	form
form	different

Table 5.2: Generated training data with window size=1.

Now our training data is ready and the three layered neural network will be trained with the one-hot-encoding of each word as its input and the one-hot-encoding of the neighbor as its target. Thus, each word is trained against its neighbor words. The number of nodes in the hidden layer determines the dimension of the vector produced for each word. For example if we want to represent every word in the vocabulary as a 5 dimensional vector, then we should have 5 nodes in the hidden layer. The weight vector at the hidden layer of the optimized neural network, of each word as input, has been taken as the vector for that particular word.

While training the model, it captures the semantic feature of every word by considering its context of appearance in the data, thus each word in the vocabulary has been described by another word in the provided data. Word2Vec training is done in one of the two ways [Ganegedara, 2018]:

- Continuous Bag of Words (CBOW) : In CBOW, the model learns to predict the target word when the surrounding context is provided. For example in our data if the word *"sugar"* and *"pressure"* are given it would predict the target word *"diabetes"*.
- Skip-gram : On the other hand, in skip-gram, the target context is predicted from a word. During training if the word is not able to predict its surrounding context

properly, the weights at the hidden layer (that means the feature vector of a word) are continuously refined. Thus, the surrounding context acts as the teacher to help a word to adjust its vector values.

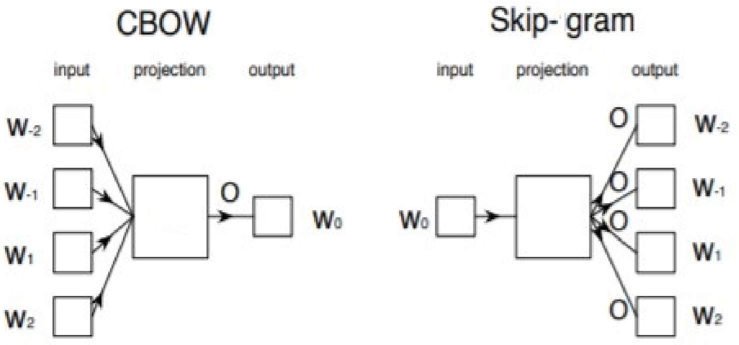


FIGURE 5.2: CBOW and Skip-gram model [Ling et al., 2015]

During each epoch of training the distance between more related words becomes closer and closer. Also the words which do not have any relation, move farther and farther. A well trained Word2Vec model places the words that conveys same meaning together, based on the context in which it has appeared in the data. The relation between words are converted into numerical distance, thus the quality is converted to quantity which can be processed by any computational model. But the words are only learned and related based on the vocabulary that has been presented to the model, and not beyond that.

5.3.2 Hyperparameters

Hyperparameters are the attributes that are need to be decided by the designer inorder to run the Word2Vec algorithm. These attributes decides how the Word2Vec should perform to achieve the desired result. The important Word2Vec hyperparameters are as follows [Mikolov et al., 2013]:

- Dimensions : The dimension of the produced word vector. The number of nodes in the hidden layer will be equal to this dimension.
- Window size: This defines how many words should be considered as a word's neighbor to capture the context. The smaller window size help to understand the syntactic structure. The larger window size helps to understand the semantic nature of the text and this takes more time to train the data.

- Minimum word count: The threshold that determines the minimum occurrence of a word to consider it to convert into a vector. Only the words occurring greater than or equal to the threshold are considered by Word2Vec.
- Downsampling: This determines how the more frequent words like *"of"* are trained.
- Epochs: The number of epochs the network should pass through to obtain the final vector.

5.3.3 Vector Representation

The vector format of each word is unique and it encodes the word's meaning in such a way that we can directly figure out the similarity or the relation between different words mathematically [Mikolov et al., 2013]. All these are done without any additional human interference. The output of the algorithm would be a dictionary with all unique words with its vector representation, which can be considered as the meaning of that word understandable by any computational model.

In one-hot-encoding, as each word can be represented by using only 0s and 1s, the dimension of each word should be equal to the number of unique words in the provided text corpora and also it does not help to capture the semantic similarity between words. On the other hand, in Word2Vec, a word can be represented using a vector with any numbers, also we can decide the dimension of the vector as needed by initialising the number of nodes in the hidden layers. The more the dimension, the result would be more generalised [Ganegedara, 2018].

For example the 5 dimensional vector representation of the words in our example data, would be as follows:

Unique Word	Embedding
know	[0.070682 , 0.03677826, 0.0182321 , 0.05239728, 0.03675885]
hindi	[-0.08355758, 0.08552153, 0.09521113, 0.01416654, 0.00212295]
gujarati	[0.04028261, -0.00268736, 0.06752592, 0.00837277, 0.04771009]
different	[0.04612756, -0.0529971 , -0.03025002, 0.09165786, 0.08244297]
form	[-0.01124162, 0.0169074 , 0.07344072, 0.09804013, 0.01206439]

Table 5.3: 5-D Vector Representation.

This numerical representation can be fed into any computational model according to our requirement, instead of providing the text in its raw form.

5.3.4 Cosine Similarity

In the case of linear algebra, the similarity between two vectors can be calculated by using the cosine similarity measurement between them. If their cosine similarity is near to 1, it means that the angle between them is near to 0 degree and they are more similar to each other. If a pair of vectors are not at all similar with each other, then their vector representation would be at an angle of 90 degree [Kenter and De Rijke, 2015]. If A and B are two vectors, then the formula for finding the cosine similarity between those vectors is as follows [Wei and Wei, 2018]:

$$cos(\theta) = \frac{A.B}{\mid A \mid\mid B \mid}$$

In our example, if we want to find the most related word with the word 'hindi', then we need to find the cosine similarity between the vector of word 'hindi' with every other words in the vocabulary. The list of vector representation of words in the vocabulary act as a lookup table to find the most related word. The multidimensional vectors of the words which are more similar to each other would be present nearer to each other in its vector space.

5.4 Related Works

- **Bengio et al. [2003]:**

  Initially most of the NLP tasks were performed on the bag-of-words concept, which does not help to preserve the semantics involved and only captured the number of occurrences of a word in the text [Lodhi et al., 2002]. In 2003, Bengio et al. [2003] has proposed the first neural probabilistic model with the objective that inorder to build a proper language model, it needs to understand the sequence of the words in the provided text. Even though the existing model based on ngrams [Goodman, 2001] was able to achieve more generalisation with the test data compared to other pre-existing models, it has resulted in high computational complexity because of high dimension of the text. This model was called as Feed Forward Neural Network Language Model (FFNNLM).

  Discussion : The developed model was capable to learn the distributed representation of words with its corresponding probability in a sequence. The model has performed well in longer context and obtained good perplexity measure. As neural network was used for probability calculation, it was able to capture the shared features and non-linearity in the data. Even though the model deals with high dimension, with larger data, the complexity was increasing linearly. So it needed more improvement in terms of architecture and computational complexity.

- **Mikolov et al. [2010]:**

  In FFNNLM, the model only captured the sequential words following a particular word inorder to calculate the probability. The model developed by Mikolov et al. [2010] was called as Recurrent Neural Network Language Model (RNNLM) and was able to capture the surrounding context of a word. Mikolov et al. [2010] criticised that the FFNNLM model need to specify the context of before hand and since for a word the context window would be different in different scenarios, FFNNLM will not be able to provide accurate results.

  Discussion : RNNLM stores temporal information and this helped to improve the perplexity of the model. Even though the model captures the context dynamically, it was not performing well with longer text.

- **Mikolov et al. [2013]:**

  In 2013, Mikolov et al. [2013] has built a model that were able to produce more generalised results than the previous models that are based on neural network [Bengio et al., 2003, Mikolov et al., 2010] and criticised that those models considers each word as separate units without taking the similarity between words into consideration. The paper has proposed two variations called as Continuous Bag of Words (CBOW) and Skip-gram model. It is been derived from their earlier work which was based on a simple neural network [Mikolov, 2007, Mikolov et al., 2009].The developed model is called as Word2Vec and it considers the fact that a particular word has multiple degrees of similarity and the representation of a word depends on this.

  Discussion : The Word2Vec model was able to capture the similarity between words depending on the context similarity of different words. For example, the words like *"big"* and *"bigger"* were closer in the vector space. Also it shown similarity in words like *"France"* and *"Italy"*, as both represents country names. Word2Vec was working fine even with larger text corpora. But the default model was not capable enough to capture the syntactic details. For example if there is a named entity as *"New Delhi"*, the model considers those as two separate words, *"New"* and *"Delhi"*. And this might cause the model to improper results.

- **Ling et al. [2015]:**

  In this paper, Ling et al. [2015] has explained some additional methods to be done, before implementing Word2Vec, so that it can capture the syntactic details of the provided text, unlike default Word2Vec. The paper criticised that the default Word2Vec alone is not enough to provide accurate results where covering the syntax details is important. Semantics refers to *"what words occurs together"*, but the syntax refers to *"to where each word goes"* in a sentence.

Discussion: The paper has described about the preprocessing techniques like POS tagging and dependancy parsing, to capture the semantics while using Word2Vec, while maintaining the simplicity and efficiency of the model developed by Mikolov et al. [2013]. And this helped to obtain more generalised and accurate results than when using the default Word2Vec alone in classifying the tweets.

- **Chiu et al. [2016]:**

  In 2016, Chiu et al. [2016] has used Word2Vec, as it was the method which provided for efficient result so far. The paper tried to capture the relation between biomedical terms in a text corpora of PMC and PubMed data and has started with the assumption that larger text corpora will give more generalised result. Chiu et al. [2016] experimented with text corpora of various sizes and hyperparameter settings.

  Discussion: Chiu et al. [2016] found that the hyperparameter settings helps to improve the Word2Vec model, but were not able to tell the exact settings to obtain accurate results, as the results with different settings varied with different datasets. The paper concluded that the larger text corpora does not help to improve the model. More than size of the data, the content (even though the text corpora is smaller) helps to find the relation between important keywords with more accuracy.

- **Ghosh et al. [2016]:**

  Ghosh et al. [2016] has used the same unstructured text corpora of PMC and PubMed data used by Chiu et al. [2016] to model diseases, agents and symptoms from it. The paper started with the assumption that when Word2Vec is trained with the whole dataset, it would be able to model the diseases and symptoms accurately. The paper has found that even though the vocabulary driven Word2Vec is capable to find the relation between different words in the corpora, for domain specific problems, like healthcare, the general vocabulary is not enough to model the solution.

  Discussion: The paper at first tried to model diseases with the raw vocabulary of data from PMC and PubMed and found that even though it finds the relation between words, the result is not more domain specific. So it tried to build a vocabulary by including all the words related with disease, symptoms and agents, so that the algorithm can understand the important keywords to model. This vocabulary can then be used in future in conjunction with the current news data to understand the diseases and symptoms in a current outbreak. Instead of Word2Vec, the developed model was called as Dis2Vec.

- **Zhu et al. [2017]:**

  As Chiu et al. [2016], this work also tried to build a model that helps to find relation between biomedical terms by using publication data of different sizes. Zhu et al. [2017] found that the result varies as the content and size of the data varies.

  Discussion: The model has given poor performance when used only recent available data as the content is not saturated for producing a complete model. With historical data, even though the model was able to find relation between words, some mostly highly correlated terms were missing. On the other hand, when the model was built using only the abstracts of the documents, the model provided keywords that are highly correlated with a particular word. But still more keywords were missed out compared to when used large historical data.

- **Khatua et al. [2019]:**

  In 2019, Khatua et al. [2019] has used Word2Vec to do theme wise classification of the unstructured tweets related with Ebola and Zika during outbreaks. The paper has started by saying that the Glove embedding are very rarely used in literatures Zhu et al. [2017]. Motivated by what Chiu et al. [2016] and Ghosh et al. [2016] said about domain specific Word2Vec, the author has tried to classify the tweets with generic pretrained Word2Vec and domain specific Word2Vec.

  Discussion: The paper has tried to answer what could be used during initial outbreak when there is scarcity of data and whether domain specific Word2Vec outperforms pretrained Word2Vec and Glove. During scarcity of data the scholarly abstracts in PubMed related with Ebola and Zika has been found useful for doing the text classification and has also found that the domain specific Word2Vec has provided accurate result than Glove and pretrained Word2Vec.

5.5 Implementation

This section provides a detailed description about the implementation details and present the results obtained by using Word2Vec in our transcribed and preprocessed interview data. Following are the steps followed to find the related keywords from the provided data using Word2Vec representation.

5.5.1 Experimental Setup

All the tasks like dataset preparation and preprocessing, Word2Vec implementation and visualizing the results are done with the help of Python programming language. Word2Vec is implemented using Gensim's Word2Vec model (Word2Vec), which can be imported into the Python environment.

5.5.2 Preparing the dataset

After preparing and preprocessing the raw transcribed interview data, by passing through each of the steps mentioned in the Dataset Preparation chapter, now we have the data with only relevant features. The data does not have any stopwords, numbers and punctuations, the named entities are recognised and attached together, the background information are removed and the words are lemmatized. Following steps are done to further prepare the data to pass it through Word2Vec:

- Each of the preprocessed question and response are combined. We need both the question and the response, as both contains relevant information.
- The combined text has been converted into a list of words. The resulting data format to be passed into the model is a list of list of texts as follows:

  [['anganwadis', 'one', 'corner', 'another', 'corner'], ['mean', 'one', 'anganwadi', 'every', 'house'], ['mean', 'people', 'sure', 'one', 'corner', 'another', 'corner'], ['mean', 'huge', 'community', 'lot', 'interior', 'ever', 'population', 'interior', 'exterior'], ...]

5.5.3 Building Word2Vec

A Word2Vec model which could take the one-hot-encoding of a unique word in the vocabulary as input and give its vector format as its output is designed with the following hyperparameters:

- Dimensions : 300
- Downsampling: 1e-3
- Minimum word count : 3
- Window size : 7
- Epochs : 20

At first, the model try to figure out the unique words in the vocabulary and find one-hot-encoding of each, which is then passed as input to find the respective embedded vector. As said before, each one-hot-encoded representation is trained against the one-hot-encoded representation of its 7 neighboring words, as the window size is 7. During each epoch, the most related words comes nearer to each other in the 300 dimensional vector space. Following python statement helps to do the model training where response2vec is an instance of Word2Vec class.

```
response2vec.train(dataToVectorize,total_examples=response2vec.corpus_count,epochs=20)
```

After training, each word will have its 300 dimensional vector format, which can be then saved into a file. We can retrieve and use it later as a pretrained model without the need of training it again on the same data. Following statement help to save the trained model in a file *response2vec300D20.w2v* in *trained300D20* folder.

```
response2vec.save(os.path.join(”trained300D20”, ”response2vec300D20.w2v”))
```

Already saved pretrained model can be loaded using the following statement:

```
w2v.Word2Vec.load(os.path.join(”trained300D20”,”response2vec300D20.w2v”))
```

5.5.4 Visualizing Vectors using t-SNE

T-distributed Stochastic Neighbor Embedding (t-SNE) [Maaten and Hinton, 2008], is a well known technique that helps to visualize the high dimensional dataset by converting it into low dimension.

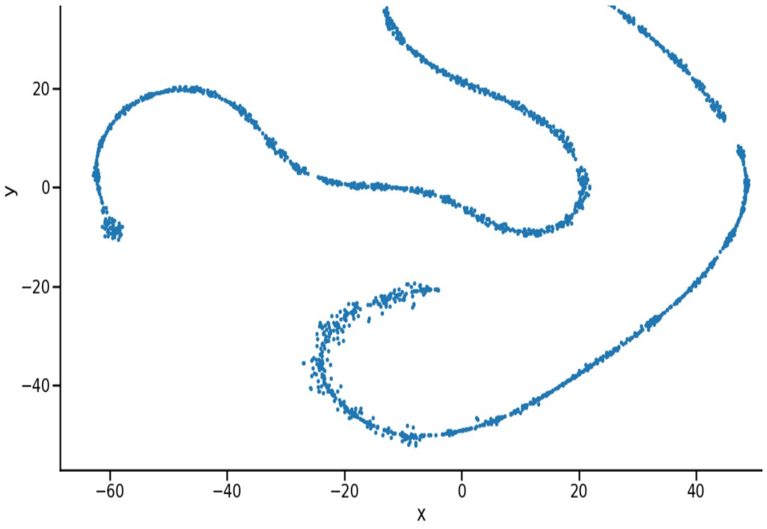


Figure 5.3: Compressed 300-D vector in 2-D vector space after 1 epoch of training.

Our vector representation of each word is of 300 dimension, which is difficult to visualize on a two-dimensional screen. Inorder to visualize it, at first we need to compress our 300D data into 2D format. t-SNE helps to perform this reduction task by preserving the original clustering in the high dimensional space, which is 300D in our case.

Figure(5.3) is the t-SNE plot obtained by training the Word2Vec for only 1 epoch. We can see that all points are clustered near to each other, without any distinction. These points move apart as the number of epochs increases and finally the most related words would be nearer to each other and will have high cosine similarity with each other. Also the words which have no relation would be very far from each other. Following is the t-SNE plot of the vectors obtained after training model for about 20 epochs.

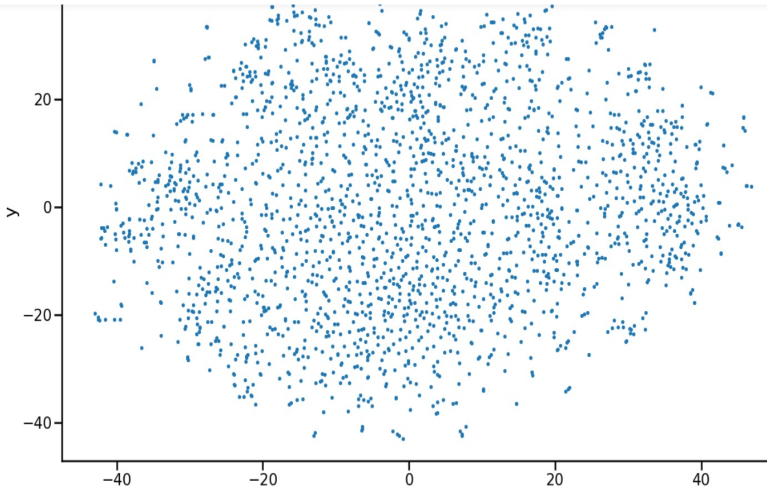


FIGURE 5.4: Compressed 300-D vector in 2-D vector space after 20 epochs of training.

For more clarity, let us now try to zoom into each part of the plot. From the following plot we can see that the words like laziness, suryanamaskar, yoga, paranayam, etc, which are related with the exercises has been clustered together.

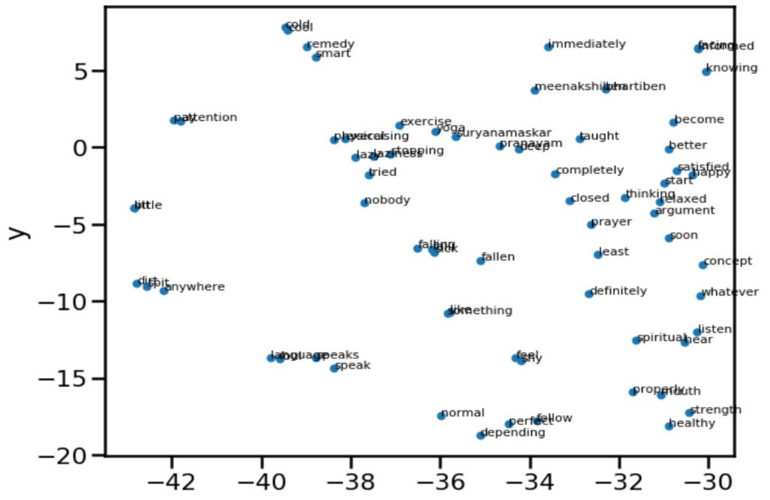


FIGURE 5.5: Zoomed visualization of t-SNE plot

5.5.5 Finding Related Words

The vector form obtained can be used for different purposes like distance calculation, similarity measure between different words or for ranking words in a document. Our project uses Word2Vec for distance calculation using cosine similarity measure to find the most related keywords. Word2Vec class is capable to provide the most related 10 words to a particular word with a single statement. For example the following statement tries to obtain the related words with the word *"language"* in the provided data.

response2vec.most_similar('language')

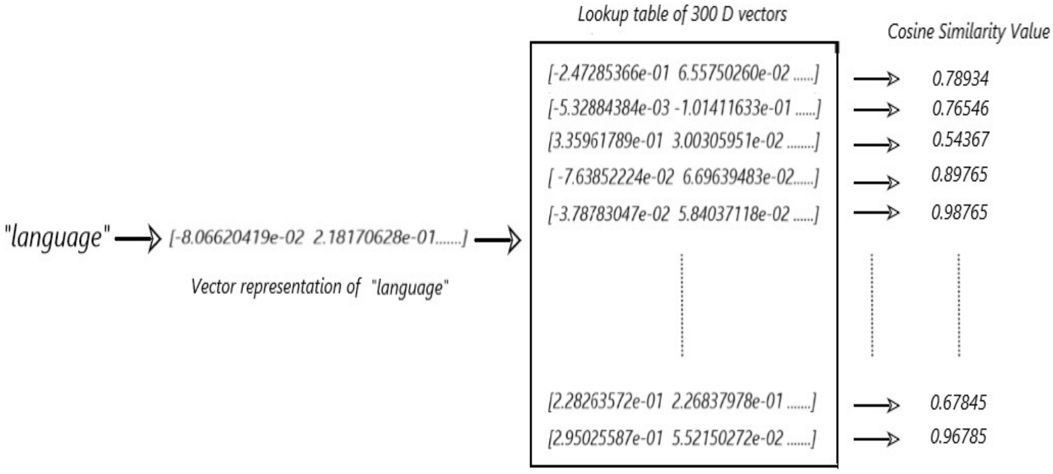

Lookup table of 300 D vectors
Cosine Similarity Value
[-2.47285366e-01 6.55750260e-02 ......]
0.78934
[-5.32884384e-03 -1.01411633e-01 ......]
0.76546
[3.35961789e-01 3.00305951e-02 ....]
0.54367
[-7.63852224e-02 6.69639483e-02......]
0.89765
[-3.78783047e-02 5.84037118e-02 ......]
0.98765
"language"
[-8.06620419e-02 2.18170628e-01.......]
Vector representation of "language"
[2.28263572e-01 2.26837978e-01 ....]
0.67845
[2.95025587e-01 5.52150272e-02 ....]
0.96785

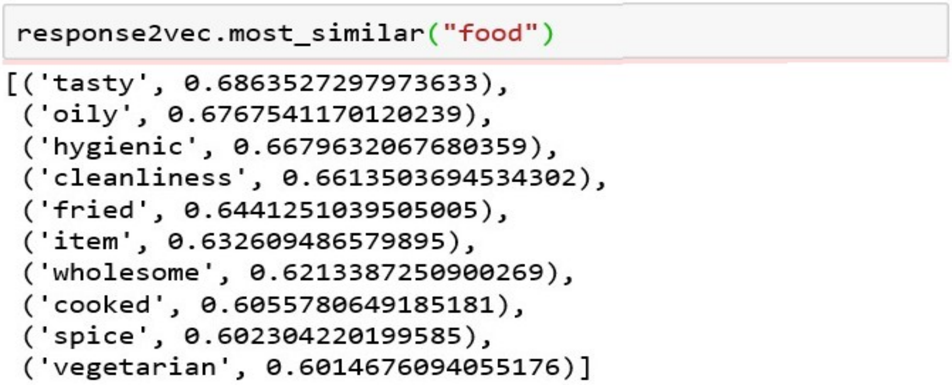
Figure 5.6: Representation of finding most related word with *"language"* .

The trained model will have the 300 dimensional vector representation of all the unique words in the vocabulary. In a skip gram model, when we want to find the most related words or context with a particular word, the vector representation of that particular word is given as the input to the model and the list of vector format of all other words acts as a look up table, from which the cosine similarity between each vector and vector of the respective word can be found. The words having higher similarity will be provided as the output context of that particular word.

5.5.6 Results

Our project need to find the codes and subcodes related with **Well Being Management** of the people in rural communities in Ahmedabad. As a human, we know certain keywords like *language, work, disease, food, fuel, devices at home, etc*, that can define the lifestyle of the people in a community. Let us try to find whether Word2Vec is capable to define about this keywords. Following are the results obtained when Word2Vec tried to drill deep into each of these keywords:

- **Food:** *response2vec.most_similar("food")*



```
response2vec.most_similar("food")
```

```
[('tasty', 0.6863527297973633),
 ('oily', 0.6767541170120239),
 ('hygienic', 0.6679632067680359),
 ('cleanliness', 0.6613503694534302),
 ('fried', 0.6441251039505005),
 ('item', 0.632609486579895),
 ('wholesome', 0.6213387250900269),
 ('cooked', 0.6055780649185181),
 ('spice', 0.602304220199585),
 ('vegetarian', 0.6014676094055176)]
```

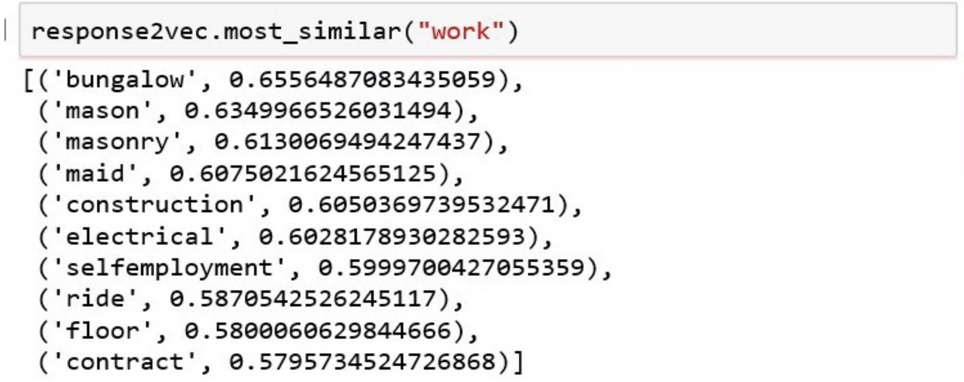
By seeing the below result, we can see that their food is tasty, oily, hygienic, fried, wholesome, spicy and vegetarian. Now let us evaluate the result.

Evaluation: Following are some dialogues about their food that tells whether the result provided above is conveying the right information or not.

*"They eat and drink from outside. Also, they are **unhygienic**. The reason could be lack of literacy or knowledge about health like what type of food should be taken or hygienic food should be taken or in hygienic food what type of food should be taken.", "May be they don't have any idea about this **wholesome** nutritious food. The moment the word nutritious comes, they think that they have to spend.", "They cook everything **spicy** and put lot of **oil**.", "They eat more of **spicy and fried** food."*

Even though Word2Vec has captured most of the information right, like the food is oily, tasty, spicy, fried, etc. In data, it is been said that they are having *"unhygienic"* food. But as the word *"hygienic"* has occurred more frequently with the word *"food"*, Word2Vec has provided that word for us.

- **Work:** *response2vec.most_similar("work")*



```
response2vec.most_similar("work")
```

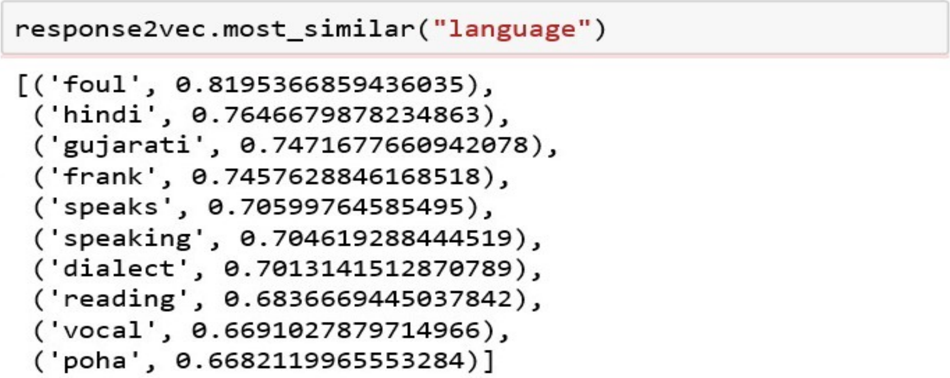
```
[('bungalow', 0.6556487083435059),
 ('mason', 0.6349966526031494),
 ('masonry', 0.6130069494247437),
 ('maid', 0.6075021624565125),
 ('construction', 0.6050369739532471),
 ('electrical', 0.6028178930282593),
 ('selfemployment', 0.5999700427055359),
 ('ride', 0.5870542526245117),
 ('floor', 0.5800060629844666),
 ('contract', 0.5795734524726868)]
```

Evaluation: Following are some dialogues about their work that tells whether the result provided above is conveying the right information or not.

”She goes for work in ***bungalow.****”,”Mostly ladies go as* ***house maids*** *and they also go to pick the papers from the garbage.”,”****Masonry work****, they go for collecting recyclable materials from garbage, pedaling rickshaws.”,”Power house, from where the* ***electricity*** *and light comes. In that also they have work.”*

Even though the information captured by Word2Vec about their work is right, some of the information like *”collecting recyclable materials from garbage”* has been missed out.

- **Language:** *response2vec.most_similar(”language”)*



```
response2vec.most_similar("language")
```

```
[('foul', 0.8195366859436035),
 ('hindi', 0.7646679878234863),
 ('gujarati', 0.7471677660942078),
 ('frank', 0.7457628846168518),
 ('speaks', 0.70599764585495),
 ('speaking', 0.704619288444519),
 ('dialect', 0.7013141512870789),
 ('reading', 0.6836669445037842),
 ('vocal', 0.6691027879714966),
 ('poha', 0.6682119965553284)]
```

Evaluation: Following are some dialogues about their language that tells whether the result provided above is conveying the right information or not.

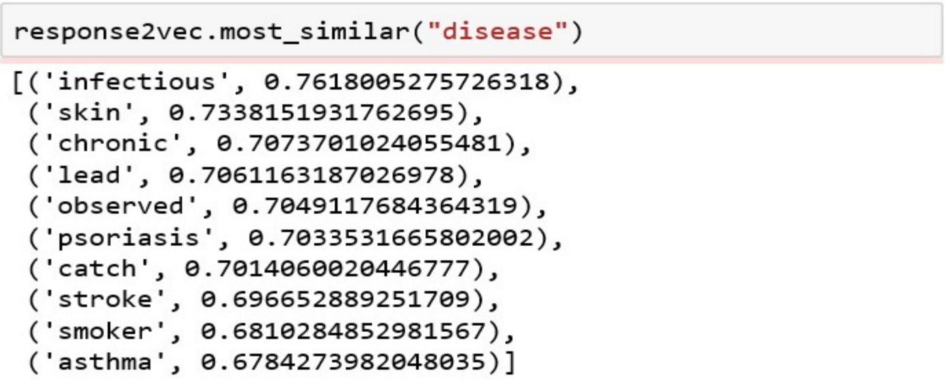
”Lot of ***foul*** *language is spoken in the house.”,”Ladies mostly use* ***Gujarati and Hindi*** *only.’, ’****Hindi and Gujarati****”,”They cannot even speak the pure Gujarati properly. They speak the simple* ***vocal*** *language”,”Even to their parents they speak dirty* ***foul*** *language which we can’t even listen to.”,”Yes, they manage* ***reading*** *and they can speak as well.”,” Because in Gujarat the are many districts and villages, and according to that the language forms of Gujarati also keep changing and* ***dialect*** *will also change.”*

Word2Vec has captured the language that is known and spoken by most of the people there : *foul, Hindi and Gujarati.*

- **Disease:** *response2vec.most_similar(”disease”)*

Evaluation: Following are some dialogues that says about their disease.

”Yes, they catch ***infectious*** *diseases more. We see more of infectious diseases over here”,”****Skin disease*** *patients and* ***asthma*** *patients are more on Tekra”,”Yes. They get* ***psoriasis****. Even they get herpes.”,”Yes. Uncontrolled diabetes, paralysis, hyper tension. There are patients with hemorrhage,* ***severe stroke****, asthmatic patients are there, COPD patients are there”*

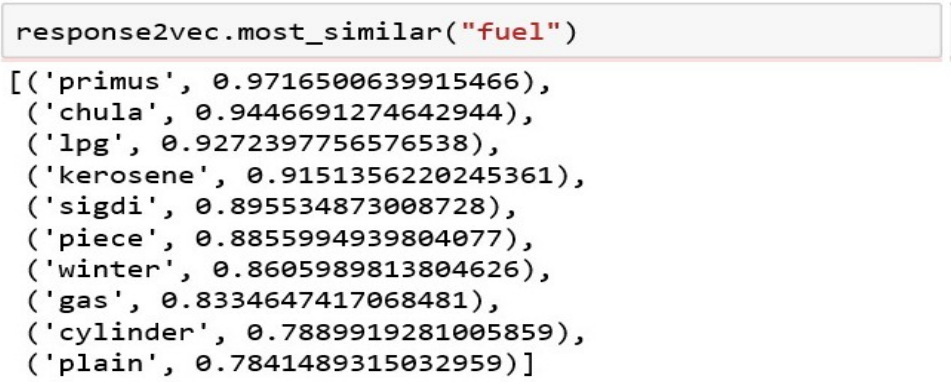


```
response2vec.most_similar("disease")
```

```
[('infectious', 0.7618005275726318),
 ('skin', 0.7338151931762695),
 ('chronic', 0.7073701024055481),
 ('lead', 0.7061163187026978),
 ('observed', 0.7049117684364319),
 ('psoriasis', 0.7033531665802002),
 ('catch', 0.7014060020446777),
 ('stroke', 0.696652889251709),
 ('smoker', 0.6810284852981567),
 ('asthma', 0.6784273982048035)]
```

Even though Word2Vec helped to capture most of the prevailing diseases there, some of the diseases are missed out.

- **Fuel:** *response2vec.most_similar("fuel")*



```
response2vec.most_similar("fuel")
```

```
[('primus', 0.9716500639915466),
 ('chula', 0.9446691274642944),
 ('lpg', 0.9272397756576538),
 ('kerosene', 0.9151356220245361),
 ('sigdi', 0.895534873008728),
 ('piece', 0.8855994939804077),
 ('winter', 0.8605989813804626),
 ('gas', 0.8334647417068481),
 ('cylinder', 0.7889919281005859),
 ('plain', 0.7841489315032959)]
```

Evaluation: Following are some dialogues that says about their fuel used for cooking.

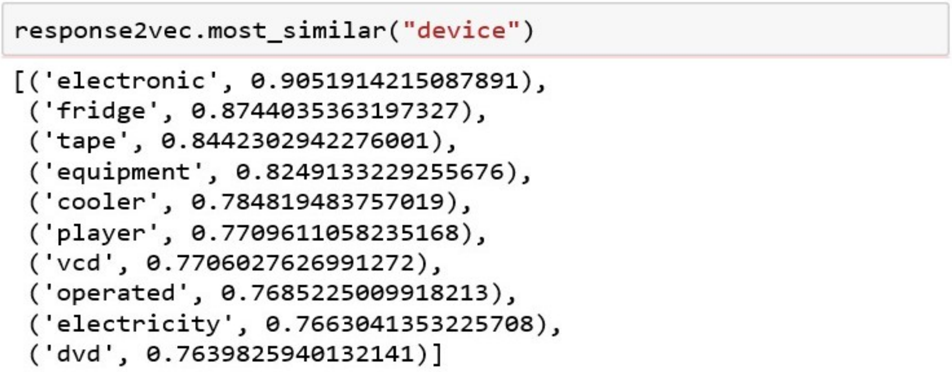
*"They use **gas and Chula**", "Many use **primus**, many use Chula and now a days in many houses gas connection has also come.", "They all have **LPG**, but they use it less so that the cylinder run for 2 to 3 months. They also use **kerosene**, as they get kerosene using their BPL card. Those who don't get kerosene will use Chula also. They bring **wood pieces** and cook on Chula.", "**Gas cylinder**."*

- **Electronic Device:** *response2vec.most_similar("device")*

Evaluation: Following are some dialogues that says about their electronic devices.

*"In electronic devices they have T.V, **fridge, radio and tape**. They use these things and also mobile.","Phone, TV, **Cooler**","Yes. They have **VCD player or DVD player**","**Fridge**, air **coolers**, fan, then iron box is there","In electronic devices they don't have anything except fan","We only have TV.", "TV is there"*

Word2Vec has captured some of the information regarding the electronic devices, but still more need to be captured. From the data it is obvious that most of the



```
response2vec.most_similar("device")
```

```
[('electronic', 0.9051914215087891),
 ('fridge', 0.8744035363197327),
 ('tape', 0.8442302942276001),
 ('equipment', 0.8249133229255676),
 ('cooler', 0.784819483757019),
 ('player', 0.7709611058235168),
 ('vcd', 0.7706027626991272),
 ('operated', 0.7685225009918213),
 ('electricity', 0.7663041353225708),
 ('dvd', 0.7639825940132141)]
```

houses have T.V. But that information is not provided by Word2Vec in top 10 most related values.

5.6 Conclusion

By using **Word2Vec**, we were able to get a lot more by representing a word by means of its context or neighbors. Deeply observe for a moment, that the Word2vec algorithm has never been taught a single rule of English syntax and semantics. Still it learned more in a flexible manner without any human intervention. Word2Vec has started its job on the interview data with a blank slate, as it knows nothing about the world. And as soon as the training completes, it was able to find relations between different keywords.

As concluded by Chiu et al. [2016], Ghosh et al. [2016] and Khatua et al. [2019], in our data, this method can be used if we have the domain specific keywords or vocabulary that helps to define the well being of a community. Also, this keywords need to be present in our preprocessed data. Otherwise, either we would end up in a blank state without having the desired result or the model might show the words that are not important to the context as the most related keyword with high probability.

Chapter 6

Text Classification

6.1 Introduction

Text Classification is a supervised learning problem in which the texts are tagged with some predefined categories on the basis of its textual content [Aggarwal and Zhai, 2012]. Extracting useful information from an unstructured data could be a very challenging task, unless it is being organized in a more efficient way. Organizing the whole data manually can be a cumbersome task, if we are having a large amount of unstructured textual data. This chapter provides an overview about the importance of text classification in Natural Language Processing and describes how it is useful for our project.

6.2 Supervised Learning

Since text classification is a supervised learning problem, we need to first understand what is meant by supervised learning. In supervised learning, we will have a labelled dataset inorder to start with building the model and the result provided by the model depends on these predefined labels, which means that the labelled dataset guides the model to obtain the desired result [Caruana and Niculescu-Mizil, 2006]. In real time, it is critical to have such a labelled training dataset. So, we will need to build it manually. The more the training data, the more generalized the result would be.

6.3 Text Classification Using FastText

The supervised version of FastText is a text classification model developed by Facebook AI Researchers [Joulin et al., 2016]. Nowadays deep learning have been used for building most of the machine learning models, as it is capable of providing more accurate results. On the other hand, these models might show problems when we are having a very large dataset. Also text classification depicts a non linear relation between the features of the text and the labels, which can be captured by a non linear classifier like a neural network. So instead of using deep learning, FastText employed a shallow neural network that imitates a hierarchical classifier. In this project we are using the FastText model, which is based on CBOW [Mikolov et al., 2013], for classifying the interview data into predefined seven categories as mentioned before.

6.3.1 Training Data and Input

The training data that needs to be passed into FastText should be a text file with the text and its corresponding labels. The label should follow a particular format, so that the model can differentiate it easily. By default, the model expects the labels to be prefixed with *"__label__"*. So the labels are presented to the network as a normal word of the text appended with *"__label__"*. The whole input data for which the label is to be predicted should be provided as a list of text, so that each list item can be assigned with a label depending on its cosine similarity with the context.

6.3.2 Feature Engineering

As in the case of text data, the words in the text are considered as the feature of that data and each row of text would be having different number of words. To feed the data into a neural network every row should be having same number of features. So when we are using a neural network, we need to tweak the number of features by one of the following ways:

- **Slicing or Padding:** Consider the following example of two rows of text.
  - *“They know how to read and write Gujarati”*
  - *“They use smartphones with touch-screen. They use Gujarati or Hindi language in their phone”*

  After the initial data preprocessing by removing stopwords and lowercasing these text would become as:
  - *“know read write gujarati”*
  - *“use smartphone touchscreen use gujarati english language phone”*

So inorder to make the number of words as same, either we have to slice the longer text or pad a default character to the shorter text, so that the number of words would become as same for all the text [Ganegedara, 2018]. Suppose for the above example if we are taking a thershold for the number of features, say 5, then the first sentence should be padded with a single default character say *"nan"* and from the second one, the extra two words from the text should be sliced. So we obtain the following:

  - *"know read write gujarati nan"*
  - *"use smartphone touchscreen use gujarati"*

- **Taking Average of Word Vectors:** The data preprocessing and word embeddings will help to convert each word of the text we are having into its corresponding numerical representation [Mikolov et al., 2013] with each word as a vector of same dimension. So for a sentence or a record we can take the average of all words comprising that record. Thus each record will be having same number of features as we need.

- **Average of NGrams**: In FastText a text is considered as combination of NGrams instead of combination of words. For example if NGram=2, then considering a word *"gujarati"*, its corresponding NGram representation would be as

  $$"gu","uj","ja","ar","ra","at","ti"$$

  The vector representation of a single word would be the average of the vector of the constituting ngrams. Similarly the vector representation of a sentence is also taken as the average of the vector of the constituting ngrams [Joulin et al., 2016].

6.3.3 Architecture

The supervised FastText classifier has the following architecture:

- Input Layer: The vector representation of the ngrams constituting the text are given as input. So, the number of input nodes will be equal to the number of ngrams in a data record.

- Hidden Layer: The embedded input features are averaged to form the hidden layer. So that the number of nodes in the hidden layer would be equal to the embedding dimension of an ngram.

- Output layer: The output layer implements a hierarchical softmax architecture, which is a binary tree having leaf node for every word in the vocabulory.

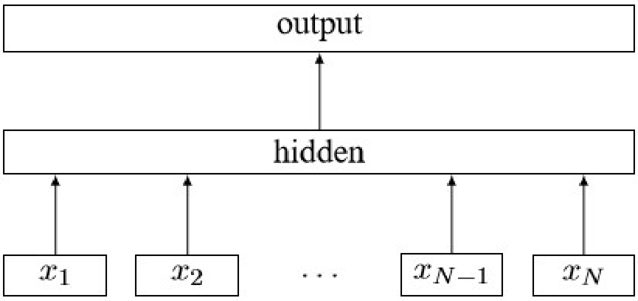


Figure 6.1: FastText Model Architecture [Mikolov et al., 2013].

6.3.4 Hierarchical Softmax

In FastText **hierarchical softmax** is used at the output layer [Mikolov et al., 2013]. To understand what is hierarchical softmax, first we need to know about the softmax function.

- **Softmax :** Usually, in the case of multiclass classification using neural network, softmax is used as the activation function at the output layer. In this case, the labels are encoded using one-hot-encoding. For example if there are seven words, then each of them would have the one-hot-encoded form as *[1000000], [0100000], [0010000], etc.* After optimizing the network if the value obtained at the output layer is as *[0.23 0.45 0.21 0.47 0.43 0.33 0.24].* Then how we will relate this with the words we are having. This is where softmax comes into picture. The formula for softmax is as follows [Mahmoud, 2018].

$$softmax(x_j) = \frac{e^{x_j}}{\Sigma_i e^{x_i}}$$

The softmax function will provide the probability for each word to be related to the corresponding text. In the specified example the value obtained for the first label is 0.23 (which is the exponential transformation of the output from the previous layer) and this needs to be converted into a probability value by using the softmax function. So it is calculated as follows:

$$softmax(label1) = \frac{0.23}{0.23 + 0.45 + 0.21 + 0.47 + 0.43 + 0.33 + 0.24}$$

During training, the hidden layer weights are refined till the target label shows higher probability. As in softmax the probabilities of each labels is presented as a list, inorder to find the label having the highest probability the entire list is to be traversed. Also in FastText, as the output layer represents all the words in the vocabulary including the words prefixed with *"__label__"*, finding the probabilities of the labels needs an exhaustive search of the entire list. So the computational complexity would be *O(n)*, if n is the total number of unique words in the vocabulary.

- **Heirarchical Softmax :** In case of FastText classification, we need to know only about the probability of the words prefixed with *”__label__”* in a given context. So searching the entire list of words for the labels would be cumbersome if we are having a very large vocabulary of words. Inorder to fix this, FastText uses a hierarchical softmax architecture at the output layer.

  Inspired by the binary tree created by Morin and Bengio [2005], FastText used a hierarchical softmax architecture at the output layer, in which each words in the vocabulary is associated with a leaf node of a binary tree. Let us deeply understand this architecture with the help of an example. If the unique words in the data are [*”hindi”, ”gujarati”, ”language”, ”foul”, ”__label__comm”, ”diabetes”, ”__label__cd”, ”disease”*]. As there are eight words in total, it can be represented using a binary tree having eight (ie 2^3) leaf nodes with each leaf node having a unique path. Therefore the depth of the balanced binary tree would be 3 (ie $log_2 8$). So if there are N unique words in the vocabulary, then we need a binary tree of depth $log_2(N)$. The structure of the tree is always stable, thus the path from root to a particular word is fixed.

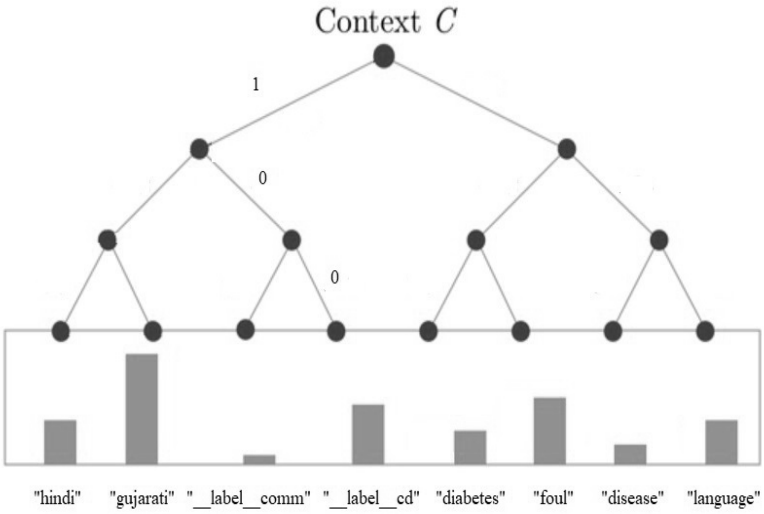


FIGURE 6.2: Example of Hierarchical Softmax Architecture

But how each words are assigned to the leaf nodes?

As there are only eight words, each word can be represented using eight different representation using 1s and 0s. So for forming eight (ie 2^3) different representation, we need only 3 bits like 000, 001, 010, 011, 100, etc. Arrangement of the words in the binary tree depends on this binary representation. 1 denotes movement from a node to its left node and 0 denotes movement from a node to its right node. So on the example tree, the word *"__label__cd"* is at 100 and the word *"__label__comm"* is at 101. Thus by looking at the binary representation of the labels, we can find its corresponding probability without doing an exhaustive search of the entire list. Thus the complexity of finding the probability of a particular label has been reduced from $O(n)$ to $O(log_2n)$

How probability of labels are calculated in hierarchical softmax?

- In hierarchical softmax, each nodes other than the leaf nodes are represented as vector $v_n^{'}$.
- We know that the sum of probability of a branching decision from a particular node in a binary tree is 1. That is if the probability for going right is p, then for going left is *1-p*. Thus the sum of probability to reach the leaf nodes will be 1.
- When using the traditional softmax, the probability of a label or word is obtained by finding the dot product of hidden layer vector h with the respective word vector $v_w^{'}$ and then applying softmax transformation on it [Mahmoud, 2018]. Instead in hierarchical softmax, the probability of a word is obtained by multiplying the probabilities at each internal node on the path to the respective word.
- In the above example, the path to reach the word *"__label__cd"* is 100. If the nodes traversed to reach *"__label__cd"* are v_1, v_2, v_3 and corresponding vector representation are $v_1^{'}$, $v_2^{'}$, $v_3^{'}$, then the probability of the word *"__label__cd"* is as following:

  $$p(\text{"\_\_label\_\_cd"}|context) = (1 - \sigma(b_1 + v_1^{'}.h(x))) * \sigma(b_2 + v_2^{'}.h(x)) * \sigma(b_3 + v_3^{'}.h(x))$$

  Therefore, in general, the formula for finding the probability of a word can be written as follows. If the leaf node is at depth d+1 with parents $n_1, n_2, ..., n_d$, then its probability is [Joulin et al., 2016]:

  $$p(n_{d+1}) = \Pi_{i=1}^{d} p(n_i)$$

6.3.5 Hyperparameters

The important hyperparameters of FastText algorithm are as follows:

- wordNgrams: This denotes the number of ngrams to be considered that constitutes a text.

- Learning rate: Learning rate denotes the speed that the algorithm takes to learn the relation between the context and the labels.
- Dimension: The word embedding dimension.
- Epochs: The number of epochs to train the network.
- Loss function: The activation function to be used while training the algorithm. It can be negative sampling, hierarchical softmax or softmax.

6.4 Related Works

- **Yoshida et al. [2011]:**

  The author has used SVM [Fan et al., 2008, Joachims, 1998], for text classification, inorder to capture the non linearity in the text data and compared the results using different Bag-Of-Words concepts like TF-IDF, LSI and multiword. From the results it has been concluded that the classification results depends on the content of the text data and LSI has given more accurate results than TF-IDF and multiword.

  Discussion: The multiword and LSI has resulted in good performance in terms of underlying semantics of the data as it was capable to capture word to word relation. TF-IDF has provided good statistical quality as it tried to capture the importance of a word in a particular document. But the computational complexity with LSI increases with the number of words in the text. So the model cannot perform well as expected with larger text corpora.

- **Moraes et al. [2013]:**

  As SVM and ANN is capable to capture the non-linear relation between the data and their label, Moraes et al. [2013] has compared the performance of both with text classification. From the results, it has been found that ANN outperformed SVM.

  Discussion: Even though ANN has provided good results, it was not stable all the time as SVM, also the training time for ANN was very much higher than that of SVM. In the presence of noisy data, ANN performance was also affected, unlike SVM.

- **Zhang et al. [2015]:**

  Deep neural networks has been used for different machine learning approaches. Zhang et al. [2015] has used Convolutional Neural Network (CNN) for text classification with ConvNets at character level. Compared different approaches of word representation like Bag-Of-Words with TF-IDF, Bag-Of-NGrams with TF-IDF and average of word embeddings. With CNN, Bag-Of-NGrams with TF-IDF has given good accuracy results. Usage of character level ConvNets has helped to give good result even with less curated user generated data.

Discussion: Even though TF-IDF with Bag-Of-NGrams has given good performance, it was not suitable for larger text corpora that ranges to millions of records. Also, when the text was represented as the average of word embeddings, CNN has given the worst performance.

- **Joulin et al. [2016]:**

  Joulin et al. [2016] has started by saying that even though deep neural network gives good performance, it is very slower compared to linear models when dealing with larger data. The paper attempted to build a text classification model by using a simple neural network with a hierarchical softmax layer at the output layer. This proposed model by Facebook AI Researchers has outperformed the character CNN [Zhang et al., 2015]. The model is called as FastText as it performs the text classification with high accuracy and within lesser time.

  Discussion: As the model represents a word as average of embedding of ngrams, it is capable to handle out of vocabulary words as well, unlike the default Word2Vec. During text classification, the hierarchical softmax architecture helps to train data which is more than one billion words in less than 10 minutes.

- **Armand et al. [2016]:**

  Armand et al. [2016] has compressed the developed FastText model [Joulin et al., 2016], and has suggested the way to optimize it in terms of the memory consumed while running the model without sacrificing its efficiency and speed. Methods like product quantization [Jegou et al., 2010], feature reduction by pruning the vocabulary, using hashing trick and bloom filter [Agarwal et al., 2014] are implemented to optimize the model. The refined model has been compared with the default FastText and CNN results of text classification and has found that the proposed model was performing better.

- **Bojanowski et al. [2017]:**

  In 2017, Bojanowski et al. [2017] has tried to showcase the importance of considering the subword information to form a word or a sentence. The model was developed using a Recurrent Neural Network (RNN) with 650 Long Short Term Memory (LSTM) units with drop out and weight decay. The model which were using the subword information has performed well than the model which were using the words as such.

Discussion: The proposed model which used ngrams were providing good results for ngrams between 3 to 6 and is able to represent the out of vocabulary words as well. Also it has been found that less ngrams (like ngrams=2) is not sufficient enough to capture the subword information.

- **Mikolov et al. [2017]:** Even though the FastText model is performing well in text classification, the model might underperform in presence of data redundancy and this could be dealt with deduplicating the data. Mikolov et al. [2017] has found that data deduplication has helped to capture the relation between different keywords more efficiently and thus produced more accurate results.

6.5 Implementation

This section describes about the implementation details and the results obtained with our data using FastText in detail.

6.5.1 Preparing the dataset

- **Training Data:** At first, we need to prepare the training dataset, from the list of questions asked, by appending the word *"__label__"* to each of the assigned categories as mentioned in the Dataset Preparation chapter. Following shows a part of our training data with labelled questions.

  __label__sm *How many meals would most of the men or women or children or old people would have in a day?*
  __label__sm *What do they generally eat in these meals?*
  __label__sm *Any differences in food for children or the ill persons?*
  __label__sm *Do you think that you eat healthy food?*
  __label__sm *Do you think that others might be considering you as a foody who care for the taste?*
  __label__sm *What is being fat, thin or proper measured body?*
  __label__sm *What activities do every family member do in a day? ....*

- **Exploratory Data Analysis:** It is important to do an Exploratory Data Analysis (EDA), inorder to deeply understand the training data. The main issue that could be faced by a text classification problem is imbalanced classes in the data. For example, if the training data contains more data (say 95%) belonging to a particular class, then the model would produce a biased result towards that class.

And it will not produce a generalized output for the provided new data. Undersampling of majority class or oversampling of minority class needed to be done as required [Han et al., 2005].

During EDA, it has been found that in our training data there is imbalanced classes. As FastText hierarchical softmax architecture deals with this class imbalance by itself, by having more deeper branch for more frequent classes, there is no need to oversample or undersample to make the classes balanced.

- **Preprocessed Data:** Both the list of questions and interview data are preprocessed and randomized inorder to avoid bias. The preprocessed training data is stored as a text file ***dataForClassifn.txt***. The resulting preprocessed training data with the label will be as follows:

  __label__pov *many household member year old younger*
  __label__ss *well wishers find*
  __label__job *service important bank account accident insurance health insurance skill development training access government scheme*
  __label__job *describe look job opportunity get help peer friend relative try employment agency ngo trust agency*
  __label__cd *think beneficial physical exercise reason*
  __label__job *long using phone popular phone*
  __label__comm *people communicate family friend worry kind worry talk.........*

6.5.2 Building FastText

The FastText model which takes the preprocessed data as its input and provides the probability measure of each of the labels to be assigned as output is designed with the following hyperparameters. After optimizing the model with different hyperparameters, following had given a more convincing categorization during FastText classification.

- Dimension: 20

  As our training data have only 66 questions, after preprocessing there were only 312 unique words. As we have less number of unique words, the dimension of a vector is initialised as 20.

- Epochs: 500

- wordNgrams: 2

The experiment done by Bojanowski et al. [2017] has concluded that the ngrams between 3 to 6 were able to capture the semantics and give more generalised result. But with our interview data, as we have very less training data, the FastText model with wordNgrams=3 were not giving good classification result and therefore took wordNgrams=2.

- Learning rate: 0.01
- Loss function: hs

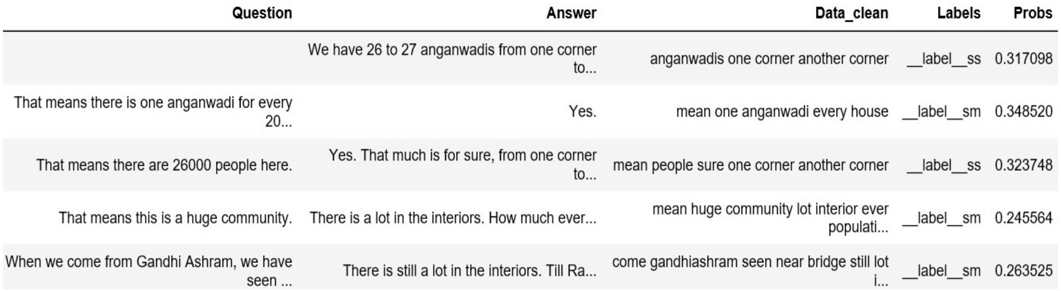
The model is trained on each unique words in the vocabulary, including the label. A 10 dimensional word embedding for each of the unique words are formed based on their context. Following python statement is executed to train the FastText model.

model = fasttext.train_supervised(”dataForClassiffn.txt”, lr=0.01, dim=20, epoch=500, word_ngrams=2, loss=’hs’)

After training, the labels are predicted for the preprocessed interview data with the following python statement, where *testdata['Data_clean].tolist()* is the list format of the preprocessed data.

labels = model.predict(testdata['Data_clean'].tolist(),k=1)]

Now the *labels* variable will have the label and its corresponding probability to be assigned as the respective text's label. The resulting classified data is as follows.



Question	Answer	Data_clean	Labels	Probs
	We have 26 to 27 anganwadis from one corner to...	anganwadis one corner another corner	__label__ss	0.317098
That means there is one anganwadi for every 20...	Yes.	mean one anganwadi every house	__label__sm	0.348520
That means there are 26000 people here.	Yes. That much is for sure, from one corner to...	mean people sure one corner another corner	__label__ss	0.323748
That means this is a huge community.	There is a lot in the interiors. How much ever...	mean huge community lot interior ever populati...	__label__sm	0.245564
When we come from Gandhi Ashram, we have seen ...	There is still a lot in the interiors. Till Ra...	come gandhiashram seen near bridge still lot i...	__label__sm	0.263525

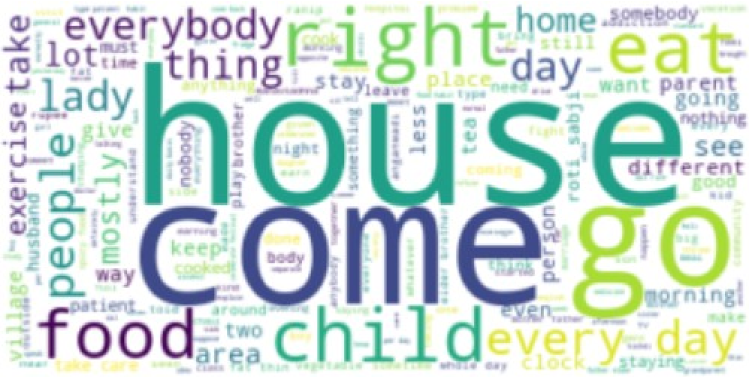
Figure 6.3: Classifed data using FastText

As there are so many responses in the data, that does not belong to any of the seven categories, their assigned label probability is very less compared to the one that surely belongs to one of the category. The probability of the predicted labels has ranged from 0.1770 to 0.9327.

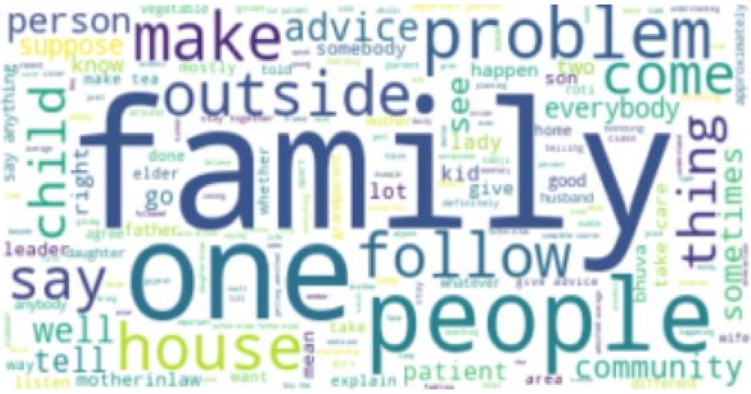
6.5.3 Result Visualization using WordCloud

WordCloud is a visualization technique that can be used to visualize the most prominent keywords in a piece of text corpora. The text or the keywords that occurred the most will be shown in a larger size. Like that each and every word in the corpora will be shown in different sizes depending on its number of occurrence. Following are the WordCloud visualization of the codes from the different categories of text corpora classified by FastText.

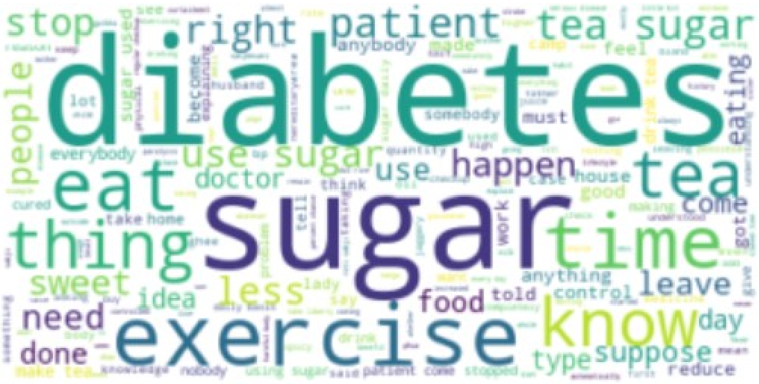
- **Self Management:** Following WordCloud visualization shows that self management data talks more about their house, child, exercise routine, food habits, daily routine, etc.



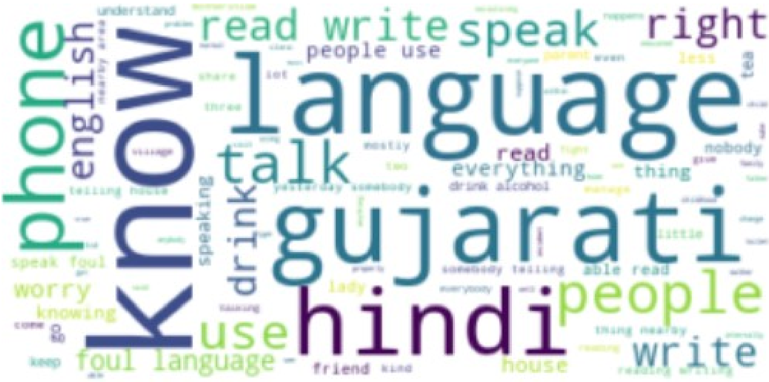
- **Social Support:** Following WordCloud visualization shows that social support data talks more about their family, people, advices given, leaders, community, etc.



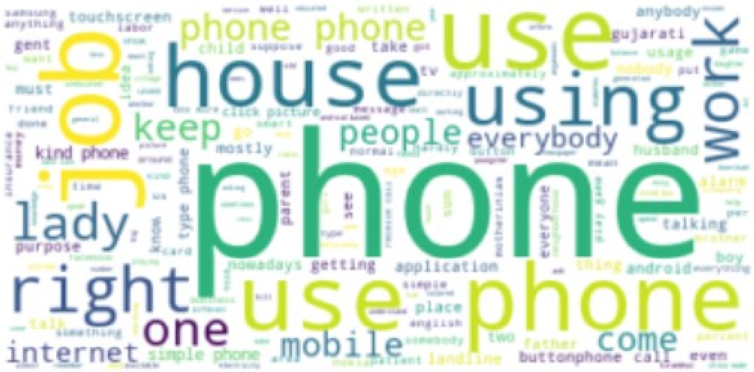
- **Chronic Diseases:** Following WordCloud visualization shows that the data categorised as chronic diseases talks more about diabetes, sugar, exercise, food habits, patients, etc.



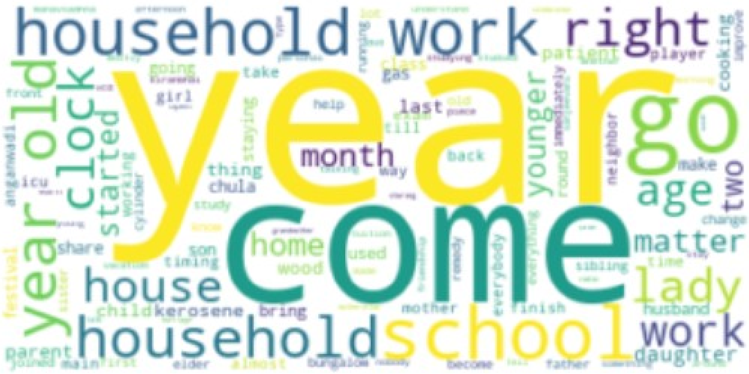
- **Communications:**Following WordCloud visualization shows that communication data talks more about their language (Gujarati or Hindi), whether they know to read and write, phone usage, etc.



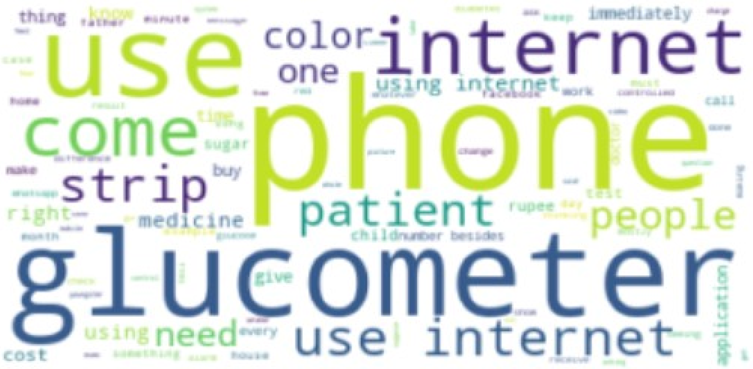
- **Job and Income:** Following WordCloud visualization shows that job and income data talks more about their work, type of phones that is being used (touch screen or button phones), etc.



- **Poverty:** Following WordCloud visualization shows that data categorised as poverty talks more about their household things, work, education, age, fuel used, etc.



- **Tools and Design:** Following WordCloud visualization shows that deal with tools and design talks more about devices like glucometer, strips, its cost, phone, internet, etc.



6.6 Conclusion

FastText with WordCloud visualization helped to find the most prominent keywords associated with different categories. But still it is not complete in the sense that it does not help to understand the details in deeper. For example, in the WordCloud visualization of data categorised as Chronic Diseases, the word *"diabetes"* is shown as the most prominent one. Also we are not able to see any other disease in the respective WordCloud visualization. This does not mean that diabetes is the most prevailing disease in the respective community. But the result has been shown like that in the WordCloud, because the interviewer was more biased towards enquiring about the diabetes prevalence in the community.

Chapter 7

Analysis Engine

7.1 Introduction

As specified in the previous chapters, neither Word2Vec nor FastText alone were able to capture the underlying theme of our interview data properly. Both the results were not capable to understand about the lifestyle of the people in the community properly. Therefore this chapter introduces an analysis engine that tries to drill deeper into the data by feeding the results of FastText into Word2Vec and provide the results in the form of graph visualization.

7.2 Implementation

This section describes about the steps followed to implement the analysis engine by combining FastText and Word2Vec.

7.2.1 Extracting keywords from FastText

Perform text classification as before using FastText. This helps to obtain the prominent keywords from each categories. Most of the obtained keywords contains many non prominent words as well which have number of occurrences as 1, 2 or 3. Therefore from the keywords that were visualised using WordCloud, only those keywords that had occurred more than 5 times from the respective categories are extracted and converted into a list format. Following is an example list of codes from the category of *"Tools and Design"*.

['internet', 'use', 'phone', 'glucometer', 'like', 'come', 'using', 'strip', 'patient', 'people', 'color', 'one', 'need', 'cost', 'medicine', 'right', 'know', 'get', 'time', 'application', 'buy', 'give', 'immediately', 'thing', 'test', 'rupee', 'number', 'sugar', 'child', 'besides', 'doctor', 'call', 'work', 'every', 'make']

7.2.2 Retrieve trained Word2Vec

The Word2Vec model that has been trained on our preprocessed interview data were saved before, in a file *"response2vec300D20.w2v"*. While building the analysis engine, there is no need to retrain our data to the Word2Vec algorithm. Instead, we can retrieve the already saved model for our task. Following is the Python statement used to load an already saved Word2Vec model *response2vec300D20.w2v.*

response2vec = w2v.Word2Vec.load("response2vec300D20.w2v")

7.2.3 Deep Drilling

For each prominent codes obtained from each categories, let us now try to obtain the relevant subcodes by drilling one level deeper into the data. As the data used in FastText and Word2Vec has been passed through the same preprocessing steps, the obtained codes from FastText classification will have its corresponding vector representation in Word2Vec. Now by giving this codes as an input to the trained Word2Vec model helps to find the related subcodes. Thus we can obtain the result, with code and subcode relation, that will help for theory formation from the unstructured data (Refer figure 2.1).

7.2.4 Visualization using Graphviz

Graphviz is a library package that can be used in Python (Graphviz), to create a graph object with nodes and edges for visualizing the relation between different nodes. In our project, the graph has been built with the root node as one of the seven categories and the subsequent nodes representing the codes and the subcodes. For example if the list of codes that has been obtained from the category *"Tools and Design"* is *['internet', 'use', 'phone', 'glucometer', 'like', 'come', 'using', 'strip', 'patient', 'people', 'color', 'one', 'need', 'cost']*, then we need to drill deeper into each of these codes to know more about those. For example, following is the graph obtained when drilled deeper into the code *"internet"* which comes under the *"Tools and Design"* category.

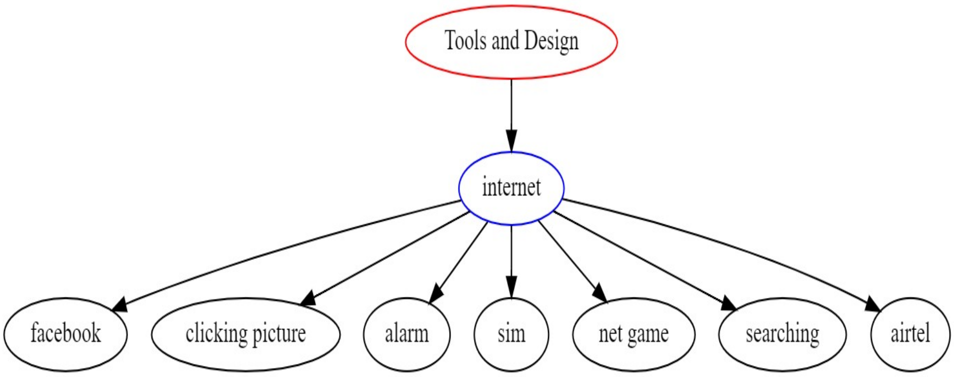


FIGURE 7.1: Example of Graphviz Visualization.

By viewing the graph 7.1 the underlying theory that we would figure out would be as follows:

"The people in the community is using internet. They are using internet for Facebook, clicking pictures, playing games and internet searching. They are having internet with Airtel connection."

Let us now check whether our assumption confirms with our actual data. Following are the few dialogues from the respondents in the interview data that are related with the internet usage.

*"Mainly they use **internet**, like seeing things on internet, listening to music, **do searching**. They also use it for **clicking pictures**, to talk on the phone.", "He **clicks pictures and post it on the internet**.", "They keep touch-screen mobile. Boys have even started using **internet** also.", "They watch videos also. Also for **sending and receiving pictures** of each other", "Some people who are using internet use Vodafone. There is also **Airtel**", "They use phones for downloading movies and songs or like you said they use **Facebook**, WhatsApp also."*

From the above dialogues from the interview data, it is obvious that the model helped to capture most of the related keywords correctly as we assumed.

7.3 Results and Evaluation

As we do not have any test data to quantify the obtained results, for evaluating our model we need to randomly compare different scenarios with our the result and the actual responses from the interview data. Following are the codes obtained from the respective categories using FastText and the graph visualization obtained when tried to drill deeper into some of the example keywords.

- **Self Management:**

  Codes:*['earn', 'day', 'bring', 'child', 'people', 'right', 'eat', 'every', 'fat', 'thin', 'proper', 'measurement', 'body', 'give', 'food', 'spicy', 'house', 'mean', 'cooked', 'exercise', 'addiction', 'everybody', 'anybody', 'rupee', 'one', 'time', 'like', 'made', 'thing', 'come', 'need', 'sometimes', 'whole', 'vegetable']*

  Graph Example 1: *Self Management − > addiction*

  Assumption from the Graph(7.2): *" The people in the community have addictions of beedi, tobacco, gutkha, miraj and masala. And these results in physical suffering."*

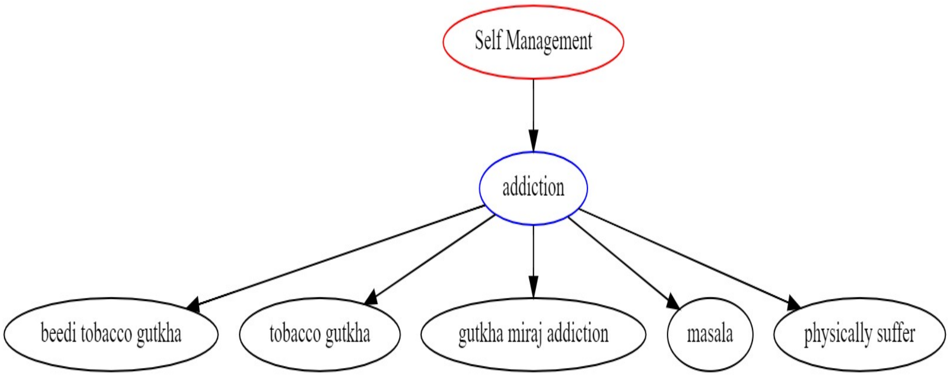


FIGURE 7.2: Graph Visualization of Self Management − > Addiction.

  Actual Responses from the Data: Let us now check whether our assumption is right by comparing it with actual data. Following are the responses from our data.

  *"Yes, some of them have these kind of **addictions** like eating **Gutkha**.","Not chikni. But yes, some people smell that smelling powder chikni. Old people smell that, but the younger ladies eat **Gutkha and Miraj**. They do have these addictions.","There are about 5 percent of ladies who eat Gutkha","Last time to leave their addiction of **Gutkha, pan masala** and alcohol , 5 to 6 people were admitted at the hospital."*

  Graph Example 2: *Self Management − > earn*

  Assumption from the Graph(7.3): *"The earning is hard and they do rag picking, luggage transferring and farming in village for their earning."*

  Actual Responses from the Data: *"No, no, but some people may have lot of trouble. Those people who go for **collecting recyclable materials from garbage**, for them it is like daily earning and daily eating with that","So mostly they do those around only like taking **luggage from one place to the other** or if someone wants to send something from one place to the other.","They do*

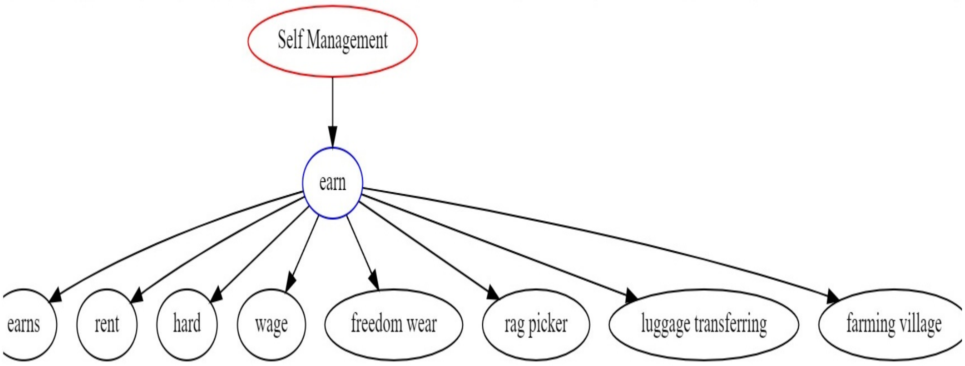


FIGURE 7.3: Graph Visualization of Self Management − > Earn.

farming in the village *and nothing else","And those who are on daily* ***wages*** *get around 6000 or so."*

- **Social Support:**

  Codes:*['family', 'outside', 'like', 'give', 'advice', 'problem', 'say', 'person', 'one', 'people', 'thing', 'follow']*

  Graph Example 1: *Social Support − > advice*

  Assumption from the Graph(7.4): Unlike the previous graphs, the follwoing graph does not help to reach into a clear conclusion about the code *"advice"*. But still from the shown subcodes *"peaceful entirely"*, *"atmosphere loving"*, *"listens totally filmy"*, it says that *"the atmosphere is good and the advices are followed by everyone"*. But the word *"forced"* implies that *"advices are not followed with their own willing"*.

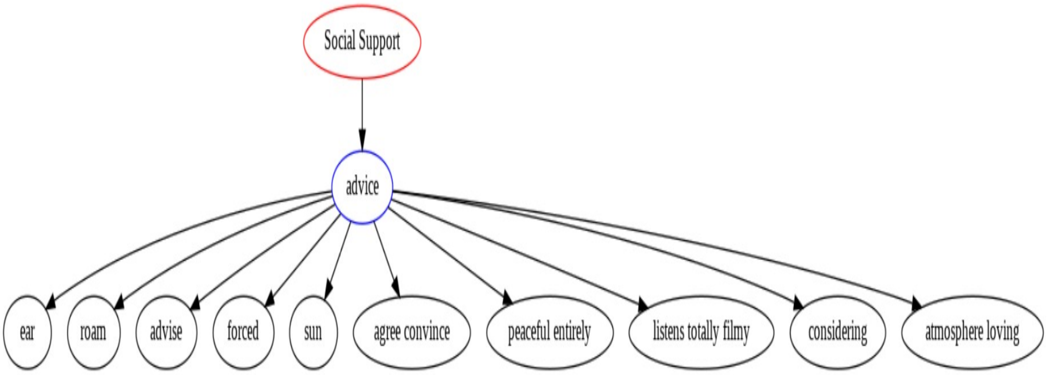


FIGURE 7.4: Graph Visualization of Social Support − > Advice.

Actual Responses from the Data: *"Because it has been 5 to 6 years, so we talk to each other like our own sisters. And we always* ***give good advices."****,"Advices are like this itself, but* ***nobody listens to."****,"I feel that as they are not giving good advice, they have become Bhuva.","We give this advice to them and tell them to* *continue their routine. But once they go home and start their routine, then they* ***don't pay much attention*** *to it."*

- **Chronic Diseases:**

  Codes: *['eating', 'eat', 'sugar', 'less', 'people', 'sweet', 'take', 'doctor', 'know', 'tell', 'thing', 'like', 'idea', 'diabetes', 'happens', 'time', 'got', 'everybody', 'say', 'every', 'day', 'tea', 'stop', 'reason', 'exercise', 'use', 'body', 'right', 'using', 'drink', 'think', 'get', 'control', 'told', 'food', 'used', 'leave', 'good', 'happen', 'anybody', 'want', 'need', 'done', 'must', 'suppose', 'nobody', 'making', 'made', 'give', 'make', 'disease', 'otherwise', 'reduce', 'type', 'anything', 'somebody', 'house', 'come', 'patient', 'daily', 'basis']*

  Graph Example 1: *Chronic Diseases − > disease*

  Assumption from the Graph(7.5): *"The community people has infectious diseases, skin diseases like psoriasis, chronic diseases, cv stroke, asthma because of smoking."*

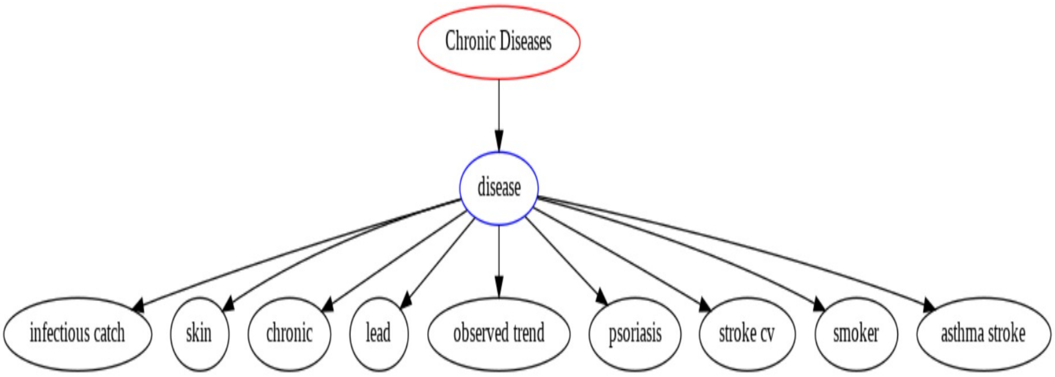


FIGURE 7.5: Graph Visualization of Chronic Diseases − > Disease.

Actual Responses from the Data: *"So the outside* ***infections*** *are caught for them very fast and their houses are very close to each other.","Yes, they catch* ***infectious diseases*** *more. We see more of infectious diseases over here.","There are 25 to 30 patients, who are elderly people with* ***chronic disease****.","****Skin disease patients and asthma patients*** *are more on Tekra.","Uncontrolled diabetes, paralysis, hyper tension. There are patients with hemorrhage,* ***severe stroke****, asthmatic patients are there"*

Graph Example 2: *Chronic Diseases − > exercise*

Assumption from the Graph(7.6): *"The community people do yoga, walking and go to gym. Exercise is done automatically. Exercise is required for some."*

Actual Responses from the Data: *"Yes , we have made arrangements to do exercises and lot of people comes to* ***do exercises****.","We don't do exercise. My*

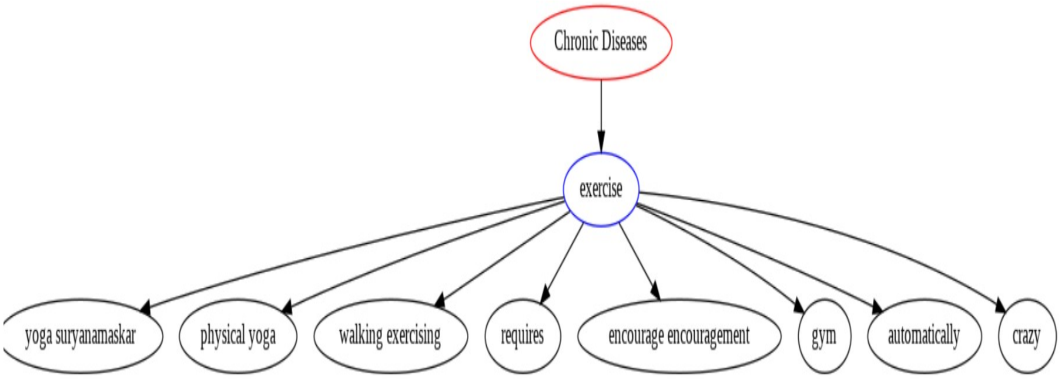


FIGURE 7.6: Graph Visualization of Chronic Diseases − > Exercise.

exercise is ***automatically done*** *while I am working","Those who do labor work is like a* ***physical exercise*** *only, but those who live sedater lifestyle, they live that way only.","Some children do like they come here and* ***do yoga or physical exercise****. But they don't take it seriously.","Yes, he* ***requires*** *doing exercises. But he is not doing"*

- **Communications:**

  Codes: *['language', 'know', 'hindi', 'gujarati', 'read', 'write']*

  Graph Example 1: *Communications − > language*

  Assumption from the Graph(7.7): *"The community people know hindi, gujarati and english. The also speaks foul language. They know to read and write. There is change in dialects"*

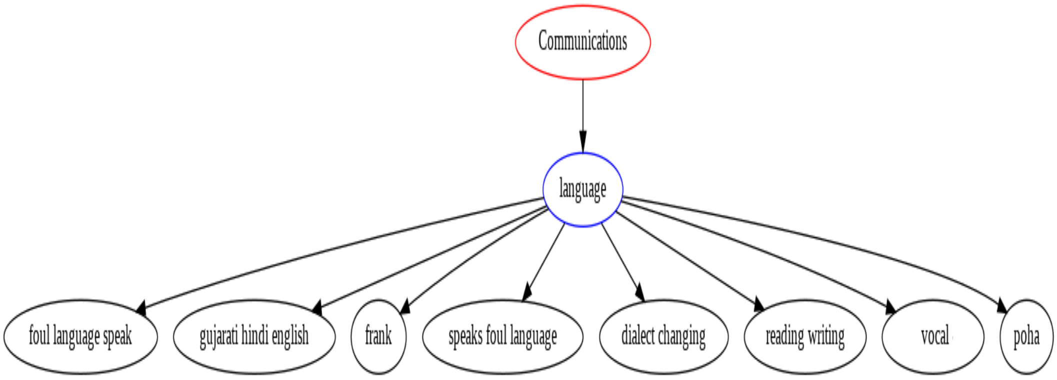


FIGURE 7.7: Graph Visualization of Communications − > Language.

Actual Responses from the Data: *" Lot of* ***foul language*** *is spoken in the house. So right from childhood kids also get influenced by this.","Even to their parents they speak* ***dirty foul language*** *which we cannot even listen to.","They keep* ***Hindi*** *as their language. Sometimes* ***Gujarati*** *is there. And* ***English*** *is also there, but very rarely. Ladies mostly use* ***Gujarati and Hindi*** *only.","****Hindi and***

Gujarati*. These two languages.","Because in Gujarat the are many districts and villages, and according to that the language forms of Gujarati also keep changing and* ***dialect will also change.****"*

- **Job and Income:**

  Codes: *['people', 'use', 'phone', 'house', 'kind', 'simple', 'touchscreen', 'boy', 'right', 'even', 'keep', 'type', 'must', 'mobile', 'landline', 'box', 'lady', 'using', 'usage', 'game', 'picture', 'getting', 'job', 'know', 'one', 'child', 'talk', 'work', 'son', 'husband', 'like', 'nobody', 'anybody', 'call', 'tv', 'everybody', 'android', 'father', 'get', 'internet', 'two', 'take']*

  Graph Example 1: *Job and Income − > phone*

  Assumption from the Graph(7.8): *"The people in the community uses phones with touchscreen and also button phones. It includes Nokia, Samsung and china made phones. They use Facebook from phones."*

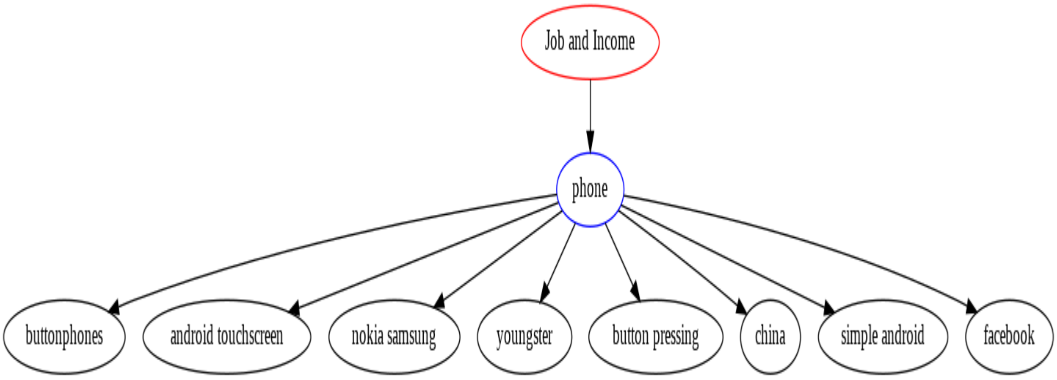


FIGURE 7.8: Graph Visualization of Job and Income − > Phone.

  Actual Responses from the Data: *"Mostly all* ***youngsters*** *have phone with them and there will be one more phone with one of the other family members.","Company hardly any, but* ***Chinese made phones****.", "Yes,* ***Nokia and Samsung****, but they are very less. We see mostly* ***china phones****.","Yes.* ***Button pressing*** *phones. Some people have those* ***china made touch-screen*** *phones.","They use phones for downloading movies and songs or like you said they use* ***Facebook****, WhatsApp also"*

  Graph Example 2: *Job and Income − > work*

  Assumption from the Graph(7.9): *"The people in the community do bungalow floor cleaning, masonry work, maid job, construction work, ride pedal rickshaw and contract work for their earning."*

  Actual Responses from the Data: *"They drive those* ***pedal rickshaw****","In municipality, they* ***clean the floor*** *or sit in office","In the morning ladies finish*

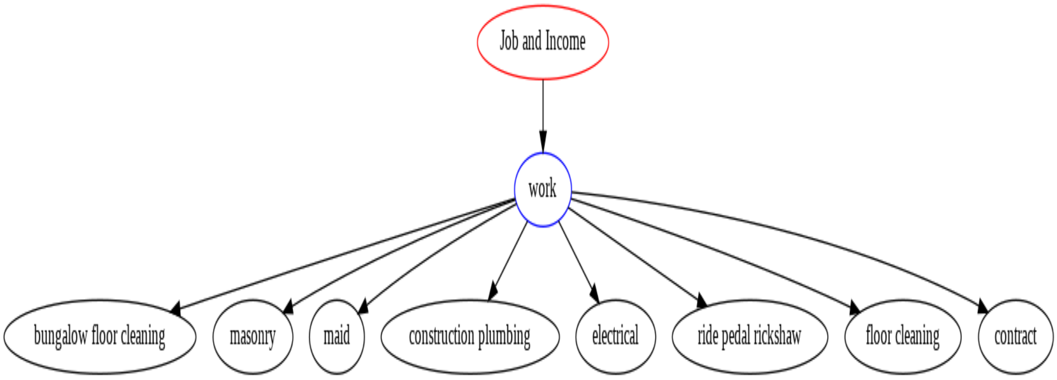


FIGURE 7.9: Graph Visualization of Job and Income – > Work.

their house hold work and go for jobs like ***cleaning floor at bungalows*** *or for collecting recyclable materials from garbage","The majority of people in that area do the* ***plumbing and construction*** *work only"*

- **Poverty:**

  Codes: *['household', 'age', 'year', 'old', 'work', 'school', 'fuel', 'device']*

  Graph Example 1: *Poverty– > fuel*

  Assumption from the Graph(7.10): *"The people in the community uses primus, chula, sigdi, lpg gas, kerosene, wood pieces as fuel."*

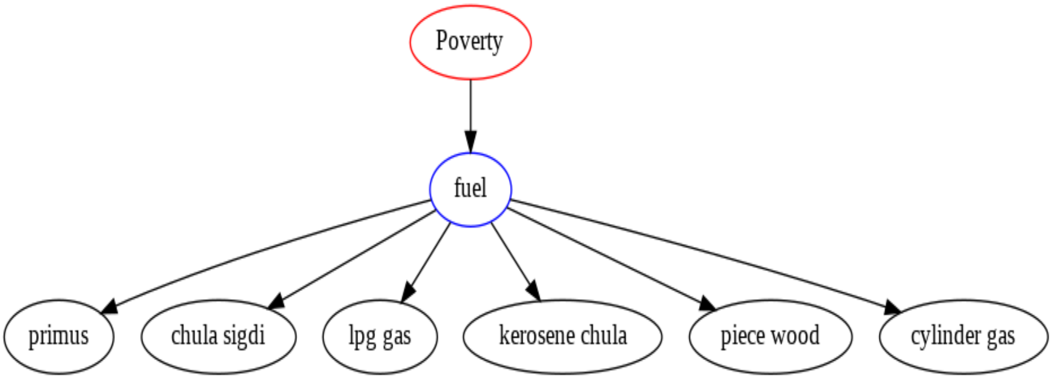


FIGURE 7.10: Graph Visualization of Poverty – > Fuel.

Actual Responses from the Data:*"They use* ***gas and Chula****", "Many use* ***primus****, many use Chula and now a days in many houses gas connection has also come.","They all have* ***LPG****, but they use it less so that the cylinder run for 2 to 3 months."," They also use* ***kerosene****, as they get kerosene using their BPL card","They bring wood pieces and cook on Chula.", "****Gas cylinder****."*

Graph Example 2: *Poverty– > device*

Assumption from the Graph(7.11): *"The people in the community have electronic devices like fridge, tape and cooler."*

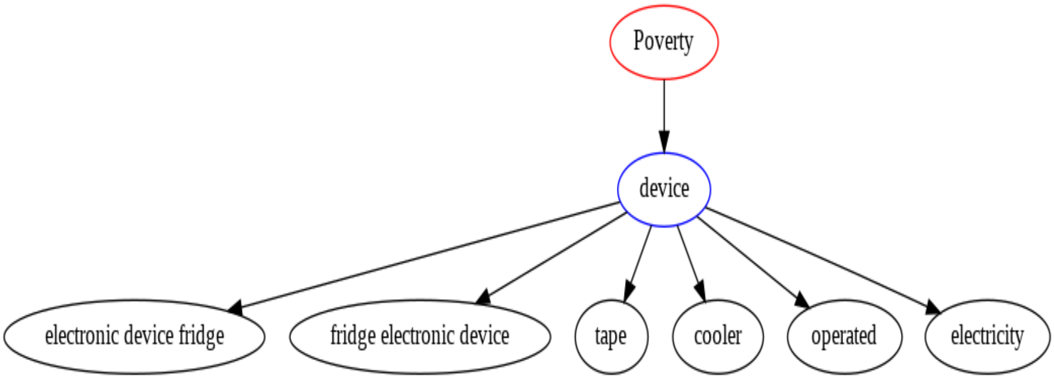


FIGURE 7.11: Graph Visualization of Poverty − > Device.

Actual Responses from the Data:*"In electronic devices they have T.V,* ***fridge, radio and tape****. They use these things and also mobile.","Only the basic items like fan, lights ,T.V. And if they are living even better then they might have* ***fridge*** *also.","Fridge,* ***air coolers****, fan, then iron box is there. Now majority of them have LCD and LED TVs in their house.","They have T.V, cooler and also have vehicles","In electronic devices they don't have anything except fan."*

- **Tools and Design:**

  Codes: *['internet', 'use', 'phone', 'glucometer', 'like', 'come', 'using', 'strip', 'patient', 'people', 'color', 'one', 'need', 'cost', 'medicine', 'right', 'know', 'get', 'time', 'application', 'buy', 'give', 'immediately', 'thing', 'test', 'rupee', 'number', 'sugar', 'child', 'besides', 'doctor', 'call', 'work', 'every', 'make']*

  Graph Example 1: *Tools and Design − > glucometer*

  Assumption from the Graph(7.12): *"The glucometer is used for testing hypoglycemia. It is cheap and needs additional strip."*

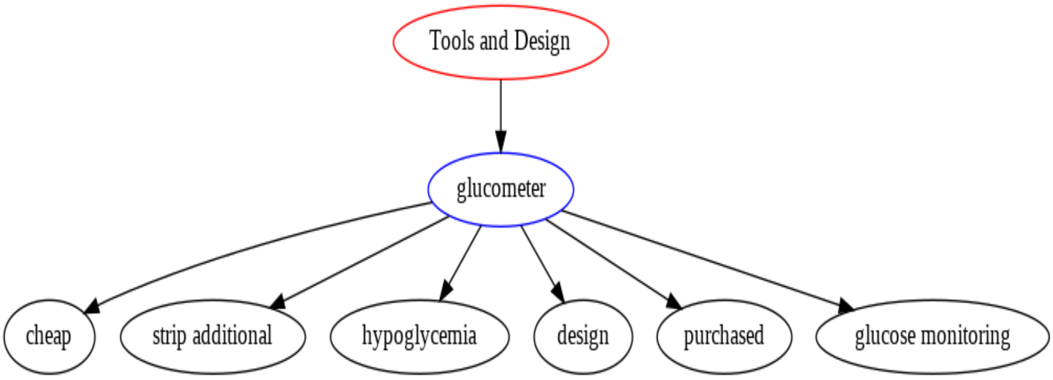


FIGURE 7.12: Graph Visualization of Tools and Design − > Glucometer.

Actual Responses from the Data: *"That test costs 18 rupees.', 'It has to be* ***purchased*** *just once. After that no charges. And it cost around 12,00 to 1500* *rupees.","No, they don't have. They* ***cannot afford that glucometer*** *in their house.","Yes,* ***some of them buy glucometer*** *for themselves at home.","We tell them to keep a glucometer and if they feel like sweating or giddy or unrest, then need to check immediately.","Because* ***the cost of glucometer*** *is not too much. It comes with free strips. So, they buy it. And* ***additional strips*** *are bought at a cost of 300 to 400 rupees for 25 strips."*

7.4 Discussion on Results

- **Word2Vec :** When we were using Word2Vec alone, it is mandatory to know the right code to be given as its input, without any change in spelling. The code to be provided should be there in the preprocessed text data. Otherwise it would not provide the result. For example, if we want to know about the *electronic devices* present in the houses, then naturally as a human we would give *"electronic devices"* as the input. But the preprocessed data has tokenized and lematized words. Therefore we need to give either *"electronic"* or *"device"* as input.

- **FastText :** Using FastText alone has helped to classify the data into seven categories. When we are getting the keywords from only the categorised questions, it might not be enough to describe the theme *"overall well-being"* of the community. The FastText classification help to get the keywords from the questions as well as answers. But the WordCloud visualization was not enough to reach to a conclusion about those seven categories.

- **FastText and Word2Vec :** The model built by pipelining the results of FastText into trained Word2Vec is able to give more meaningful output than when used those models individually. When the codes obtained from FastText (by including the questions and the answers) has been given as input to the Word2Vec, the problem of using the wrong word as input has been solved by this model. Also the model has provided the subcodes based on the respective categorised data as the result.

  By going deeper into the codes obtained from different predicted classes, an important point to note is that, even though most of the results confirms with the actual data, some of the results are not meaningful enough to reach into a conclusion. This is because of the reason that the interview data is not curated and even after preprocessing it contains many words that are not domain specific. The model has vectorised all the words - either it can be domain specific that helps to define the lifestyle of the community or not. Creating a domain specific vocabulary as suggested by Ghosh et al. [2016] can be a solution to this problem.

7.5 Generalised Framework

We had successfully managed to develop a generalised framework that have the following functionalities:

- **Importing Libraries :** For smooth working of our model we need the following libraries to be imported using the following statements:

  ```
  import os
  import docx2txt

  import pandas as pd
  import numpy as np
  import nltk
  import re
  import csv
  import fasttext
  import graphviz
  from graphviz import render
  from collections import Counter

  import multiprocessing
  import sklearn.manifold
  import gensim.models.word2vec as w2v
  ```

- **Data Preparation :** If we have our unstructured interview data in word format organised in folders, which in turn is in a root folder, then we can convert all those word files into a single tabular form with the following statement.

  ```
  Obj=InterviewDataPreparation('D:/FullData')

  data=Obj.to_dataframe()
  ```

  InterviewDataPreparation is the class specific for preparing data into a tabular form. *'FullData'* is the name of the root folder where the word files are organised in sub-folders. After execution of these two statements *data* will have the whole interview data in tabular form with columns *['Asked by', 'Question', 'Answered by', 'Answer'].*

- **Data Preprocessing :** *DataPreprocess* class is specifically designed for performing the preprocessing task involved in our project.

  Obj1=DataPreprocess()

  data['Answer_clean']=data['Answer'].apply(Obj1.preprocess).apply(Obj1.join)

  The above statement helps to preprocess the data in the column *'Answer'*, and store the result in the column *'Answer_clean'*.

- **Word Vectorization :** The class *Vectorization* helps to vectorize the contents of the provided dataframe, by considering both the *Question* and *Answer* columns.

  Vectorization(dataframe=dataframe_df, filename='response2vec.w2v', epochs=20)

  The resultant vector model will be saved in the provided file *'response2vec.w2v'*, which we can retrieve and use later if needed. The arguments *dataframe,filename* and *epochs* are mandatory. Other optional arguments are *num_features, min_word_count, context_size* and *downsampling* which are same as the hyperparameters specified in section 5.3.2. By default the dimension of the vector is preset as 300.

- **Text Classification :** The class *ClassifyText* helps to classify the interview dataframe into categories depending on the labels provided in the training data. The training data should be in a text file with label prefixed with *__label__* and the corresponding text as a single line as mentioned in section 6.5.1.

  classes=ClassifyText(train_data='training_data.txt',
  class_mappings='class_mappings.txt', dataframe = dataframe_df)

  The arguments *train_data, class_mappings* and *dataframe* are mandatory. The class label mappings file should contain the data of what each label represents, with each line for each labels. For example, in our data, there are seven labels. So the 'class_mappings.txt' file should have the content as following:

  __label__sm *Self Management*
  __label__ss *Social Support*
  __label__cd *Chronic Diseases*
  __label__job *Job and Income*
  __label__pov *Poverty*
  __label__comm *Communications*
  __label__tool *Tools And Design*

  Other optional argumants are *lr, dim, epoch, word_ngrams* and *loss* which are the hyperparameters specified in section 6.3.5. After executing the above statement, the WordCloud visualization of the codes from the respective categories will be displayed, from which we can obtain an overall overview about those categories. Following statement helps to obtain the codes belonging to different categories into a dictionary named *codes*, with keys as the labels and the list of the respective subcodes as its corresponding values.

codes=classes.get_keywords()

- **Analysis Engine :** The class *TextualGraph* combines the functionality of the classes *ClassifyText* and *Vectorization* and provides the graph representation of the codes and subcodes of the respective categories.

  TextualGraph(train_data='training_data.txt',
  class_mappings='class_mappings.txt', data='interview_data.csv',
  filename='response2vec.w2v',pre_trained=False, vector_dim=300, vector_epoch=1,
  min_word_count=3, context_size=7, downsampling=1e-3, fast_text_epoch=500,
  fast_text_lr=0.01, fast_text_ngrams=2, fast_text_loss='hs')

  The arguments *train_data, class_mappings, data* and *filename* are mandatory. The argument *data* should have the name of the *csv* file of the interview data which should have the columns *Question* and *Answer* mandatorily. Others are optional. If the argument *pre_trained=True*, then the already saved pretrained vector from the file *'response2vec.w2v'* is retrieved and used. Otherwise the provided data is trained for vectorization and stored the vectors in the specified *'response2vec.w2v'* file which we can use later. The output graph visualization images will be organized in folders of respective categories inside a root folder *"VisualGraphs"*.

Chapter 8

Conclusion

This project tried to address the overall well-being of underserved communities in Ahmedabad, using a Data Science approach on the unstructured interview data. This is the first time that a machine learning approach has been used to find themes from an interview data. The process involved data translation with transcription and processing them using Word2Vec and FastText.

When the model was built using the Word2Vec and FastText alone, it was not capable enough to understand the data and provide a meaningful result. Therefore tried to build an analysis engine that pipelined the results of the FastText and Word2Vec models. This helped to define the lifestyle of the community more deeper than before. We had successfully managed to design a generalised framework of the model and compared the produced results with the actual data from the interview. The model takes the categorised questions and the interview data in tabular format as input and provides the codes and subcodes as graph visualization. After the model execution, those graph image files will be automatically stored in organised format in a folder.

The results obtained from the model helped to identify most of the themes from the interview data within minutes, which would take hours when manually investigated through the whole interview data. These results can be used to understand the problems in the community and the causes for it, and thus take necessary actions according to that.

8.1 Future Work

As we saw, even though most of the important keywords related with the lifestyle of the underserved community in Ahmedabad has been found using the model, still most of the questions has not been answered properly by it. This is due to the fact that the human responses are not curated and it results in the introduction of unnecessary words into the model.

Building a domain specific vocabulary would help to deal with this problem to a great extent. But creating a vocabulary for the domain ***"Understanding the Well-being"*** could be a very time consuming task, as people in different area will have different kind of lifestyle. With the domain specific vocabulary, the model would only have to vectorize those specific words and thus the model would be free from the intruding unnecessary words.

Appendix A

Project Plan (stakeholders, tasks and deliverables, gantt chart, risk analysis)

A.1 Project Stakeholders

Stakeholders	**Requirements**
Project Supervisor	Well-documented extensive research work
Company	Extracting useful information from the data
Myself	Good balance of exploration and exploitation

A.2 Project Tasks

Milestones
Literature Review
Decide on the approach to be followed.
Decide the evaluation metric.
Prototyping
Evaluation
Iterative Refinement
Result Analysis
Documentation
Submission of the Report

A.3 Risk Analysis

Risk	Type	Likelihood	Impact	Mitigation
Author illness	People	Low	High	In extreme cases apply for an extension of deadline
Dataset corruption or loss	Technology	Low	High	Keep local backups and utilize version control solutions
Insufficient Computing Resources	Technology	Medium	High	Use Heriot Watt servers to run expensive operations
Company Requirement changes	Project	Low	High	Form independent and different deliverable for company and for dissertation. Thesis deliverable will not be changing.
Project schedule slippage	Project	Medium	Medium	Providing sufficient buffer during planning.

A.4 Project Schedule

Task	**Duration**	**Start**	**Finish**
Research and Planning	75d	15/01/19	30/03/19
Literature Review	35d	15/01/19	20/02/19
Research Report Writing	48d	07/02/19	25/03/19
Project Planning	10d	20/03/19	30/03/19
Dataset Preparation	40d	06/05/19	17/06/19
Data Evaluation	10d	06/05/19	16/05/19
Dataset Translation and Transcription	53d	07/05/19	30/06/19
Building Framework	33d	20/06/19	23/07/19
Data Preprocessing	10d	30/06/19	10/07/19
Build Initial Prototype	10d	26/06/19	06/07/18
Prototype Planning and Design	3d	26/06/19	29/06/19
Develop prototype of Word2Vec model	7d	29/06/19	06/07/19
Testing and Evaluation	3d	06/07/19	09/07/19
Develop prototype of FastText classification model	6d	09/07/19	15/07/19
Testing and Evaluation	3d	15/07/19	18/07/19
Combining FastText and Word2Vec model	4d	18/07/19	21/07/19
Testing and Evaluation	2d	21/07/19	22/07/19
Iterative Refinement of the models	25d	09/07/19	03/08/19
Hyperparameter and Architecture Optimization	10d	09/07/19	19/07/19
Visualizing the results	5d	19/07/19	24/07/19
Testing and Evaluation	10d	24/07/19	03/08/19
Result Evaluation	5d	24/07/19	29/07/19
Compile All Results	5d	29/07/19	03/08/19
Conclusions and Findings	6d	04/08/19	10/08/19
Report Submission	21d	25/07/19	15/08/19

A.5 Research and Planning

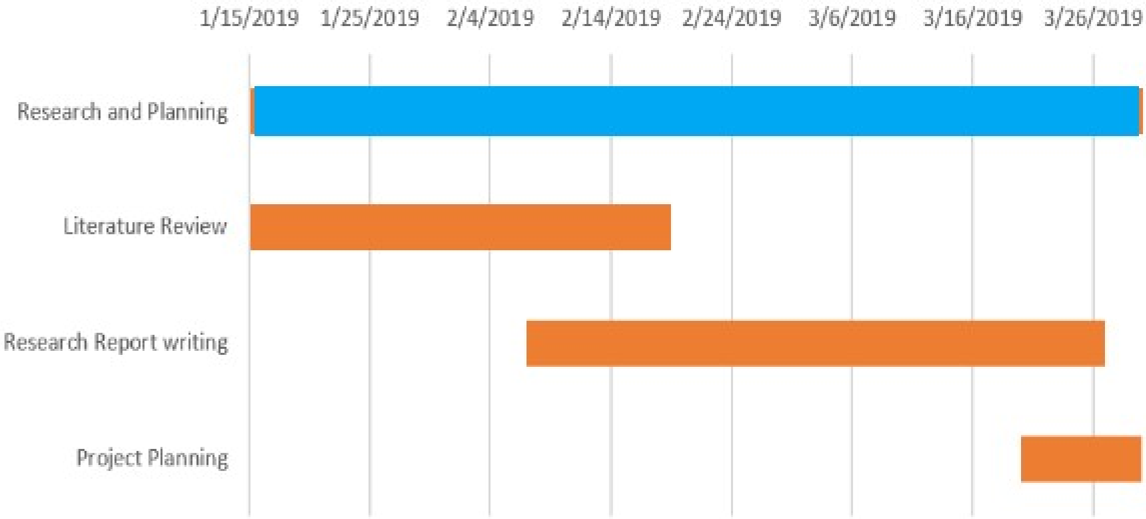


Figure A.1: Research and Planning

A.6 Project Plan

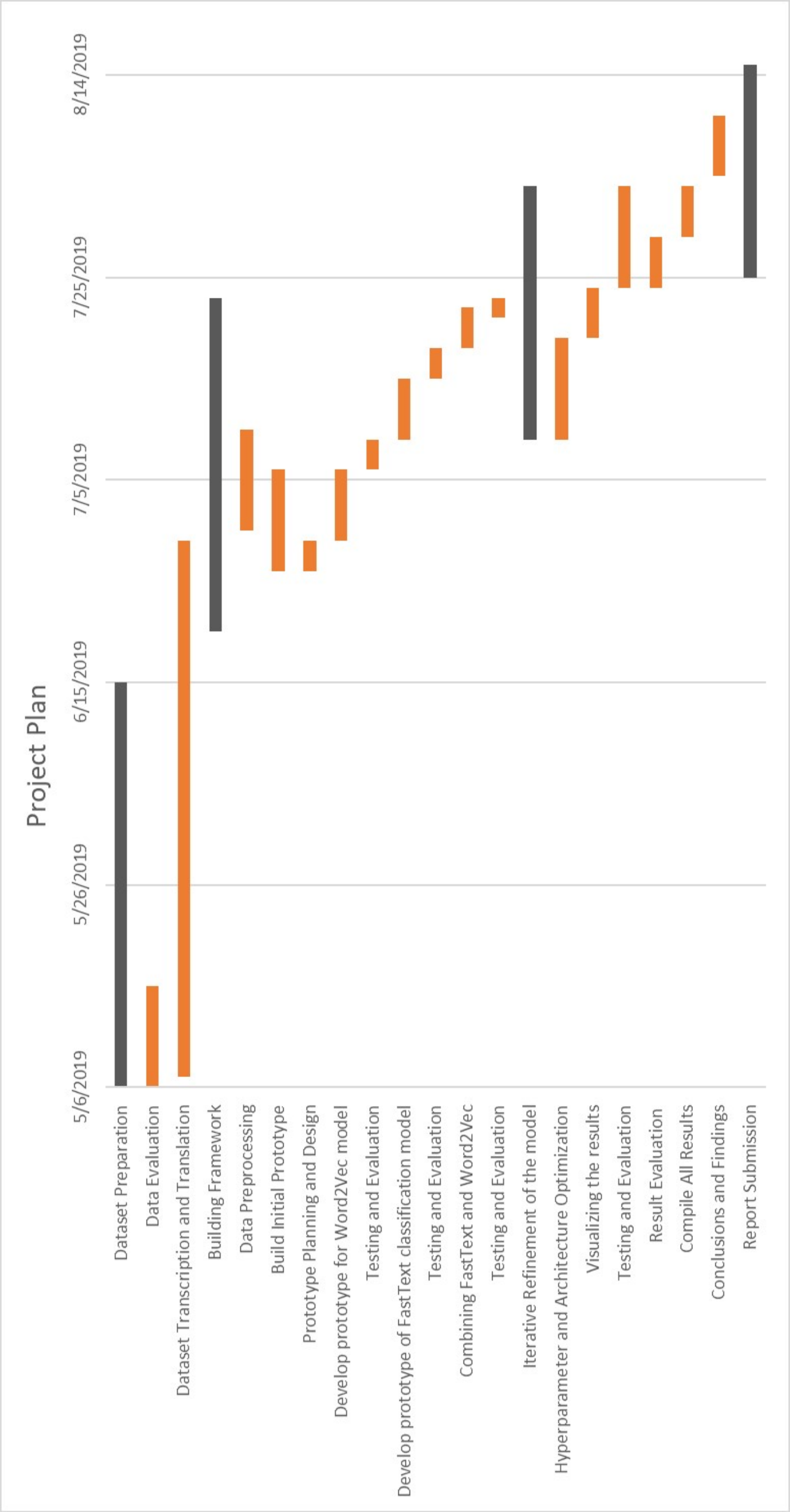


FIGURE A.2: Project Planning

Appendix B

Functional and Non-Functional Requirements

B.1 Functional Requirements

- **Using Word2Vec :** Using Word2Vec help to uncover the underlying theme of the data, as it helps to identify the semantic relationship between different terms used in the data.

- **Text Classification by using a simple ANN :** The questions that were asked by the interviewer were already categorised into seven different categories. Initially the model need to be trained on these categorised questions. Then later the model helps to classify our whole data into these seven categories.

- **Visualization using Word Cloud :** The categorised text data can be visualised using a Word Cloud. This helps to identify the subcodes within the seven main codes.

- **Comparative Analysis :** The interpreted results can be compared with the general lifestyle of the respective community identified from the interview data and can assess the degree of overlapping between them using the necessary benchmarks.

- **Building Analysis Engine by pipelining Text Classification and Word2Vec:** The obtained keywords from the classified data are given as input to Word2Vec. This would help to provide the codes and subcodes from the respective categories.

- **Visualization using Graph :** The relationship between different codes and subcodes obtained from the built analysis engine can be visualised and interpreted using a connected graph with nodes and edges with directions.

B.2 Non-Functional Requirements

- **Technical Documentation and Code Styling :** The programming codes used in the research should be designed and documented as per PEP 8 Style guide standards. Adequate text and illustrations need to be provided for explaining how the code operates.

- **Usability :** The developed model should be easier to learn and implement for another user.

- **Scalability :** The model should be scalable without having any issues when dealing with large datasets.

- **Reliability :** The model should be able to retrieve the information from the supplied preprocessed text data without any failure.

- **Cross Platform Compatibility :** Since the development is done on python engine, it is compatible with Linux, Windows and MacOS operating systems. For compatibility, we need to ensure that python 3.5+ has been installed.

- **Generalised Framework :** A generalised framework of the model need to be developed that can take the categorized question and the interview data as input and provide the graph visualization images as output.

Appendix C

PLES Issues

C.1 PLES Issues

C.1.1 Professional and Legal Issues

- The written code will be tested frequently and ensured that it meets to the high standard by following clean coding and providing sufficient documentation for clarity.
- Any software, third party libraries, or any other products will be implemented only if permitted by their license.
- Use of external information, figure or tables are properly referenced and cited.
- All the work done for this research will be preceded with the approval and knowledge of my supervisor and other stakeholders. They will be informed on a regular basis regarding the progress of the research.
- The dataset used in this research has been provided by **Community Tracks**. The dataset contains the interview data conducted among the rural community related with the subject, and is free from any breach of confidentiality.
- The company has granted legal permission for using the dataset for academic purposes to Heriot Watt University.

C.1.2 Social and Ethical Issues

As the dataset are provided by the company **Community Tracks** and has granted the permission for using it for this research purpose, therefore the project has not came across any social and ethical issues.

Bibliography

Agarwal, A., Chapelle, O., Dudík, M., and Langford, J. (2014). A reliable effective terascale linear learning system. *The Journal of Machine Learning Research*, 15(1):1111–1133.

Aggarwal, C. C. and Zhai, C. (2012). A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer.

Anandarajan, M., Hill, C., and Nolan, T. (2018). *Practical Text Analytics: Maximizing the Value of Text Data*, volume 2. Springer.

Armand, Grave, E., Bojanowski, P., Douze, M., Jégou, H., Mikolov, T., and Joulin (2016). Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Barrasso, T. (2018). Basics of text analysis and visualization. Available at `https://itnext.io/basics-of-text-analysis-visualization-1978de48af47` [Accessed: 28 February 2019].

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Caruana, R. and Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, pages 161–168. ACM.

Chiu, B., Crichton, G., Korhonen, A., and Pyysalo, S. (2016). How to train good word embeddings for biomedical nlp. In *Proceedings of the 15th workshop on biomedical natural language processing*, pages 166–174.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12. 2493-2537.

Creswell, J. W. (2015). *30 Essential Skills for the Qualitative Researcher*, 1st ed.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6). 391-407.

Dumais, S.and Platt, J., Heckerman, D., and Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. *Proceedings of the seventh international conference on Information and knowledge management.* 148-155.

Elliott, V. (2018). Thinking about the coding process in qualitative data analysis. *The Qualitative Report*, 23. 2850-2861.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874.

Fernández-Reyes, F. C., Hermosillo-Valadez, J., and Montes-y Gómez, M. (2018). A prospect-guided global query expansion strategy using word embeddings. *Information Processing & Management*, 54(1):1–13.

Ganegedara, T. (2018). *Natural Language Processing with TensorFlow: Teach language to machines using Python's deep learning library.* Packt Publishing Ltd.

Ghosh, S., Chakraborty, P., Cohn, E., Brownstein, J. S., and Ramakrishnan, N. (2016). Characterizing diseases from unstructured text: A vocabulary driven word2vec approach. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1129–1138. ACM.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning.* MIT press.

Goodman, J. T. (2001). A bit of progress in language modeling. *Computer Speech & Language*, 15(4):403–434.

Grove, S. K., Burns, N., and Gray, J. (2012). *The practice of nursing research: Appraisal, synthesis, and generation of evidence.* Elsevier Health Sciences.

Han, H., Wang, W.-Y., and Mao, B.-H. (2005). Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer.

Jegou, H., Douze, M., and Schmid, C. (2010). Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer.

Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2016). Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Kenter, T. and De Rijke, M. (2015). Short text similarity with word embeddings. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 1411–1420. ACM.

Khatua, A., Khatua, A., and Cambria, E. (2019). A tale of two epidemics: Contextual word2vec for classifying twitter streams during outbreaks. *Information Processing & Management*, 56(1):247–257.

Kreps, G. L. (2005). Disseminating relevant health information to underserved audiences: Implications of the digital divide pilot projects. *Journal of the Medical Library Association : JMLA*. 93(4 Suppl), S68-S73.

Kumavat, D. and Jain, V. (2015). Pos tagging approaches: A comparison. *International Journal of Computer Applications*, 118. 32-38.

Kusner, M., Sun, Y., Kolkin, N., and Weinberger, K. (2015). From word embeddings to document distances. In *International conference on machine learning*, pages 957–966.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436.

Ling, W., Dyer, C., Black, A. W., and Trancoso, I. (2015). Two/too simple adaptations of word2vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1299–1304.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., and Watkins, C. (2002). Text classification using string kernels. *Journal of Machine Learning Research*, 2(Feb):419–444.

Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Mahmoud, H. (2018). The softmax function, simplified. Available at https://towardsdatascience.com/softmax-function-simplified-714068bf8156 [Accessed: 18 July 2019].

Manns, B. J. (2015). Evidence-based decision-making 7: knowledge translation. In *Clinical Epidemiology*. Springer. 485-500.

Mikolov, T. (2007). Language modeling for speech recognition in czech. *Ph. D. dissertation, Masters thesis.*

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781.*

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., and Joulin, A. (2017). Advances in pre-training distributed word representations. *arXiv preprint arXiv:1712.09405.*

Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association.*

Mikolov, T., Kopecky, J., Burget, L., Glembek, O., et al. (2009). Neural network based language models for highly inflective languages. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4725–4728. IEEE.

Moraes, R., Valiati, J. F., and Neto, W. P. G. (2013). Document-level sentiment classification: An empirical comparison between svm and ann. *Expert Systems with Applications*, 40(2):621–633.

Morin, F. and Bengio, Y. (2005). Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pages 246–252. Citeseer.

Neale, S., Donnelly, K., Watkins, G., and Knigh, D. (2018). Leveraging lexical resources and constraint grammar for rule-based part-of-speech tagging in welsh. *Proceedings of the LREC 2018 Conference.* 3946-3954.

Pipino, L. L., Lee, Y. W., and Wang, R. Y. (2002). Data quality assessment. *Communications of the ACM*, 45. 211-218.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. *CoNLL '09 Proceedings of the Thirteenth Conference on Computational Natural Language Learning.* 147-155.

Rose Marie Nieswiadomy, P. (2012). An overview of qualitative research. *Foundations of Nursing Research*, 6th ed.:46,336.

Rosebrock, A. (2016). A simple neural network with python and keras.

Saldaña, J. (2015). *The coding manual for qualitative researchers.* Sage.

Sandeloweski, D. (1995). Sample size in qualitative research. *Research In Nursing And Health*, 18. 179-183.

Suter, W. N. (2012). Qualitative data, analysis, and design. *Introduction To Educational Research*, 2nd ed.:344.

UnitedNations (2015). 17 goals to transform our world. *Sustainable Development Goals*. Available at https://www.un.org/sustainabledevelopment/health/ [Accessed: 06 February 2019].

Wei, G. and Wei, Y. (2018). Similarity measures of pythagorean fuzzy sets based on the cosine function and their applications. *International Journal of Intelligent Systems*, 33(3):634–652.

WHO (2017). World bank and who. Available at https://www.who.int/news-room/detail/ [Accessed: 18 July 2019].

Yoshida, Zhang, Wen, T., and Tang, X. (2011). A comparative study of tf* idf, lsi and multi-words for text classification. *Expert Systems with Applications*, 38(3):2758–2765.

Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

Zhu, Y., Yan, E., and Wang, F. (2017). Semantic relatedness and similarity of biomedical terms: examining the effects of recency, size, and section of biomedical publications on the performance of word2vec. *BMC medical informatics and decision making*, 17(1):95.